19/9/2024

Incorporating Solar Charging in Routing Problems in Industrial and Urban Areas

Maarten Ipskamp

INDUSTRIAL ENGINEERING AND MANAGEMENT

EXAMINATION COMMITTEE
First supervisor: dr.ir E.A. Lalla
Second supervisor: dr. A. Trivella

Company supervisor: dr. ir. B. Gerrits,
Director of Distribute.

UNIVERSITY OF TWENTE.

Management Summary

This thesis is conducted at Distribute and investigates the incorporation of solar charging into routing problems in both industrial and urban use cases. The industrial area use case is called the SAVED project and facilitates the development of an autonomous transport system at the XL business park in Almelo. The aim of the SAVED project is to let an autonomous electric truck transport containers between an inland terminal and a set of warehouses. The number of container orders per day varies between 10 and 50 with an average of approximately 28. The urban area use case is a hypothetical use case at the Campus of the University of Twente in which electric drones and street robots deliver packages to different locations at the campus. In this hypothetical use case, the number of orders varies between 50 and 400.

The main similarity between these two use cases is that all vehicles in both use cases have a single-unit capacity and can be charged either with energy from the grid or with energy generated by solar panels. Therefore, in both use cases, a problem occurs where the vehicles have to be scheduled for both charging and operating, and the goal is to charge as efficiently as possible by using as much solar energy as possible.

The primary objective is to develop a solution approach that integrates solar charging into the routing problems at both use cases, which means charging as efficiently as possible, while still meeting the time windows of all orders. Therefore, the main research question is:

How can sustainable charging be integrated into Electric Vehicle Routing Problems in industrial and urban areas?

The first step in answering this question is an extensive literature review. This literature review provides a comprehensive overview of relevant topics for this thesis, with emphasis on Electric Vehicle Routing Problems, combined with time dependency. The conclusion was that the literature on the combination of all relevant concepts was scarce and that based on the solution methods of the closest papers, Adaptive Large Neighborhood Search (ALNS) is the best solution approach for our research.

In the solution approach, a conceptual graph is used in which the nodes represent the trips and the edges between two nodes represent the distances between the end and start locations of those specific nodes. A problem formulation is provided fitting both use cases combined with a set of necessary assumptions. Then, a mathematical model is provided for the problem formulation without the use of solar charging. As the main solution approach, a constructive heuristic is used combined with the ALNS. In the constructive heuristic, the trips are evenly distributed over the vehicles and each vehicle charges to 100% when it does not have enough battery to execute the trip. The ALNS then iteratively destroys and rebuilds the solution by including move operations, switch operations, partial charging, charging when it is not necessary, and waiting times before charging, with the main objective being the minimization of charging costs.

Experiments with the ALNS were done with multiple instance sizes, varying in number of orders, number of vehicles and number of solar panels, representing both use cases. In most experiments, the time windows were considered to be hard, while the objectives to minimize were the overall traveling time and the charging costs. First, the parameters of the ALNS algorithm are tuned to ensure that the performance balances efficiently between the quality of the solution and the computation time. The ALNS is tested against the mathematical model for SAVED instances and the conclusion was that the performance of the ALNS was 5.5% worse than the exact approach in terms of traveling time and charging costs, when the number of container jobs is 10 while reaching the solution in less than 3 seconds. On the contrary, the exact optimization method reaches its solution in 461 seconds. When the instances get larger, the exact

optimization does not come to an optimal solution within the time limit, while the heuristic still reaches its solution in approximately 3-4 seconds.

In other scenarios, the ALNS is tested against a ‘No Solar Panel’ policy, in which the use of solar panels is neglected. For the SAVED use case, the performance of the ALNS was on average 26.5% better than the ‘No Solar Panel’ policy in terms of charging costs and traveling time, with the improvements varying between 8% and 44% for different instance sizes. It can also be concluded that collaboration between the companies at SAVED can lead to a further 8.5% improvement in costs. For the Campus use case, the average improvement is 24.8%. In that use case, on average 77% of the energy needed for executing the trips, comes from solar panels, while for some instances this goes up to 82%, while it is only 64% when in the ‘No Solar Panel policy’. Also, variable time windows and different weather conditions are tested to simulate real-world complexities. Furthermore, the option of battery usage is investigated, and a sensitivity analysis is executed, in which the influence of soft time windows and the relative weights of the penalty costs and the charging costs are tested.

The conclusion of the thesis is that incorporating solar charging into routing problems can be done with the help of an ALNS algorithm, and is beneficial for both the industrial and urban use case. The practical contribution of this research is twofold. First, the XL business park can reduce its charging costs by incorporating the solution approach, and second, the method is easily generalizable for other use cases by changing the parameters of the trips, the vehicles, and the solar panels. The contribution to theory is a new mathematical model, which is tested and validated, and an ALNS method incorporating solar charging into routing problems. The biggest limitation is that only vehicles with a single-unit capacity are used. Ideas for future research include incorporating vehicles with multi-unit capacity, an optimization study for the number of solar panels at both use cases, and testing the approach on multiple other but similar use cases, to further test the generalizability of the approach.

Preface

This thesis that lies in front of you is the result of my 6-year journey as an Industrial Engineering and Management student on both a bachelor and a master level. During this period, I was fortunate to gain valuable knowledge, develop myself on a personal level, and connect with many people for interesting conversations and laughs. Now, this journey has come to an end with this research, executed at one of the most unique companies to ever exist.

I would like to thank Berry Gerrits for giving me the opportunity to execute my research at his company Distribute and providing assistance and feedback whenever necessary. Furthermore, I have to thank him for creating a positive and energetic vibe at the company, and even teaching me how to brew beer. I also want to thank former employee Robert Andringa for his valuable assistance and many interesting discussions, some about the research, and some about football. I also want to thank my fellow graduate students at the company, Yoran Nijenhuis and Marijn Schotman, for valuable peer feedback and in-depth conversations. Furthermore, I have to thank the complete company for its positive work vibe and humor, and the many laughs at the Friday (and Monday) afternoon drinks.

Furthermore, I want to express my gratitude to Eduardo Lalla, my lead supervisor, for elevating this thesis to a higher level, with many meetings, discussions, and valuable insights. I also want to thank my second supervisor, Alessio Trivella, for improving my thesis due to his constructive feedback. Lastly, I would like to thank my family, friends, and fellow students for providing me input and support.

Maarten Ipskamp

Haaksbergen, September 2024

Table of Contents

List of Abbreviations

ACO	=	Ant Colony Optimization
ADV	=	Autonomous Delivery Vehicle
AGV	=	Autonomous Green Vehicle
ALNS	=	Adaptive Large Neighborhood Search
CTT	=	Combi Terminal Twente
EV	=	Electric Vehicle
EVRP	=	Electric Vehicle Routing Problem
ILP	=	Integer Linear Program
KPI	=	Key Performance Indicator
LP	=	Linear Program
MIP	=	Mixed Integer Program
MTSP	=	Multiple Traveling Salesman Problem
NS	=	Neighborhood Search
SAVED	=	Samenwerkend Autonoom Vervoer op Bedrijventerreinen
RADR	=	Road Automatic Delivery Robots
RVRP	=	Recharging Vehicle Routing Problem
SADR	=	Sidewalk Automated Delivery Robots
TSP	=	Traveling Salesman Problem
UT	=	Universiteit Twente
VRP	=	Vehicle Routing Problem

List of Figures

List of Tables

1 Introduction

This chapter introduces the research of this thesis by first introducing the research context in Section 1.1. Section 1.2 explains the research problem of this thesis and states the research goal. Section 1.3 states the research questions of our research while Section 1.4 presents the research design of this thesis, combined with an outline of the thesis.

1.1 Research Context

In this section, we provide the research context for our thesis. This thesis is carried out at Distribute. Distribute is a research and innovation company specialized in unmanned systems and smart robotics. It was started by two students in 2016. They create and simulate distributed planning and control systems for unmanned systems in the logistics and transportation industry. They are currently working on multiple projects concerning Industry 4.0, Digital Twinning, and Autonomous Systems (Gerrits, 2016).

This thesis focuses on two different projects in which Distribute is involved which both cover autonomous driving in first- and last-mile logistics in industrial and urban areas. One project is called the SAVED project and facilitates the transformation to the use of an autonomous electric truck at the XL business park in Almelo, while the other project investigates the option of doing autonomous deliveries with the use of electric drones or street robots at the University of Twente.

In Section 1.1.1, we provide context on autonomous green vehicles and electric vehicles in routing problems. In Section 1.1.2 we discuss charging characteristics of electric vehicles because those strategies are relevant to the context of our research. In Section 1.1.3 we provide background on the two use cases that Distribute is involved in.

1.1.1 Autonomous Green Vehicles and Electric Vehicles

This research focuses on the use of AGVs in routing problems. AGVs were introduced in 1955 and are driverless transportation systems used for the movement of materials. AGVs have a wide range of benefits, such as increased productivity, reduced labor costs, and reduced energy consumption, and have application opportunities in manufacturing, healthcare, and logistics (Fragapane et al., 2021).

An advantage of AGVs in routing problems is that it minimizes human intervention in routing problems. In contrast to humans, who need breaks, AGVs can operate 24/7, increasing operational efficiency. It also improves safety, since there are no more human errors in the system. It decreases the chances of accidents, since AGVs are designed with sensors and avoidance systems, and there are no opportunities for human errors such as lack of concentration. They operate more consistently without human operators, which leads to more predictable delivery times and overall logistics planning.

This research also focuses on the use of electric vehicles (EVs) in routing problems. EVs offer a sustainable future for transportation and logistics, by reducing gas emissions and minimizing the dependency on fossil fuels. In routing problems, the use of EVs introduces new challenges. New characteristics should be considered, such as range limitations and the locations of charging stations. However, the inclusion of EVs in routing problems reduces environmental impact and contributes to sustainable logistical solutions.

1.1.2 Charging Characteristics

In our research, EVs are charged with the use of solar energy with the help of photovoltaic panels. These panels convert solar power into electricity. Incorporating solar energy in the charging system makes the system less reliable on the grid because vehicles can be charged without using energy from the grid. Using energy from the grid is the most basic method of charging electric vehicles. The power output from the grid is constant and the charging rate is equal to the price of electricity at that specific moment.

The two most common solar panels are 60-cell panels and 72-cell panels. The 60-cell panels are about 165 by 99 centimeters and have a power output of around 280-320 watts, and the 72-cell panels are about 196 by 99 inches and have a power output of around 340-400 watts (*Solar Photovoltaic Panel Sizes: A Complete Guide*, 2023).

However, the output of solar panels depends heavily on the power of the sun. This means that the output varies per day and during the day. Figure 1.1 (*How Much Electricity Do Solar Panels Produce?*, 2022) shows the output of a set of solar panels for 2 different days. The blue line shows the power output on a day in the summer when there are almost no clouds and then the sun shines at full intensity all day, while the orange line shows the power output on a day with a mix of clouds and sunshine. It shows that on a cloudy day, the output is lower and more irregular.

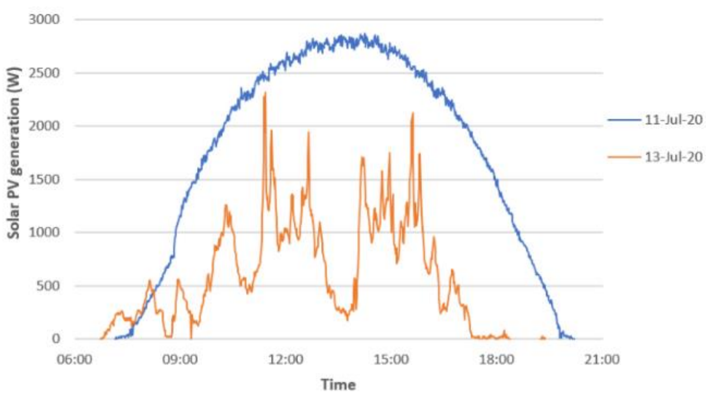

Figure 1.1. Power Output of a Set of Solar Panels for Two Different Days

1.2 Two Use Cases

In this section, we explain the context of the two use cases of our research. The use case representing an industrial area is called the SAVED (Samenwerkend Autonoom Vervoer op Bedrijventerreinen) project. This project aims to develop an automated transport system at the XL business park in Almelo. The XL business park has an inland terminal (CTT), a transport company (Bolk), and two warehouses (Bleckmann and Timberland). The goal of the project is to let one AGV transport containers between the inland terminal and the warehouses where the AGV picks up full containers at the terminal, delivers those at the warehouse, and transports the containers back to the inland terminal when they are empty. The AGV has a capacity of one container and therefore has to drive back and forth between the warehouses and the terminal. All containers have time windows in which they need to be delivered at the warehouses. Besides the truck being an AGV, it is also an EV, which can be charged at the inland terminal with the use of both energy from the grid or from solar panels which are located at the inland terminal. This use case is a small use case with only 1 or 2 vehicles. However, because of the charging characteristics and the vehicle being an AGV, it is perfectly suitable for demonstrating sustainable charging strategies.

The other use case that Distribute is interested in and shares similarities with the industrial use case represents an urban area. This use case is a hypothetical use case at the UT Campus. The UT Campus use case is a use case in which we develop a simulation of a heterogeneous fleet of AGVs that delivers small packages between locations at the UT Campus. This heterogeneous fleet consists of drones and street robots which can also only transport one package at a time. The packages either be distributed from the depot to customers, from a customer to the depot or between customers.

All packages have time windows in which they need to be delivered to the customers, depending on customer preferences. We assume that the fleet has a limited range and can be charged at the depot using solar energy or energy from the grid.

Figure 1.2 shows a graphical representation of both situations. It shows that the biggest similarity between both use cases is that the vehicle can only transport one unit of demand at a time, which means that we can use the same solution approach for both use cases. The industrial area/SAVED use case has fewer customers and required trips, and does not include pickups and deliveries between customers. Only 1 vehicle is used, which can be charged at the depot and has a relatively big range. In contrast, the urban area/Campus use case has more different customers and more required trips. In this use case, a heterogeneous fleet of AGVs is used which have a relatively smaller range so more charging is required.

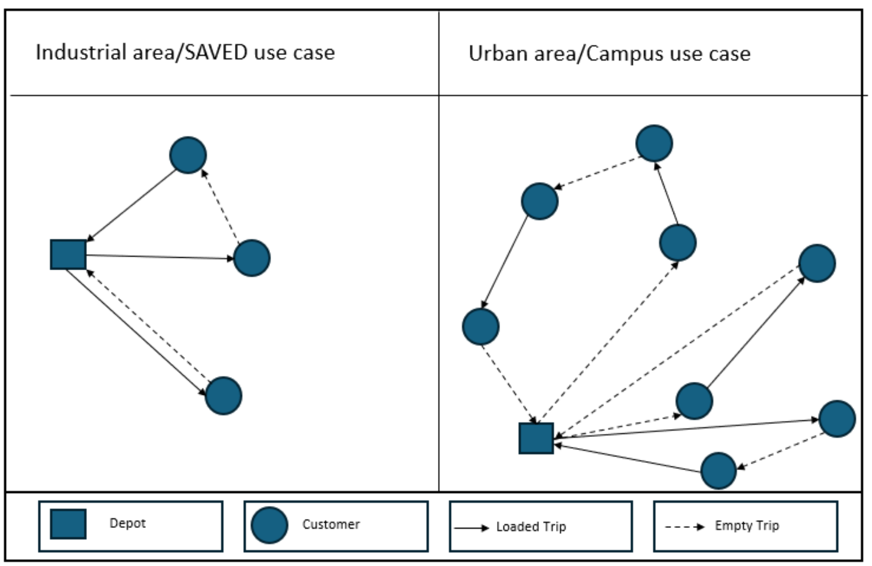

Figure 1.2. Graphical representation of the logistical process at both use cases.

1.3 Problem Statement and Research Goal

The problem that this thesis researches is that a sustainable charging strategy needs to be developed for both the industrial use case and the urban use case incorporating solar charging. This problem is related to an Electric Vehicle Routing Problem (EVRP). The EVRP is a variant of the normal Vehicle Routing Problem (VRP) in which a set of customers orders to be visited by several vehicles as efficiently as possible while incorporating constraints such as capacity constraints or time-window constraints. The EVRP is a variant with Electric Vehicles (EVs). This means their characteristics need to be taken into account. EVs have a limited driving range and there are often limited charging stations available.

If solar charging is included in routing problems, solutions also depend on the power output of the solar cells and the weather forecast for the day. This means that the EVRP model needs to factor in variable energy input for the charging stations, which influences the ideal charging moments. There is relatively little insight into how to develop a logistical strategy for AGVs incorporating the use of solar energy. Although many research has been done on EVRPs (see for example Qin et al. (2021)), there is not much knowledge on EVRP in combination with solar charging.

The main research goal is to integrate sustainable charging strategies in routing problems. This means that we want to come up with a logistical strategy for scheduling and routing AGVs while incorporating charging on solar energy. This consists of a tool, where the parameters can be filled in, such as charging locations, customers, orders, time windows of those orders, and the weather forecast. The output is a strategy consisting of a schedule for the AGVs of when to fulfill the orders and when to charge. We want to experiment with these charging strategies in two environments. We want to experiment in an industrial area and an urban area, since first- and last-mile logistics is an important concept in those two areas. As said in the research background, routing in the first- and last-mile logistics is not done very efficiently since almost 53% of transport costs arise from these logistics. This makes those areas perfectly suitable for experimenting with different sustainable charging strategies.

1.4 Research Methodology

In this section, we define the research methodology, including the research questions, corresponding to our research goal and research approach. We define a main research question and a set of sub-research questions guiding us to provide an answer to the main research question. Our main research question is:

How can sustainable charging be integrated into Electric Vehicle Routing Problems in industrial and urban areas?

This research question is answered in 4 stages. Figure 1.3 shows a graphical representation of this research design with the inputs, research questions, and outputs per chapter.

Stage 1. Literature review

To provide an answer to the main research question, we must start with an extensive literature review of existing EVRP concepts and problems in combination with solar charging and corresponding solution methods. We first explore the classic EVRP, and its relevant concepts and features. We research existing charging methods, the use of a heterogeneous fleet, and the use of autonomous vehicles in EVRP. Furthermore, we need to expand our knowledge of how solar charging and the time-dependency of solar charging influence EVRP models. After we have enough knowledge of all relevant concepts, we examine the closest related researches and their solution methods. This part of the research is executed in Chapter 2, with the following set of sub questions:

1. What is proposed in the literature for modeling and solving EVRP with solar charging?
    a. What can the literature teach us on different concepts and problems in EVRP with solar charging?
    b. What can the literature teach us on solution methods for the EVRP with solar charging?

Stage 2. Problem Formulation & Solution Approach

After the literature review, we should have enough knowledge to design a solution approach. To design this solution approach, we first create a conceptual model and a mathematical formulation of the problem that fits both use cases. Then we describe the solution method, which can be an exact optimization method or a heuristic. In this phase, we also create our tool, which processes the input (for example the locations, the orders, and the weather forecast) to an output with the list of all tasks per vehicle and charging moments. This part of the research is executed in Chapter 3, with the following set of sub questions:

2. How should the solution approach be designed?
    a. What are the requirements necessary for designing the solution approach?
    b. Which assumptions have to be made to design the solution approach?
    c. Which solution methods are most suitable for solving the EVRP in both use cases?

Stage 3. Solution Evaluation

In this stage, we experiment with our solution approach in both use cases. For the industrial use case, we do a small context analysis of the SAVED project. The goal of the context analysis is to conceptualize the use case so that we can create parameters and experimentation instances. For the urban use case, we have to come up with hypothetical data instances, reflecting different demands at the campus. We can then test our solution approach for those instances with different experimentation settings. This research stage is executed in Chapter 4 with the following set of sub questions:

3. How does the solution perform for different experiments in industrial and urban use cases?
    a. How can we parametrize the use cases to input for our simulation?
    b. What are the different experiments for testing our solution strategy?
    c. How does our solution strategy perform for our considered experiments?

Stage 4. Conclusion

After experimenting with our designed solution method, we can interpret the results and investigate what the benefits of our solution design are quantitatively for both the XL business park and the UT-campus use case. After this last step, we should have enough information to draw conclusions about our solution approach and provide recommendations regarding the implementation of the solution approach. In this step, we also answer our main research question and conclude whether we reached our research goal which is implementing sustainable charging in AGV routing and scheduling. This part of the thesis is executed in Chapter 5 with the following set of sub questions:

What can we conclude from the results of our experiments?

a. What can be concluded and recommended for the XL business park use case based on the results of our experiments?
b. What can be concluded and recommended for the Campus use case based on the results of our experiments?
c. What should be researched further based on the results of our research?

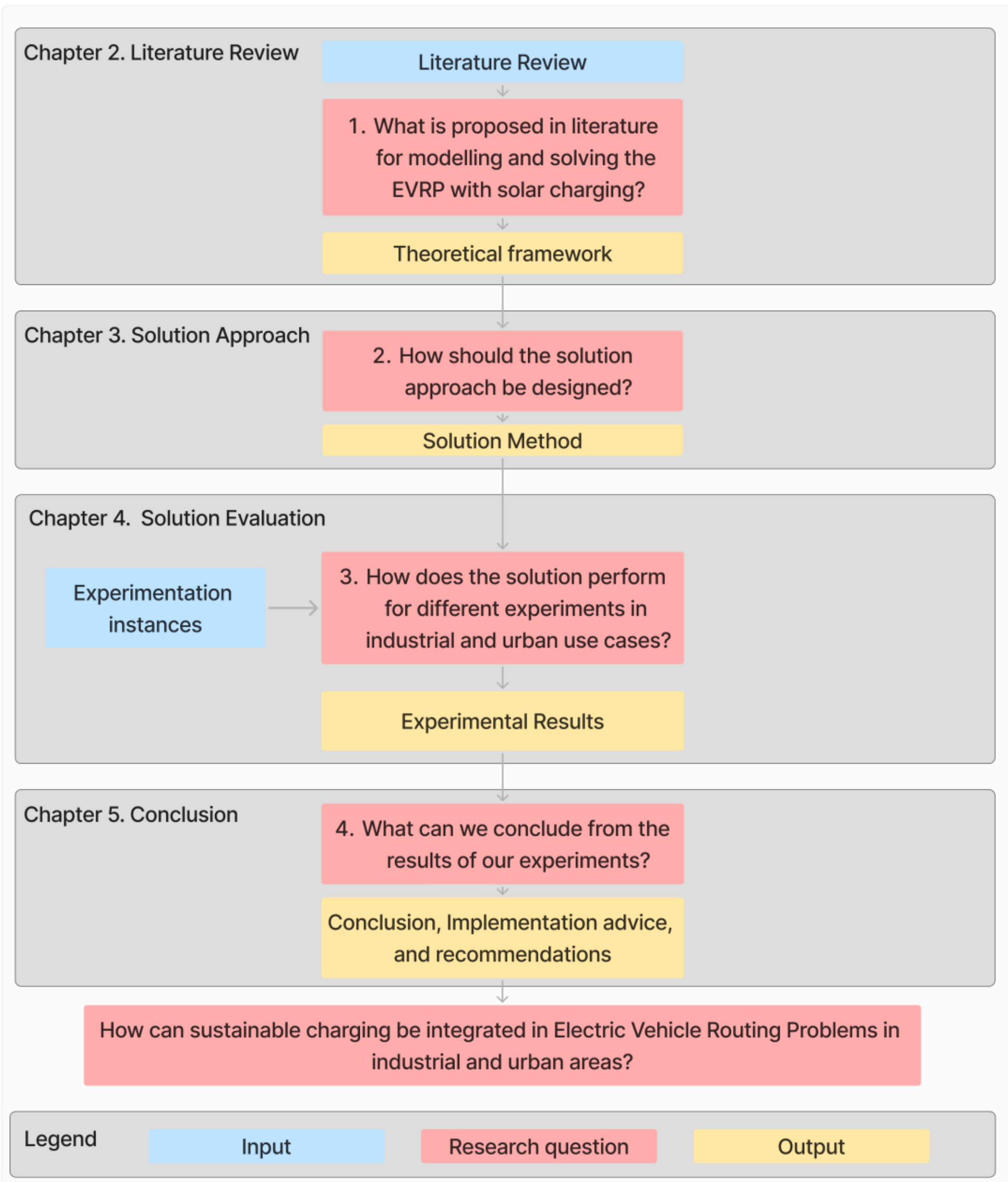

Figure 1.3. Graphical representation of Research Design

2 Literature Review

In this chapter, we answer the following research question and subquestions:

What is proposed in the literature for modeling and solving EVRP with solar charging?

a. What can the literature teach us on different concepts and problems in EVRP with solar charging?
b. What can the literature teach us on solution methods for the EVRP with solar charging?

In Section 2.1, we discuss the different concepts and problems within EVRP in combination with solar charging, autonomous delivery, and truck scheduling, while in Section 2.2 we discuss possible solution methods.

2.1 Concepts and Problems

2.1.1 Multiple Traveling Salesman Problem

The Multiple Traveling Salesman Problem (MTSP) is an extension of the Traveling Salesman Problem (TSP). In a TSP, the objective is to find the shortest route while visiting a set of customers exactly once and returning to the original city. In the MTSP, there are multiple salesmen who each visit a subset of customers while making sure that the complete set of customers is visited exactly once, with the objective of minimizing the total distance of all salesmen together. This problem differs from the classic Vehicle Routing Problem (VRP), because in the VRP other constraints such as capacity constraints are involved. Since our use cases consider vehicles with a single-unit capacity, we could model the use cases as a MTSP, where the trips are the customers, instead of a VRP. However, because there is relatively little few literature on MTSP combined with electric vehicle constraints, we move our focus to the VRP. For an extensive literature overview of all MTSP-related problems, we direct the reader to Cheikhrouhou & Khoufi (2021).

2.1.2 Classic Vehicle Routing Problem

The Vehicle Routing Problem (VRP) was first introduced by Dantzig & Ramser (1959). They created a model which is called the Truck-Dispatching problem. This is an extension of the Traveling Salesman Problem (TSP). The model aims to find a set of routes for a homogenous fleet of vehicles that visit all customers and satisfy all their demands without exceeding the capacity of the vehicle. Clarke & Wright (1964) then expanded the research field by developing a heuristic to improve the method used by Dantzig & Ramser.

The goal of a VRP is to produce a set of routes for a set of vehicles starting at a depot in such a way that each customer is served, and no capacity constraints are exceeded. This means that it is an extension of the TSP so that the customers can be divided over the vehicles.

Laporte et al. (1985) introduced a mathematical model of the VRP considering the capacity and distance constraints. They also include multiple subtour prevention constraints. Nowadays there are multiple formulations for the VRP (Munari et al., 2016). The most common formulations are either vehicle flow formulations (Toth & Vigo, 2002, Elatar et al., 2023) or set-partitioning formulations (Agarwal et al., 1989).

These models are the basics of many variants in a broad research field. Figure 2.1 shows a taxonomy of relevant features for our research. This taxonomy is based on the taxonomy of Eksioglu et al., (2009), but only shows the relevant features for our research. We distinguish the features for both use cases. If the feature is marked yellow it fits with our industrial use case, if it is marked orange it fits with the urban use case and if the feature is marked green it fits with both use cases. The first feature is whether the number of stops on the route is known. This is the case for both use cases since we assume that all information is known upfront. Feature 2 determines whether the splitting of the load is allowed. This is not the case, since the load of our vehicle routing problem consists of containers in the industrial use case and packages in the urban use case. Feature 3 specifies whether the demand quantity is deterministic or not. The demand quantity in our research is deterministic and also static (Feature 4) since we assume everything is known upfront.

Feature 5 represents the time horizon. In our use case, the time horizon is only 1 period since we only solve the VRP for one day at a time. In our experiments, we run multiple days, however, for each day we solve the problem independently. We also work with only 1 depot in both use cases (Feature 6). Feature 7 indicates whether the research involves backhauls. Backhauls occur when customers do not only need goods to be delivered but also have goods that should be picked up. This does not happen in our urban use case, however, it happens in our industrial use case but not necessarily simultaneously. For a complete overview of VRP with backhaul with different variants, we direct the reader to Koç & Laporte, (2018).

Feature 8 states whether there is a specific number of vehicles and if all vehicles should be used. This is the case for both our use cases. In the industrial use case, only 1 or 2 vehicles are used, while in the urban use case, multiple vehicles are used. These vehicles are capacitated since they can only transport one container/package at a time (Feature 9) and are homogeneous in the industrial use case and heterogeneous in the urban use case (Feature 10). We also assume that there are service times included in our research because it takes time to couple and decouple containers or packages (Feature 11). These service times are deterministic since (de)coupling takes approximately the same time for each container/package.

The last two features contain the involvement of time windows in our research. In this variant, each customer has a time window in which it should be visited. Time windows can be soft or hard. Time windows are called hard when they cannot be violated. Time windows are called soft if they can be violated, however this leads to penalty costs. This means that the problem relaxes to a normal VRP however with a different objective, namely including penalty costs (Kallehauge et al., 2005). Our research consists of a mix of hard and soft time windows since the beginning of each time window cannot be violated. This would lead to an infeasible solution since the containers/packages cannot be transported before they are ready. The end of the time windows may be violated in order to create feasible solutions, however, this leads to penalty costs.

1. Number of stops on route	**8 Number of vehicles**
1.1 Known	8.1 n Vehicles
1.2 Partially known	8.2 Up to n Vehicles
2 Load splitting constraint	8.3 Unlimited number of vehicles
2.1 Splitting allowed	**9 Capacity Consideration**
2.2 Splitting not allowed	9.1 Capacitated vehicles
3 Customer service demand quantitity	9.2 Uncapacitated vehicles
3.1 Deterministic	**10 Vehicle Homogeneity**
3.2 Stochastic	10.1 Homogeneous vehicles
3.3 Unknown	10.2 Heterogenenous vehicles
4 Evolution of information	**11 Service Times**
4.1 Static	11.1 Deterministic
4.2 (Partially) Dynamic	11.2 Time Dependent
5 Time Horizon	11.2 Vehicle Type Dependent
5.1 Single Period	11.3 Stochastic
5.2 Multi Period	11.4 Unknown
6 Number of Depots	**12 Time window type**
6.1 Single Depot	12.1 Restriction on customers
6.2 Multiple Depots	12.2 Restriction on roads
7 Backhauls	12.3 Restriction on depot
7.1 Only Linehaul	12.4 Restriction on drivers
7.2 Simultaneous pick-up and delivery	**13 Time Window Structure**
7.3 Either Linehaul or backhaul but not both	13.1 Hard
7.4 Linehaul and backhaul but not simultaneous	13.2 Soft
	13.3 Mix

Industrial use case	Urban use case	Both use cases

Figure 2.1. Taxonomy with relevant VRP features for our use cases.

2.1.3 Electric Vehicle Routing Problem

Over the last few years, more variants of the VRP have been researched. One of the more recent variants is the Electric Vehicle Routing Problem (EVRP). The EVRP is a variant of the VRP, which is an electric vehicle with a limited driving range. This means that the charging of those vehicles needs to be included in the model. The objectives that have been considered are the minimization of the driving distance or travel time, the minimization of the number of vehicles used, or the minimization of costs which include driving costs, charging costs, or penalty costs for missing the time windows (Qin et al., 2021).

Figure 2.2 shows an example of a solution of the EVRP with charging stations. There are three tours and that the long tours 2 and 3 need visits at charging stations to keep the battery level positive. In contrast to customers, charging stations can be visited multiple times and by different vehicles (Ghorbani et al., 2020).

The EVRP was first studied by Conrad & Figliozzi (2011). They let their vehicles charge the customer during their trip. Their problem was called the Recharging Vehicle Routing Problem (RVRP). They created a mathematical model for this problem and solved it for small instances with a heuristic. The next research was done by Erdoĝan & Miller-Hooks (2012). They called their problem the Green Vehicle Routing Problem, and they experimented with a heterogeneous fleet of Alternative Fuel Vehicles (AFVs), with different driving ranges and a set of refueling stations. They created a Modified Clarke and Wright heuristic, together with a new constructive heuristic and an improvement heuristic.

The most common extension of the EVRP is the EVRP with time windows. This was first done by Schneider et al. (2014). They created a model including time windows and recharging stations. They solved their problem using a metaheuristic which combines a Large Neighborhood search with a Tabu Search. Meng & Ma, (2020) researched the EVRP with soft time windows. In this case, penalty costs were involved in the time windows for customers were not met. For an extensive overview of all problems and models in the world of EVRP, we direct the reader to Fernández Gil et al., (2022)

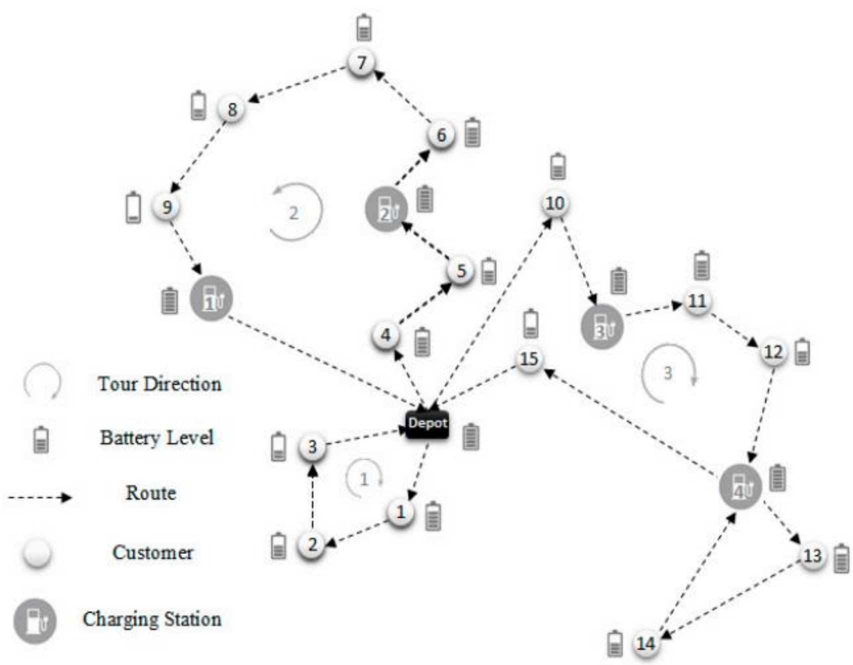

Figure 2.2. Graphical Representation of a feasible solution to an EVRP

In this section, we discuss different features of EVRP which are relevant to our research. First, we discuss different charging methods. This is relevant to our research since we experiment with these charging methods. Then we provide insight into different types of fleets used in the EVRP. Variants of EVRP exist with a homogeneous fleet of EVs, in which all EVs have the same parameters. However, in other variants of EVRP, the set of EVs is heterogeneous. This means that EVs have different parameters, such as charging rate, speed, and consumption model. We also discuss models in which the fleet consists of a combination of conventional vehicles and electric vehicles. Then we discuss the use of solar charging for electric vehicles. This is highly relevant to our research since the electric vehicles used in our use cases are charged with the use of solar energy. Another feature we discuss in that section is the inclusion of time dependency in the EVRP models. This feature is applied in our research since the research takes into account that the charging rate is lower at periods with sunshine in contrast to cloudy periods. This means that our EVRP is time-dependent.

Recharging Methods

There exist multiple recharging methods for the EVRP. In this section, we discuss the full charging method, partial charging method, and battery swapping method. We also discuss the difference between linear charging and non-linear charging.

- Full Charging. The most commonly used method is the full recharging method. In this case, the EVs are recharged until their battery is completely full (see for example Afroditi et al., (2014)). An advantage of this method is that there are fewer decision variables in the model, which leads to a shorter computation time. A disadvantage is however that it might not be the most optimal solution for an EV to recharge to 100% when it can also recharge less and still complete its tour.
- Partial Charging. If partial charging is used as a charging method, it means that the battery does not have to be fully charged at the recharging station. This means that the charging amount/time becomes a decision variable in the EVRP. Felipe et al., (2014) were the first to model an EVRP with partial charging. They presented several heuristics, either constructive or local search heuristics within a non-deterministic Simulated Annealing framework. Keskin & Çatay, (2016) combined the EVRP with partial charging with time windows. They formulated the problem as a Binary Mixed Integer Linear Problem and solved it using an Adaptive Large Neighborhood Search. Desaulniers et al. (2016) researched four different variants of the EVRP, namely:

    1. full recharging policy with one recharge per route allowed
    2. full recharging policy with multiple recharges per route allowed
    3. partial recharging with one recharge per route allowed
    4. partial recharging with multiple recharges per route allowed

    They solved all four different variants to optimality for instances up to one hundred customers and twenty-one recharging stations using a branch-and-cut optimization method.

- Battery Swapping. Another recharging option is the use of battery swapping stations. In these stations, the empty battery can be swapped for a full one. This was first considered by Yang & Sun, (2015), however they put more emphasis on locating those stations. Verma, (2018) proposed an EVRP with battery-swapping stations and time windows. At each charging station, the vehicle could be charged traditionally or receive a full battery. In this model, charging was cheaper, while battery swapping was quicker. A model was formulated, together with an algorithm to solve the problem.

Mao et al. (2020) proposed a model in which there were two options at each recharging station, namely partial recharging and battery swapping. This was different from the model of Verma (2018) because in this new model, the vehicles could be charged partially instead of only full charging. They formulated a Mixed Integer Program and used an improved Ant Colony Optimization method to solve their problem.

- Non-Linear Charging. Next to different recharging options, there are also two types of recharging functions. Almost all research described until now was executed using a linear charging function. This means that the charging speed is independent of the battery level. In real life, this is not the case since the charging speed is lower as the battery level approaches 100%. The first research on non-linear charging was done by Montoya et al., (2015). They considered a partial charging method and modeled a concave function of charging time. They presented a computational study of a comparison of this non-linear charging method with the traditional linear charging method. Later, Montoya et al., (2017), expanded on their previous study by solving the EVRP with non-linear charging with the use of a hybrid metaheuristic. They concluded that neglecting the nonlinear charging process may lead to expensive or infeasible solutions.

In our research we use partial charging as the recharging method since it leads to a larger solution space. We do not use non-linear charging because the we assume the charging speed to be linear with the variability in the charging costs.

Mixed Fleet

Another characteristic of the EVRP is whether a homogeneous fleet is used or a mixed fleet. Within the mixed fleet category, there exist two subcategories namely a mixed fleet consisting of only EVs and a mixed consisting of both EVs and conventional vehicles.

- Only EVs. The first subcategory consists of a mixed fleet consisting of only EVs. In this case, problems combine different types of EVs with different features, such as load capacity or driving ranges. Hiermann et al., (2016) combined the mixed EV fleet with the EVRP with time windows. In this case, EVs differed on aspects such as load capacity, battery capacity, charging rate, and power consumption. They solved their problem to optimality with the use of a branch-and-price method and proposed a hybrid heuristic as well, combining an Adaptive Large Neighborhood Search with a local search. Sassi et al. (2015) came up with a Tabu Search to solve the mixed fleet EVRP. They let their vehicles charge at either the depot or at recharging stations with different charging technologies.
- EVs and Conventional vehicles. The other subcategory is the mixed fleet with different variants of vehicles which does not necessarily have to be EVs. Sassi et al., (2014) proposed a formulation including a mixed fleet of conventional vehicles and EVs while also considering time-dependent charging costs. Lebeau et al., (2015) also developed a model including a mixed EV and conventional fleet, consisting of different features such as load capacity, battery capacity, and driving costs, while recharging was only possible at the depot. Goeke & Schneider, (2015) created a model with a homogenous set of EVs and a homogeneous set of conventional vehicles. They also included an energy consumption model, incorporating speed and the weight of the cargo load distribution. They used an Adaptive Large Neighborhood Search algorithm to experiment with their model.

Macrina et al., (2019) combined the problem of a mixed fleet with time windows and partial recharging. They proposed an iterative local search heuristic to optimize the routing and charging strategies.

Solar Charging and Time-Dependency

In this section, we provide insight on the inclusion of solar charging in the EVRP in combination with time-dependency and the use of time-of-use pricing in the EVRP models. These concepts are combined in one section because they are related to each other because the average output of solar panels over the day is time dependent. At noon, the average solar panel output is higher than for example at the end of the afternoon.

- Solar Charging. Figure 2.3 (Hossain et al., 2020) shows the average power production of a 1 kW solar panel per hour of the day. It shows that the average peak output is approximately 550 W and occurs between noon and 1 pm. This is logical since the sun has the most intensity at those times, while solar panels generate almost no energy at all between 7 pm and 6 am. It shows that even in the peak moment, the average available solar power is not close to the available power output of 1 kW. This means that the output is highly dependent on the weather on a certain day, as Figure 1.1 shows.

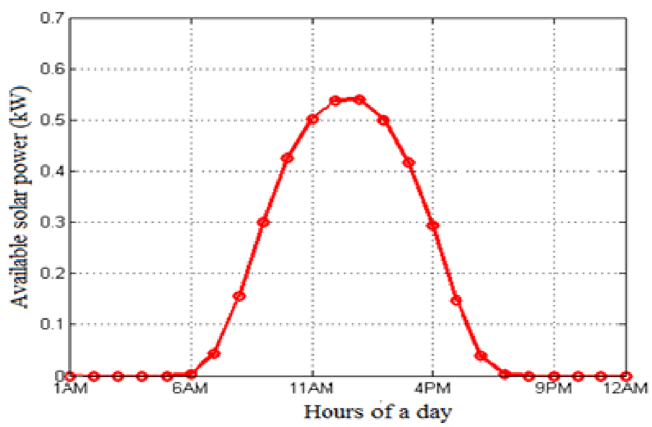

Figure 2.3. Average power output for a 1 kW solar panel over the day

Figure 2.4 shows the seasonal variability of the output of a solar system. As a specific example, the solar production from a 5 kW Solar System in Australia is shown. It shows that the output in the summer is higher than in the winter and that in the summer the solar system produces output from approximately 5 am to 6 pm, while in the winter the output production starts at 8 am and ends at 4 pm. A conclusion that can be drawn is that the power output of a solar system, is highly dependent on the weather, the time of the day, and the period of the year. Other factors influencing the output of solar systems are the location of the system, the incline of the panels, and the exposure of the panels (Lugo-Laguna et al., 2021).

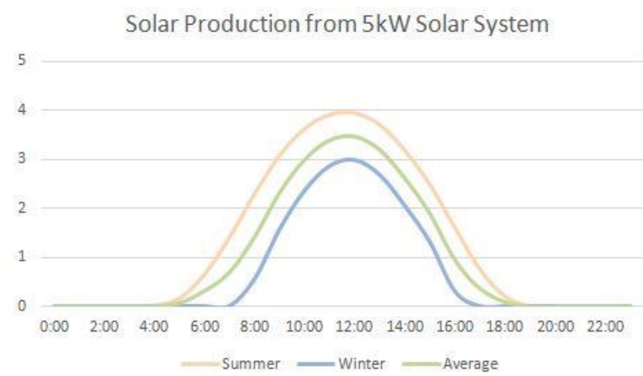

Figure 2.4. Seasonal difference of the power output of a 5 kW Solar System

There has been little research on combining EVRP with solar charging. Most research in this category has been done on the use of different charging methods in one problem. These methods differ in charging speed and charging rate. This has for example been done by Fan et al. (2023). They developed a Mixed Integer Model with three different types of energy generation, namely traditional electrical energy, wind-based electric energy, and solar-based electric energy. They developed an Ant Colony Optimization heuristic to solve the problem. Earlier, Keskin & Çatay (2018) developed a model using partial charging considering multiple charging speeds, which they solved using a matheuristic.

- Time-Dependency. For our problem, the relevant part of the use of solar charging is the time-dependency of solar power. This means that Time-Dependent EVRP is a relevant variant of the EVRP. In a Time-Dependent EVRP, some parameters of the EVRP are dependent on the time. This could be driving costs, speed, charging speed, or charging costs.

The first to include time-dependency in the EVRP was Sassi et al. (2014). In their case, the parameter dependent on the time was the charging costs. They also included different charging methods with three different levels of charging, where in the lowest level the charging rate was very low, and in the highest charging level, the charging speed was higher. They also included time windows for both the customers and the charging stations. They provided a Mixed Integer Programming model and developed a couple of heuristics including a Charging Routing Heuristic and a Local Search Heuristic with three different insertion strategies.

Shao et al. (2017) also included time-dependency in the EVRP. They used variable traveling time dependent on the time to reflect a dynamic traffic environment. Equally to Sassi, they also use three different charging levels. Lu et al. (2020) also modeled a time dependent EVRP including time windows. They could solve their problem to optimality for small instances with an Integer Linear Problem and created an Iterated Variable Neighborhood Search heuristic to solve bigger instances. They also optimize the speed and departure time on each arc of the route.

Zhang et al. (2022) also used time-dependent travel speeds and included time-dependent congestion tolls. They divided their model in only three periods, however, in these periods, the time-dependent variables are both the travel speed and the traveling costs. They provided a mixed-integer linear programming method, an Adaptive Large Neighborhood Search heuristic, and experimentations with this heuristic compared to traditional optimization software.

- Time-of-use electricity pricing. Ham & Park (2016) were the first to include time-of-use pricing of electricity in the model EVRP. Their retail prices vary hour-by-hour to simulate changes in wholesale prices. Their objective is to minimize the electricity costs together with the number of used vehicles and total travel distance. In their solution, which they developed with the use of constraint programming, the charging of the battery is done mostly in the off-peak periods. Later, Liu et al. (2018) used time-of-use pricing in their model as well. They created a reserving charging decision model for EVs that are in need of charging services, considering traffic conditions, and charging resources with the objective of minimizing driving time and charging expenses. However, they did not combine their model with energy-efficient routing.

  Ferro et al. (2018) did combine energy-efficient routing with time-of-use energy prices. They also used the possibility of partial charging, time windows, and an energy consumption model including the load of the vehicle, the travel speed, and route congestion conditions. They also included a maximum power level at the recharging stations to model those stations more real. They presented a Mixed Integer Model and a preprocessing algorithm to reduce the problem dimension. However, they could only solve small instances of the problem, with a customer limit of fifteen, to optimality. They concluded that to increase the instance size, one should switch to either a matheuristic or a metaheuristic such as an Adaptative Large Neighborhood Search.

  Kumar et al., (2023) built on the work of Ferro et al. (2018), and included charging flexibility by using the possibility of battery swapping and multiple charging levels. This is to the best of our knowledge the most integrated model of the Time-Dependent EVRP since it integrates capacity constraints, time windows, different recharging methods (battery swapping, partial recharging, and different power levels), and time-of-use pricing. They created a Mixed Integer Linear Program and solved it for big instances with the use of a matheurstic, which is a variant of the Ant Colony Optimization algorithm.

In our research, the time dependency feature is highly relevant. The reason for this is that the charging costs depend on the time of the day. Since there is more solar energy available around noon, charging is cheaper around noon and the solution approach has to account for that. So the variable that is time dependent is the charging costs. However, these costs do not only depend on time but also on the number of vehicles charging at the same time.

2.1.4 Autonomous VRP

This section focuses on autonomous VRP. This is relevant because the vehicles in our use cases are both autonomous vehicles. Autonomous vehicles are vehicles that do not need manual driving to deliver their packages. Examples of these vehicles include delivery robots and drones.

Delivery Robots/Vehicles

The first type of autonomous vehicles we discuss is delivery robots/vehicles. Delivery robots fall into one of the following categories: Sidewalk Automated Delivery Robots (SADR), Road Automatic Delivery Robots (RADR), and Autonomous Delivery Vehicles (ADV). According to Srinivas et al. (2022), we can divide the autonomous VRP into two major categories, namely the autonomous driving-only problem and the hybrid problem.

- Using automated driving robots/vehicles only, customers still need to be present at their location to pick up the package from the delivery robot. To model this feature, most problems include time windows for the customer. Sonneberg et al., (2019) proposed a MIP formulation for the routing problem including hard time windows. In their problem, they used autonomous unmanned ground vehicles as the vehicles in their problem. They integrated it with a location problem for the depot and they solved a small case study with three depots and ten customers. Gnegel et al., (2021) solved the problem using soft-time windows. Their unmanned vehicles were street robots with a single-unit capacity. This means that the depot needed to be visited after each trip. They modeled the problem as a Mixed Integer Quadratic Problem, and they minimized the sum of the quadratic penalty costs to ensure that larger delays are penalized more heavily. They then solved their problem with a Branch & Refine algorithm and an Iterative Refinement algorithm.
  Instead of using time windows, Ulmer & Streng, (2019) used pick-up locations as customers in their model, where the end-customers can later pick up their package using an unique access code. They solved a dynamic problem with autonomous vehicles, to ensure same-day delivery when the customer orders their package. They solved their problem with the help of a Policy Function Approximation algorithm and can optimize a problem with 10 robots and 1000 orders across 12 pick-up locations.
  Reed et al. (2022) used autonomous delivery vehicles in their problem to transport delivery personnel to places close to customer locations, then the delivery person makes a tour to visit a set of customers, and the personnel is later picked up by the autonomous delivery vehicle and transported to other customer locations. They modeled their problem as an Integer Problem and they showed that using this concept, they could reduce the delivery times by 30%.
- Hybrid Problem. In the hybrid problem, both conventional vehicles and autonomous vehicles are used in one problem, with different variants such as a two-tier model in which a conventional truck transports several packages in the first tier to local hubs, from which the autonomous vehicles deliver the packages to the end-customers in the second tier. Another variant is called the mothership model in which a conventional vehicle transports a set of small autonomous vehicles together with the packages, and deploys them at certain locations. The autonomous vehicles then transport the packages to the end customer. Another variant is called the platoon model in which autonomous vehicles independently deliver packages in zones that are friendly to autonomous vehicles, however, they follow a conventional vehicle to guide them through non-autonomous friendly zones. In both our use cases, the only vehicles that are used are autonomous vehicles, so

there is no hybrid problem. However, for a review of all hybrid problems, including subcategories we direct the reader to Srinivas et al., (2022)

Drone Routing

The second type of autonomous vehicle relevant to this research is drones. Drones are unmanned aerial vehicles. Since they can travel through air, they are quicker than street robots. They most often have a limited flying range, which makes them mainly suitable for first- and last-mile delivery. However, because of their limited capacity, the routing problems most often become multi-trip problems. Within drone routing, there are also two major categories: drone-only routing and drone-truck routing. However, in our research, we do not consider a problem with both trucks and drones, so we focus on the drone-only routing problem. For an extensive overview of drone-truck routing problems, we direct the reader to Liang & Luo (2022).

Schonfeld & Choi (2017) focused on a drone-only routing problem, where a drone can carry multiple packages within a certain limit. They used numerical optimization to optimize the number of drones for a service area to minimize the total costs of the system. The study shows that drones are more economical in high-density areas. Dorling et al., (2017), proposed two VRP models for the drone routing problem. One minimizes costs for a certain delivery time limit, so it minimizes the number of drones and their charging costs. In contrast, the other minimizes the overall delivery time for a certain budget. They modeled the energy consumption linear with the payload and battery weight and used that approximation to develop a Mixed Integer Program for the VRP. They solve their problem with the Simulated Annealing method.

Cheng et al., (2020) formulated a multi-trip drone routing problem with energy function. This means that this problem is a combination of an EVRP and a drone-routing problem. They included a non-linear energy consumption function, with the payload and travel distance as parameters. They modeled the problem as a 2-index problem and developed a branch-and-cut algorithm to solve the problem.

2.1.5 Container Drayage Operations

Since the transport method at the XL business park is the transport of containers between companies, it is relevant to research container drayage operations. In container drayage operations, there are multiple customers who either demand a container or have a container ready to be picked up. A set of trucks drives across the area to transport the containers as efficiently as possible. This is a variant of the VRP however with strict capacity limitations since a truck cannot often transport multiple containers at a time.

Figure 2.5, made by Schulte et al. (2017), shows multiple variants of the container drayage problem. The most basic variant is the bilateral problem. In this variant, containers can only be transported one at a time, and only one customer can be served before going back to the depot. In the triangular problem, a truck can still not carry more than one container, however, it can visit two containers in one route. It can occur that the first customer demands a container from the base, while the second customer has a container ready to be picked up and transported back to the depot. This is what happens in the second variant shown in. In the third variant, one customer both demands a container from the depot and has a container ready to be picked up for another customer. So in this case, intercustomer demand is involved. In the fourth case, the rectangular case, there is also intercustomer demand, however, no customer both demands a container and has a container ready to be picked up. In the last case, the truck has a capacity of more than one container, so it can drive longer routes to visit multiple customers in one tour. (Schulte et al., 2017)

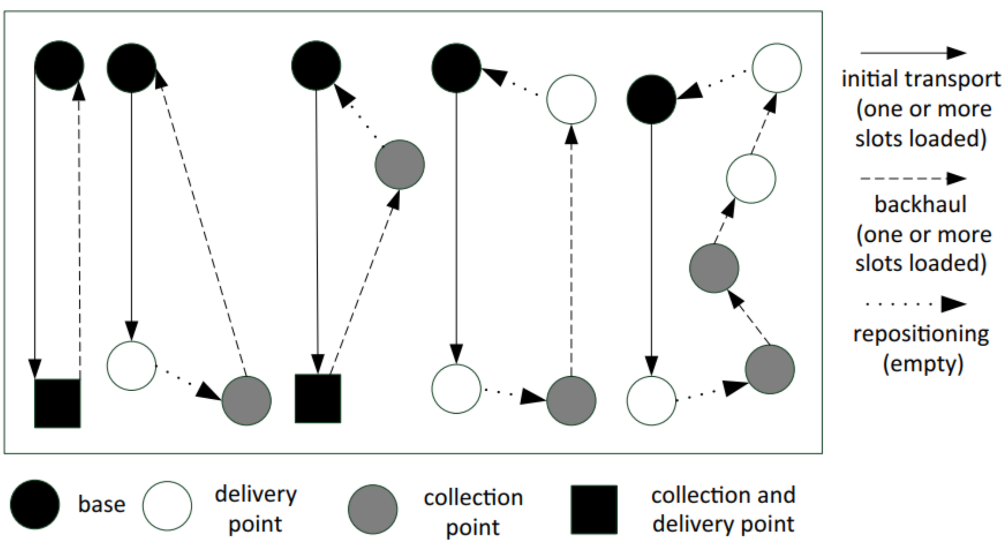

Figure 2.5. Graphical representation of multiple variants of the Container Drayage Problem, (Schulte et al., 2017)

The variant that is most suitable for the XL business park is the second variant. The containers at the XL business park either need to be picked up at the depot and delivered to the customer or picked up at the customer and delivered at the depot. So there is no intercustomer demand. However, it might be the case that customers are both a delivery point and a collection point, so in that way, it fits more with the third variant.

One of the first research on this container drayage problem was done by Wang & Regan, (2002). They model the problem as a MTSP with time windows. They considered a set of loads that must be moved in a local area, so including intercustomer demand. They consider each trip as a node in a traveling salesman problem and develop a mathematical formulation to model the problem on a daily basis. They use an iteration heuristic to solve this problem in which time constraints are replaced by binary flow variables.

Jula et al. (2005) developed a similar model and used multiple approaches to come to a solution. They first solved the problem to optimality by using dynamic programming. However, when the problem becomes too big, they switched to a metaheuristic including a Genetic Algorithm. Ileri et al., (2006) were the first to include the repositioning of empty containers in their model. The repositioning of empty containers was required to facilitate the transport of loaded containers. The transport was executed by a heterogeneous fleet. They created cost-effective schedules with the use of column generation which is an exact method. Xue et al. (2014) solved a similar problem with empty containers. In this case, the truck could be detached from its trailer and assigned to a new task. They solved their problem with the use of a Tabu Search algorithm.

Schulte et al., (2017) created a mathematical model of the container drayage problem in a terminal. In their case, the jobs were nodes in a graph and the distance between the nodes depended on the start- and end location of each job. They divided their container jobs into different categories based on whether they were import or export containers and full or empty containers. Then they could calculate the service time of each job depending on its category and the travel time. Then they created an asymmetric distance matrix based on the different start and end locations of each job and the category of the job. They solved their mathematical problem with the use of a commercial solver.

R. Zhang et al. (2015) solved a problem where a truck could transport more than one container at a time. This means that the truck can drive longer routes before returning to the depot. They modeled the problem as a multiple traveling salesman problem and solved their problem with the use of a Tabu Search. They did however not take time-windows into account. Vidović et al. (2017) addressed a problem in which time windows are taken into account and the trucks could carry two containers. They created a mixed-integer model for small instances and solved larger instances with the use of a variable neighborhood search.

Heilig et al., (2017) connected the container drayage problem with an interterminal transport. Their case fits our research as well since our research takes place in a semi-closed environment which can be seen as a terminal. They did not take a depot into account, however, their trucks have initial starting locations. This means that they only take intercustomer demand into account, with the inclusion of soft time windows. They proposed two greedy heuristics and two hybrid simulated annealing algorithms, which they tested using real locations in the port of Hamburg.

2.1.6 Combination EVRP and Container Drayage

Since the SAVED project involves an electric truck delivering containers, it is relevant to look into the combination of EVRP and container drayage problems. The closest research to our problem was done by Dessouky & Yao, (2023). They called their problem the mixed fleet drayage routing problem. This means that they considered a heterogeneous fleet consisting of both normal trucks and trucks driven by electricity. Their work was based on the study of Giuliano et al. (2020). This was a simulation study in which trucks might only have one or two stops outside the depot and the depot was the only charging location. There was no intercustomer demand involved in this problem, however, they did consider a heterogeneous fleet consisting of both conventional trucks and EV trucks.

Dessouky & Yao, (2023) formulated their problem as a mixed integer problem. They considered that only the electric trucks need to be charged during working hours. The battery consumption rate depends on whether the container is empty or loaded and the weight of the load. They also included non-linear charging times. The objective of the model is to minimize a combination of the charging costs and the emission costs of both types of trucks. The model can be solved with commercial optimization software for small instances. For bigger instances, they created a large neighborhood search algorithm, which can solve the problem for instances with more than a hundred units of demand.

2.1.7 Table of Relevant Works

Table 2-1 shows an overview of all relevant works discussed in this literature review, together with their characteristics. The left column shows the authors of the articles together with their year of publication. Then the next column shows whether the problem concerns EVRP and/or Autonomous Vehicle Problem and/or Container Drayage Problem. For the EVRP papers, the characteristics of the problem are provided in the next section of columns, namely the charging method (CM), whether the charging time is linear or non-linear (CT), the charging location (CL), the fleet and traditional VRP characteristics (VRPC). The Autonomous Fleet Characteristics (AFC) are provided in the column after the EVRP characteristics and the Container Drayage Characteristics (CDPC) are presented in the columns after the Autonomous Fleet Characteristics. The second-to-last column shows the objective of each column, while the last column shows the solving method categorized in either an exact method, a heuristic, a metaheuristic, or a matheuristic in combination with a mathematical model (MIP).

Using this table, we can select the closest works related to our research. In the next section, we look at the solution methods of these selected works and choose the solution method for our solution approach. From the EVRP problems, we select the work of Ferro et al., (2018), and Kumar et al., (2023). These two works took time dependency into account with time-dependent charging costs, which perfectly fits our research. They also take partial charging into account as their charging method and hard time windows. From the Autonomous Vehicle Problem, we select Cheng et al., (2020) since they incorporate an energy function and the recharging option in their problem. From the Container Drayage Problems, we select Schulte et al. (2017) as a close work since they also included the repositioning of empty containers as a characteristic. This fits with our research since we also take into account empty containers that should be transported back to the depot. They also included hard time windows and their problem does not allow for transporting multiple containers at the same time, which fits our problem. They also did not include intercustomer demand.

We also select the work of Giuliano et al., (2020) and Dessouky & Yao (2023) since they combined the EVRP with the Container Drayage Problem. Giuliano even considered the possibility of partial charging and charging at the depot and no intercustomer demand, which perfectly fits our problem for the industrial use case.

Table 2-1. Table of relevant works

Papers	Year	Problem			CM				CT		CL			Fleet			VRPC				AFC			CDC				Objective	Solving Method
		EVRP	Autonomous VRP	Container Drayage	Full Charging	Partial Charging	Battery Swapping	Multiple Charging Levels	Linear	Non-Linear	Recharging Stations	Depot	Customer	Homogenous	Heterogenous fleet of Evs	Mix of conventional and Evs	Capacity Constraint	Hard Time-Windows	Soft Time-Windows	Time-Dependent	Robots	Autonomous Vehicle	Drones	Inclusion Empty Containers	Intercustomer demand	Heterogenous Fleet	Capacity > 1		
Conrad&Figliozzi	2011	•				•			•				•	•			•	•										Number of Vehicles + costs	MIP + Heuristic
Erdogan&Miller	2012	•			•				•		•			•				•										Distance	MIP + Heuristic
Schneider et al	2014	•			•				•		•			•			•	•										Distance	MIP + Metaheuristic
Meng&Ma	2020	•			•		•		•		•			•			•		•									Costs	MIP + Metaheuristic
Afroditi et al	2014	•			•				•		•			•			•	•										Number of Vehicles + Distance	MIP
Felipe et al	2014	•				•			•		•			•			•											Costs	Metaheuristic
Keskin&Catay	2016	•				•			•		•			•			•	•										Distance	MIP + Metaheuristic
Desaulniers	2016	•			•	•			•		•			•			•	•										Costs	Exact Optimization
Verma	2018	•			•		•		•		•			•			•	•										Costs	MIP + Metaheuristic
Mao et al	2020	•				•	•		•		•		•	•			•	•										Costs	MIP + Metaheuristic
Montoya et al	2015	•			•	•				•	•			•														Time	MIP + Exact Optimization
Montoya et al	2017	•			•	•				•	•			•														Time	MIP + Metaheuristic
Hiermann et al	2016	•			•				•		•				•		•											Costs	MIP + Exact Optimization + Metaheuristic
Lebeau et al	2015	•			•				•			•				•	•	•										Costs	MIP + Heuristic
Sassi et al	2014	•				•		•	•		•	•				•	•	•		•								Costs	MIP + Heuristic
Sassi et al	2015	•			•			•	•		•	•			•		•											Costs	Metaheuristic
Goeke&Schneider	2015	•			•				•		•					•	•	•										Distance + Costs	Metaheuristic
Fan et al	2023	•				•		•	•		•			•			•											Costs	MIP + Metaheuristic
Keskin&Catay	2018	•				•		•	•		•			•			•	•										Recharging Costs	MIP + Matheuristic
Shao et al	2017	•			•			•	•		•			•			•			•								Costs	MIP + Metaheuristic
Lu et al	2020	•				•			•		•			•			•	•		•								Costs	MIP + Exact Optimization + Metaheuristic
Zhang et al	2022	•				•			•		•			•			•			•								Costs	MIP + Metaheuristic
Ham & Park	2016	•				•			•		•			•			•	•		•								Number of Vehicles + Distance	MIP + Exact Optimization
Ferro et al	2018	•				•			•		•			•			•	•		•								Costs	MIP + Exact Optimiztion
Kumar et al	2023	•				•		•	•		•			•			•	•		•								Costs	MIP + Matheuristic
Sonnenberg et al	2019		•														•	•			•							Costs	MIP + Exact Optimization
Gnegel et al	2021		•														•		•		•							Costs	MIP + Exact Optimization
Ulmer & Streng	2019		•														•				•							Costs	Heuristic
Reed et al	2022		•														•					•						Delivery Time	MIP + Exact Optimization
Schonfeld & Choi	2017		•														•						•					Costs	Heuristic
Dorling et al	2017		•														•						•					Costs + Delivery Time	MIP + Metaheuristic
Cheng et al	2020	•	•		•				•					•			•						•					Costs	MIP + Exact Optimization
Wang & Regan	2002			•													•	•							•			Costs	MIP + Heuristic
Jula et al	2005			•													•	•							•			Costs	MIP + Exact Optimization + Metaheuristic
Ileri et al	2006			•													•	•						•	•	•		Costs	MIP + Exact Optimization
Xue et al	2014			•													•	•						•	•			Costs	MIP + Metaheuristic
Schulte et al	2017			•													•	•						•				Costs	MIP + Exact Optimization
R. Zhang et al	2015			•													•								•		•	Costs	MIP + Metaheuristic
Vidovic et al	2017			•													•	•							•		•	Costs	MIP + Metaheuristic
Heilig et al	2017			•													•		•						•	•		Costs	MIP + Metaheuristic
Giuliano et al	2020	•		•		•				•		•				•	•							•		•		Distance	MIP + Exact Optimization
Dessouky & Yao	2023	•		•	•					•	•					•	•							•	•	•		Costs	MIP + Exact Optimization + Metaheuristic

2.2 Solution Methods

In this section, we provide the solution methods for the closest works to our thesis. We first discuss the exact approaches, and then the metaheuristics corresponding with the closest works we discussed in the previous section.

2.2.1 Exact Methods

Exact methods are methods that could be used to solve a problem to optimality. These methods are most suitable for small problems since they often require large amounts of computation time. One of the most common exact methods for VRP variants is the use of Integer Linear Programs (ILP) which is a mathematical program with linear constraints and an objective function. These models can be solved with the use of for example the branch and bound method. This method divides the problem into sub-problems. Solving these sub-problems leads to bounds of the optimal solution. This bound helps in reducing the solution space, because if the bound is worse than the current optimal solution, the whole solution region involving that subproblem can be eliminated (Theurich et al., 2021).

Branch and Cut method is an extension of the branch and bound method, where after a relaxation of the problem is solved, for example, a problem without integer constraints, certain inequalities are added to narrow down the solution space and eliminate non-integer values.

Commercial solvers can be used to solve these problems to optimality. These solvers implement branch-and-bound methods, or branch-and-cut methods to solve problems. Two of the most common solvers are CPLEX and Gurobi. They are high-performance optimization solvers employed to solve complex optimization problems, including ILP and mixed-integer linear programming. Dessouky & Yao (2023) used Gurobi as optimization software for small instances, however, they switched to an Adaptive Large Neighborhood Search as instances got bigger. Giuliano et al. (2020) used Gurobi as well, however, they divided their optimization problem into 2, a minimum cost circulation problem to determine a set of optimal vehicle trips, which is a Linear Program and can be solved to optimality with the use of Gurobi, and a bin-packing problem to assign the vehicle trips to the fewest number of trucks as possible. On the other hand, Schulte et al. (2017) and Ferro (2018) used CPLEX as their solver after they created a mathematical program to display their problem.

2.2.2 Metaheuristics

Since VRP variants are NP-hard, the computation time of exact methods increases exponentially with the expansion of the problems. Schulte et al (2017), could not reach an optimal solution within two hours if the number of trips to be scheduled approached 50. (Meta)heuristics are methods to shorten the computation time of solving the problem. Heuristics are methods that do not lead to an optimal solution for a problem, however, they lead to good solutions within a reasonable amount of time. Examples of normal heuristics are the Nearest Neighbor heuristic in which a vehicle starts at an initial customer and then travels to the nearest customer until the capacity constraint is met or all customers are served or the Clarke & Wright Savings Algorithm, in which two routes are merged into one if this leads to a feasible solution and the most savings in terms of the objective value.

Metaheuristics are high-level heuristics which are more generic than normal heuristics. Metaheuristics are not problem-specific, meaning they are suitable for more problems instead of only routing problems. They frequently involve a search through a solution space (Abualigah et al., 2023). In this section, the metaheuristics corresponding with our closest works are described, namely the Ant Colony Optimization and the Adaptive Large Neighborhood Search.

Ant Colony Optimization

Ant Colony Optimization (ACO) is an optimization algorithm based on the probabilistic behavior of ants. The method simulates the movement of artificial ants. They probabilistically travel in the solution space while leaving pheromones on the routes. They leave more pheromone on better routes. Then routes with more pheromones are visited more often with a higher possibility by other ants, since they can “smell” the pheromones on the better route. This means that the ants both explore and exploit routes until they improve so much that they are concentrated on the best route. This makes ACO very suitable for global optimization. Since the method involves exploring, the chances of ending in a local optimum are relatively small. (Y. Wang & Han, 2021)

Kumar (2023) used the ACO method in their research including time-of-use pricing in their EVRP. In their proposed approach they map the EVs as artificial ants and let them travel over the routes with a probability based on the level of pheromone on the route, the distance of the route, the time windows of the customers, and the battery level of the EV. If they cannot do a route because of their low battery level, a trip to a charging station is inserted in the route. After all the customers are visited per ant, the ant with the lowest objective value is chosen as the optimum, then all pheromone trails are updated with the use of a certain formula, and another iteration is performed until a certain termination criterion is met.

The advantages of the ACO method are that the chances of ending up at a local optimum are very small, it does not need an initial solution, and it is very suitable for routing problems. Disadvantages are that the ACO method may converge slowly to a solution in large solution spaces and it requires a significant amount of parameter tuning to find the right balance between exploration and exploitation.

Adaptive Large Neighborhood Search

The Adaptive Large Neighborhood Search (ALNS) is a metaheuristic that uses destroy and repair methods to improve current solutions. It is an improvement heuristic, which means that it starts with an initial solution. In a classic Neighborhood Search (NS), a neighborhood is defined as the set of solutions that can be obtained by applying a small change to the current solution. In terms of a VRP, this could be swapping customers between vehicles or changing the order of when the customers should be visited by the vehicle. This neighborhood is then evaluated by calculating the objective value of the neighbor solution and the current solution is updated if a better solution is found. All steps are performed iteratively until a termination criteria is met.

In a Large Neighborhood Search, the same procedure as in a NS is followed, however, the neighborhoods of a current solution are broader. It often consists of a destruction and a construction phase. In a destruction phase, a part of the current solution is destructed, such as the complete route of one vehicle. This is the moment in which the neighborhood is explored. Then in the construction phase, the solution is constructed again using problem-specific heuristics (Le Colleter et al., 2023). The Adaptive Large Neighborhood Search is an extension of the Large Neighborhood Search, in which the destruction phase is monitored, to see which neighborhood selection strategies lead to the best improvements of the current solution. The approach during the destruction phase is then dynamically adjusted, by intensifying the destruction strategies that lead to better objective values and neglecting destruction strategies that do not lead to better objective values (Windras Mara et al., 2022).

The ALNS method is used in the research of Dessouky & Yao, who combined the EVRP with the Container Drayage Problem. They started with the creation of an initial solution by using the nearest-neighbor approach. They started by scheduling 1 truck with their nearest neighbor until their working time limit was met, and then they scheduled the next until all trips were scheduled.

In this initial solution approach, they considered the trucks to be fuel-based. In their ALNS method, they made the transition to the use of Electric Trucks. They started each iteration by substituting truck types and inserting charging trips with the use of a greedy insertion algorithm for each substitution from fuel to electricity. Then, trucks that exceed their working time, due to the new inclusion of their charging time, lose some of their tasks. They also randomly removed tasks from all the trucks with a certain probability. This completes the destruction face. In the construction face, they assigned the tasks back into trucks or even employed additional trucks to ensure demand satisfaction, and then they optimized the routes for each truck with the use of a commercial solver. This method could solve a problem with 12 customers and 2 charging stations within 10 seconds, and it also solved scenarios with more than 300 customers to optimality with a computation time of two hours.

Besides their low computation time, other advantages of ALNS that we find are that many neighborhoods can be explored in one method and that problem-specific heuristics can be included in the algorithm. Disadvantages are that the method may be too dependent on the initial solution and that the algorithm might end up in a local optimum if the neighborhood structure is not chosen properly.

2.3 Conclusion

In this literature review section, we reviewed concepts and problems in the area of EVRP and Container Drayage Problems. We first analyzed the classic VRP, together with features that are relevant to both our use cases. Then information is provided on the classic EVRP, with different charging types, such as full charging, partial charging, battery swapping, and non-linear charging. We also discussed the inclusion of a mixed fleet, with the option of incorporating only EVs or a combination of EVs and conventional vehicles. Then a review of the use of solar energy and time-dependent EVRP was provided. After that we discussed autonomous vehicle routing problems and Container Drayage Problems and the last problem we reviewed was the combination of Container Drayage Problems and EVRP. Then, we combined all reviewed papers in a table with an overview of all features and we selected the closest papers to our research. After that, we described the solution method of the closest papers in general and for the specific problems of the paper.

To conclude about the problems and concepts in our research field, we can say that there has been an extensive amount of research done on EVRP with different features. The most relevant features for our research are the different charging types since we can experiment with these charging types in our solution design phase and the experiments phase, and the inclusion of solar energy and time-dependency, since that is similar to our problem at both use cases.

The same holds for Container Drayage Problems. Much research has been done on that topic, however, not much research has been done on the combination of Container Drayage Problems and EVRP. The Container Drayage Problem is suitable to model the industrial use case since that use case involves container transport at an inland terminal, however, it is also suitable to model the urban use case. The urban use case involves the transport of packages by drones or street robots. Concept-wise, a problem in which a truck transports one container at a time, or a problem in which drones transport one package at a time, can be modeled the same.

Much research has been done on autonomous vehicle routing problems as well. Most of these researches include combinations with manual vehicles which is not relevant to our problems since they consist of autonomous vehicles only. However, it is important to note that a routing problem for autonomous vehicles does not differ from routing problems for manual vehicles, since the concept of the routing problem does not change, only the parameters change, which means we can use the concept of a manual routing problem for our research as well.

Concluding about possible solutions, we can say that, by looking at the closest works to our research, small instances of problems are solved with an exact method with the help of a commercial solver. This holds for problems combining EVRP with container drayage problems, problems regarding autonomous routing, and problems where time dependency is included in the EVRP. Since the problem in the industrial use case (the XL business park) is relatively small, we try to use an exact method as well to come to the most optimal solution for those small instances. Because the problem at the XL business park will be solved on a daily basis, an exact method might still be suitable despite its long computation time. After all, the solver is allowed to run the whole night to come to an optimal solution, since the input of the problem will not change overnight. However, when the problem instances get bigger in the urban use case, we switch to a metaheuristic. The metaheuristics that were executed in our closest researches were the ACO method and the ALNS method. The metaheuristic that we chose is the ALNS method because of the reason that it is used in a container drayage problem, which is closer to our research and also fits the urban use case. In this method, we can also include different construction heuristics to experiment with. It also has a very low computation time and does converge quickly to a good solution.

The answer to the main research question of this chapter is that many research has been done on concepts close to our research, such as container drayage problems and EVRP with time dependency, however, there has been little research done on the specific problem of both our use cases, which includes both container drayage problem and time-dependency in an EVRP. However, we find that there are exact optimization methods available for small instances with the help of a commercial solver and that we can use the ALNS method to come to good solutions in larger instances.

3 Solution Approach

In this chapter, we answer the following research question and subquestions:

2. How should the solution approach be designed?

a. What are the requirements necessary for designing the solution approach?
b. Which assumptions have to be made to design the solution approach?
c. Which solution methods are most suitable for solving the EVRP in both use cases?

In Section 3.1, we provide the problem formulation for both use cases. In Section 3.2 we provide a mathematical model for our problem formulation and in Section 3.3 we explain our solution methodology.

3.1 Use cases and Problem formulation

In this section, we provide information about both our use cases in sections 3.1.1 and 3.1.2. We use this information to develop a conceptual graph in Section 3.1.3 Then in Section 3.1.4 we provide our problem formulation, while in Section 3.1.5 we discuss the assumptions for our solution approach.

3.1.1 Industrial Area/SAVED

The first use case considered in this thesis is the industrial area use case. The industrial area covers a semi-private environment at the XL business park in Almelo. It consists of a depot and multiple warehouses. In the use case, one AGV delivers containers from the depot to the warehouses or from the warehouses to the depot. There is no container transportation between warehouses, and there might be more than 1 container transport order to one warehouse. Figure 3.1 shows an image of the map of the XL business park. The CTT location is marked green and the three customers are marked red. The black arrows show the route from CTT to Timberland, while the black dotted arrows show the route from CTT to Bleckmann. Since Bolk is very close to CTT, the route from CTT to Bolk is not shown.

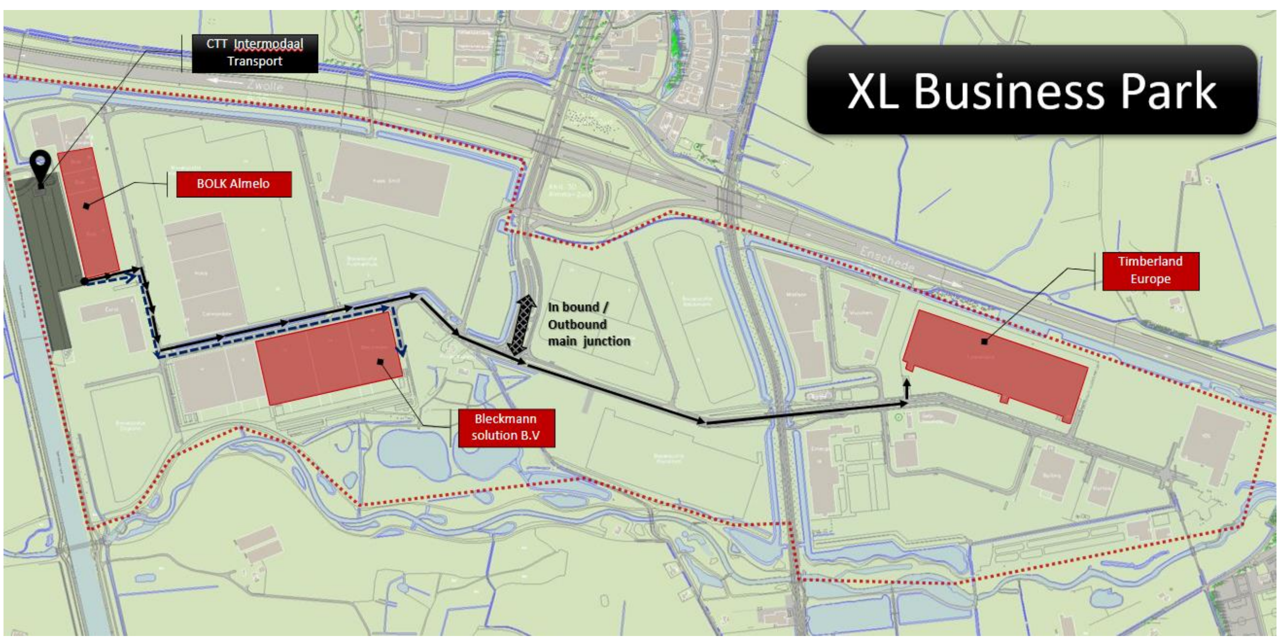

Figure 3.1. Map of the XL business park with relevant locations and routes

3.1.2 Urban Area/Campus

The urban area use case covers a use case at the campus of the University of Twente. Figure 3.2 shows a map of the University of Twente. The use case consists of a depot and multiple customer locations all over the campus. The depot is located close to the numbers 5, 6, 7 and 8 in the figure. The customer locations could be student houses or office buildings. A set of drones and street robots deliver packages from the depot to one of the customers. It can only transport one package at a time. This means that we make use of the same conceptual graph as in Section 3.1.1, in which the nodes represent the trips and the edges represent the distance between these trips.

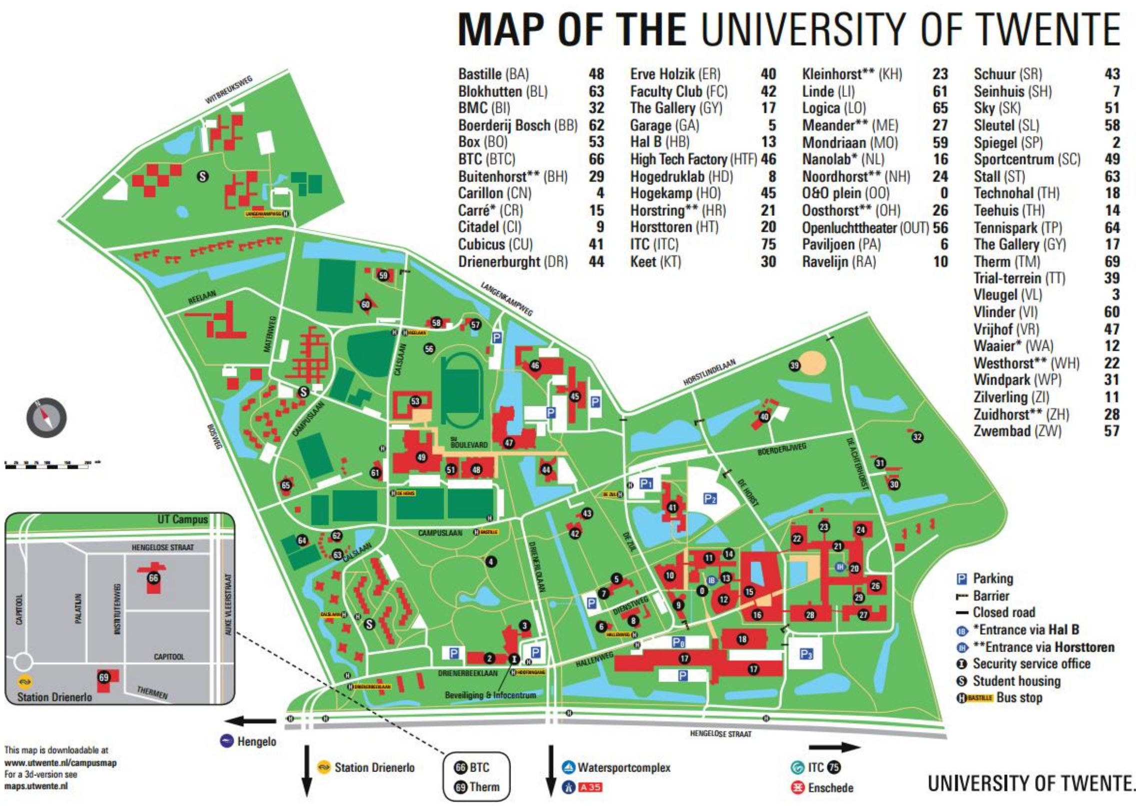

Figure 3.2. Map of the University of Twente

3.1.3 Conceptual Graph

The similarity between both use cases is that they are both EVRP, sharing the same problem graph. In both use cases, all vehicles can only transport one container or package at a time, which changes the routing graph. A standard routing problem consists of a graph with nodes and edges $(G = (V, E))$. In that case, the nodes are the customers and the edges represent the distances between the customers.

In our case, however, the nodes are the container transport orders and the edges represent the distances between the end location of one trip and the start location of the other trip. The reason for this is that the vehicles can only transport one package at a time. If for example, two packages A and B need to be transported from the depot to customers X and Y, the vehicle has to go back to the depot in between. The goal is to still model the route as Depot-A-B-Depot. This cannot be done using a standard routing graph, because then the distance between A and B is equal to the distance between customers X and Y. This is not correct, since the vehicle has to go back to the depot in between. So, to still model the route as Depot-A-B-Depot, the edges have to represent the distances between the end location of the first trip (X), and the start location of the second trip (Depot).

Figure 3.3 shows an example of how this influences the graph in the industrial area use case. The left graph is a conventional graph in which the nodes are locations. In this hypothetical case, we have one depot and 3 customers as nodes. The edges show the distances in kilometers between these locations. As an example, we have 3 container transportation trips. The first trip is from the depot to customer 1, the second is from customer 2 to the depot, and the third is from the depot to customer 3. This is represented by the arrows in the left graph.

In the right graph, the nodes represent the trips. Node 1 represents a trip from the depot to customer 1. Node 2 represents a trip from customer 2 to the depot and node 3 represents a trip from the depot to customer 3. Each node has a certain distance, namely the length of the trip. This is represented by the number close to the node. For node 1 this is 5 because the distance between the depot and customer is 5. It can be seen that there are two edges between each node. This is because our model now has an asymmetrical distance matrix. For example, the distance from node 1 to node 2 is 12, since trip 1 ends at customer 1 and trip 2 starts at customer 2, and from the left graph, we can see that the distance between customers 1 and 2 is 12. However, the distance from node 2 to node 1 is 0. This is because trip 2 ends at the depot and trip 1 starts at the depot.

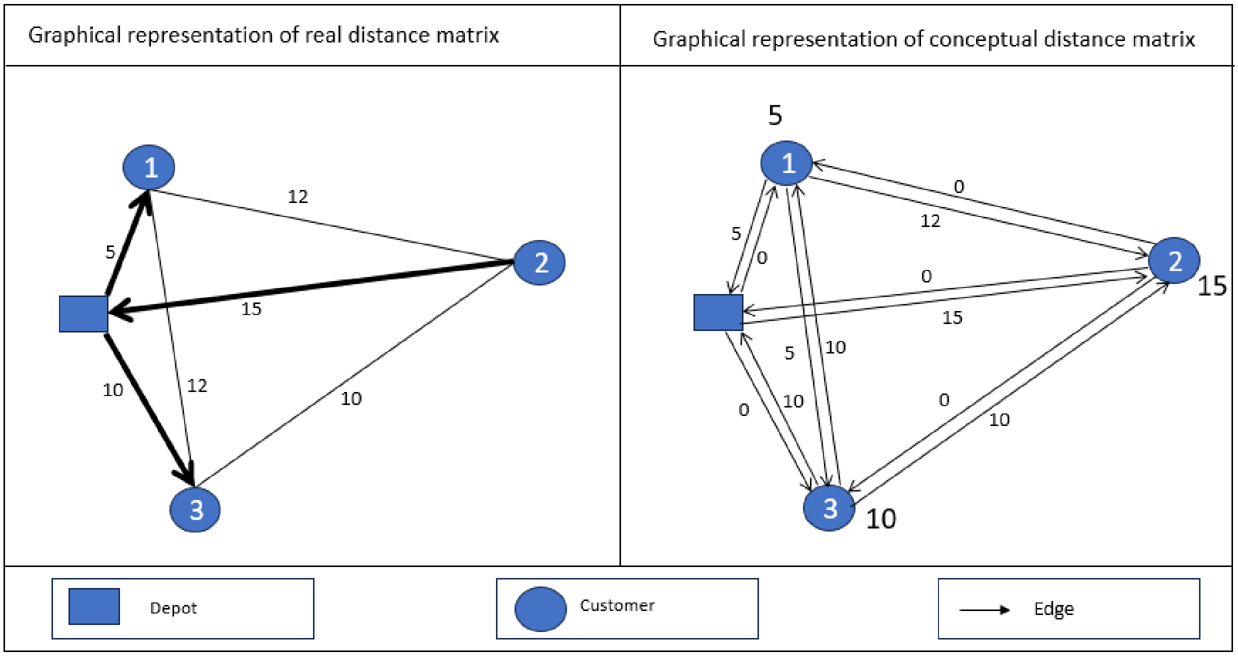

Figure 3.3. Conceptual graph to model both use cases

3.1.4 Problem Formulation

The overall problem formulation that covers both use cases, is an EVRP consisting of a graph in which the nodes are either the depot or a set of trips *(T)*. Each trip has a certain distance, (dis_i), time window $(stTW_i$ *and* $endTW_i)$, and service time (loading and unloading) (s_i). The start of the time window is a hard constraint that cannot be violated while violating the end of the time window leads to penalty costs.

Furthermore, our problem has a set of vehicles *(V)*, which have a certain speed (sp_k). Because each vehicle has a constant speed, and each trip has a certain distance, we can say that each trip has a certain duration (du_{ik}), depending on the vehicle. Furthermore, each vehicle has an energy consumption factor per minute without transporting packages (eec_k), battery capacity (bc_k), and battery level at the start of the day (bs_k) *for k in V*.

The energy consumption factor per trip depends on both the weight of the trip and the characteristics of the vehicle. To model this, each trip has a certain weight factor (wf_i). The energy consumption per minute of the vehicle is then the empty energy consumption factor multiplied by the weight factor. Each vehicle can be charged at the depot. This happens with a certain constant charging speed *(cs)*. The charging rate, however, depends on the solar power available at that moment and the number of vehicles charging at the same time. At the end of the day, the vehicles need to have a certain threshold battery level value, to be ready for the next day *(be)*.

3.1.5 Assumptions

In this section, we state the list of assumptions that are made to model our electric vehicle routing problem. These assumptions are used for the mathematical model and the solution method.

General Assumptions

- All demand is known at the start of the day, including time windows.
- The weather forecast is known at the start of the day.

Vehicle Assumptions

- All vehicles have a single-unit capacity.
- All vehicles have a constant speed.
- All vehicles have a constant energy consumption function.
- The energy consumption rate depends on the weight of the package transported.
- The distance between two locations for the vehicles is the Euclidean distance between those locations multiplied by a certain distance factor.

Charging Assumptions

- All vehicles are charged with a constant charging speed.
- The vehicles can be charged using the partial charging strategy.
- A vehicle can be charged at any moment.
- The charging rate depends on the amount of solar energy available and the number of vehicles charging.
- There is no limit on the number of vehicles charged at the same time.

3.2 Solving Methods

Table 3-1 shows a comparison of the three solving methods for the problem of Section 3.1. In the rest of the chapter, we provide 2 solving methods for the problem. In the next section, we present a mathematical model of the problem, which can be solved with an exact optimization method. while in the section thereafter we provide the ALNS heuristic for solving the problem. It shows that there are multiple objectives for the problem, namely the charging costs (CC), the traveling time (TT), and the penalty costs for missing the time window of each trip (PC). It can be seen that both the mathematical model and the heuristic can optimize all three objectives, with both hard and soft time windows, but not at the same time (orange dots). However, the mathematical model excludes solar charging, since the solar charging rate depends on the weather and the number of vehicles charging at the same time, which to the best of our knowledge cannot be captured in a mathematical model. Furthermore, it does not consider a mixed fleet. The main value in the model lies in the routing component of the problem, since that can be solved to optimality. Therefore, it can be seen as a benchmark for the routing component of the heuristic.

In contrast, the ALNS takes solar charging into account. The heuristic consists of a constructive heuristic and an improvement heuristic. The constructive heuristic only accounts for full charging, while the improvement heuristic accounts for partial charging. They both take solar charging and the use of a mixed fleet into account.

Table 3-1. Comparison of different solving methods and their characteristics

		Objectives			Time Windows		Charging Method		Solar Charging		Mixed Fleet	
		CC	TT	PC	Hard	Soft	Full	Partial	Yes	No	Yes	No
Mathematical Model		●	●	●	●	●		●		●		●
ALNS	Constructive Heuristic	●	●	●	●	●	●		●		●	
	Improvement Heuristic	●	●	●	●	●		●	●		●	

3.3 Mathematical Model

In this section, we provide a mathematical model that fits both use cases with the characteristics described in the previous section. The model is a Mixed Integer Non-Linear Problem (MINLP).

Sets

$T = Set\ of\ Trips, T\{0\}\ is\ depot$
$V = Set\ of\ Vehicles$

Parameters

$du_i =\ Duration\ of\ trip\ i$	$\forall i \in T - \{0\}$
$stTW_i = Start\ of\ time\ window\ of\ trip\ i$	$\forall i \in T - \{0\}$
$endTW_i = End\ of\ time\ window\ of\ trip\ i$	$\forall i \in T - \{0\}$
$s_i = Servicetime\ (Loading + Unloading)\ of\ trip\ i$	$\forall i \in T - \{0\}$
$t_{ij} =\ Travel\ time\ between\ trip\ i\ and\ trip\ j$	$\forall i, j \in T$
$eec_k = Energy\ consumption\ per\ unit\ of\ time\ without\ load\ of\ vehicle\ k$	$\forall k \in V$
$bc_k = Battery\ capacity\ of\ vehicle\ k$	$\forall k \in V$
$bs_k = Battery\ level\ of\ vehicle\ k\ at\ start\ of\ the\ day$	$\forall k \in V$
$wf_i = Weightfactor\ of\ trip\ i$	$\forall i \in T - \{0\}$
$cs = Charging\ speed$	
$cr = Charging\ rate$	
$cp = Penalty\ costs\ per\ time\ unit\ of\ being\ late$	
$ct = Costs\ of\ travelling\ without\ load$	
$be = Battery\ threshold\ value\ for\ the\ end\ of\ the\ day$	
$M = Sufficient\ big\ number$	

Variables

$X_{ijk} = binary\ variable\ (1\ if\ vehicle\ k\ drives\ from\ trip\ i\ to\ trip\ j, 0\ otherwise)$	$\forall i, j \in T, \forall k \in V$
$S_i = Start\ time\ of\ trip\ i$	$\forall i \in T - \{0\}$
$E_i = End\ time\ of\ trip\ i$	$\forall i \in T - \{0\}$
$Y_k = binary\ variable\ \ (1\ if\ vehicle\ is\ used, 0\ otherwise)$	$\forall k \in V$
$C_i = binary\ variable\ (1\ if\ charging\ takes\ place\ after\ trip\ i, 0\ otherwise)$	$\forall i \in T - \{0\}$
$CT_i = Charging\ time\ after\ trip\ i$	$\forall i \in T - \{0\}$
$BS_i = Battery\ level\ of\ the\ vehicle\ doing\ trip\ i\ at\ the\ start\ of\ the\ trip$	$\forall i \in T - \{0\}$
$BE_i = Battery\ level\ of\ the\ vehicle\ doing\ trip\ i\ at\ the\ end\ of\ the\ trip$	$\forall i \in T - \{0\}$
$W_{ijk} =\ Help\ Variable\ 1\ to\ linearize\ the\ multiplication\ of\ X_{ijk} * (1 - C_i)$	$\forall i, j \in T, \forall k \in V$
$Z_{ijk} = Help\ Variable\ 2\ to\ linearize\ the\ multiplication\ of\ X_{ijk} * C_i$	$\forall i, j \in T, \forall k \in V$

Objective

$$Min\ (cr * \sum_{i \in T - \{0\}} CT_i + ct * \sum_{i \in T} \sum_{j \in T} \sum_{k \in V} W_{ijk} t_{ij} + Z_{ijk}(t_{i0} + t_{j0}) + cp * \sum_{i \in T} P_i \qquad (1)$$

Constraints

$$\sum_{j\in T-\{i\}}\sum_{k\in V} X_{ijk} = 1 \qquad \forall i \in T-\{0\} \qquad (2)$$

$$\sum_{j\in T-\{i\}} X_{ijk} = \sum_{j\in T-\{i\}} X_{jik} \qquad \forall i \in T-\{0\}, \forall k \in V \qquad (3)$$

$$\sum_{j\in T} X_{0jk} = 1 \qquad \forall k \in V \qquad (4)$$

$$S_i + du_i + s_i \leq E_i \qquad \forall i \in T-\{0\} \qquad (5)$$

$$\sum_{i\in T-\{0,j\}}\sum_{k\in V}(X_{ijk}E_i + W_{ijk}t_{ij} + Z_{ijk}(t_{i0}+t_{0j}) + X_{ijk}CT_i) \leq S_j \qquad \forall i \in T-\{0\} \qquad (6)$$

$$CT_i \leq M * C_i \qquad \forall i \in T-\{0\} \qquad (7)$$

$$Y_k \geq M * \sum_{j\in T-\{0\}} X_{0jk} \qquad \forall k \in V \qquad (8)$$

$$S_i \geq stTW_i \qquad \forall i \in T-\{0\} \qquad (9)$$

$$E_i \leq endTW_i + P_i \qquad \forall i \in T-\{0\} \qquad (10)$$

$$BS_i - \sum_{j\in T-\{i\}}\sum_{k\in V} X_{ijk}\, du_i eec_k wf_i \geq BE_i \qquad \forall i \in T-\{0\} \qquad (11)$$

$$\sum_{i\in T-\{0,j\}}\sum_{k\in V}(X_{ijk}(BE_i - eec_k(C_i(t_{i0}+t_{0j}) + (1-C_i)t_{ij}) + CT_i * cs)) = \sum_{i\in T-\{0,j\}}\sum_{k\in V} X_{ijk}BS_j \qquad \forall i \in T-\{0\} \qquad (12)$$

$$BE_i - \sum_{j\in T-\{i\}}\sum_{k\in V}(X_{ijk}\, eec_k t_{i0}) \geq 0 \qquad \forall i \in T-\{0\} \qquad (13)$$

$$\sum_{k\in V}(X_{i0k}\,(BE_i - eec_k t_{i0} + CT_i * cs)) \geq \sum_{k\in V}(X_{i0k}\, be) \qquad \forall i \in T-\{0\} \qquad (14)$$

$$\sum_{j\in T-\{i\}}\sum_{k\in V} X_{ijk}(bc_k - BS_i) \geq 0 \qquad \forall i \in T-\{0\} \qquad (15)$$

$$\sum_{j\in T-\{0\}}(X_{0jk}\,(BS_j + eec_k t_{0j})) = \sum_{j\in T-\{0\}}(X_{0jk}\, bs_k) \qquad \forall k \in V \qquad (16)$$

$$W_{ijk} \leq X_{ijk} \qquad \forall i,j \in T, \forall k \in V \qquad (17)$$

$$W_{ijk} \leq 1 - C_i \qquad \forall i,j \in T, \forall k \in V \qquad (18)$$

$$W_{ijk} \geq X_{ijk} - C_i \qquad \forall i,j \in T, \forall k \in V \qquad (19)$$

$$Z_{ijk} \leq X_{ijk} \qquad \forall i,j \in T, \forall k \in V \qquad (20)$$

$$Z_{ijk} \leq C_i \qquad \forall i,j \in T, \forall k \in V \qquad (21)$$

$$Z_{ijk} \geq X_{ijk} + C_i - 1 \qquad \forall i,j \in T, \forall k \in V \qquad (22)$$

$$X_{ijk}, C_i = Binary \qquad \forall i,j \in T, \forall k \in V \qquad (23)$$

$$S_i, E_i, P_i, CT_i, BS_i, BE_i \geq 0 \qquad \forall i \in T \qquad (24)$$

The objective value is the sum of all charging costs, the costs of driving without a container/package, and the costs of using each vehicle. Constraint (2) ensures that each trip is done once. Constraint (3) ensures that the vehicle that starts with a trip is the same vehicle that ends the trip while constraint (4) ensures that each vehicle starts at the depot. Constraint (5) represents the time flow between the start of the trip and the end of the trip including service time. Constraint (6) represents the time flow between the end of a trip and the start of the next trip. If there is no charging after trip i is executed, Z_{ijk} is 0, W_{ijk} is 1 and the time between trips is t_{ij}. Else, the time between trips is the charging time and the travel time to the depot for charging. Constraint (7) ensures that if the charging time after a trip is higher than 0, the binary variable is 1. Constraint (8) ensures that if the vehicle leaves the depot, the corresponding binary variable is 1. Constraint (9) guarantees that a trip cannot be started before the start of its time window, while constraint (10) ensures that the end of the time window is met, while otherwise leading to penalty costs. Constraint (11) represents the battery flow between the start and end of a trip. Constraint (12) represents the battery flow between the end of a trip and the start of the next one, including the loss of energy by driving between the two trips and the gain of energy by charging. Constraint (13) guarantees that the battery level and the end of a trip plus the energy loss from driving to the charging station is always above 0. Constraint (14) guarantees that the battery level after the last trip including the energy loss from driving back to the depot and the charging is above a certain threshold value. Constraint (15) ensures that the battery level is always lower than the battery capacity of the vehicle while Constraint (16) ensures that the battery level at the first trip of the day is equal to the start battery level of the vehicle minus the energy loss of driving to the first trip. Constraints (17-22) linearize the multiplication of $X_{ijk} * (1 - C_i)$ and $X_{ijk} * C_i$, while Constraints (23) and (24) are the sign restrictions.

3.3.1 Validation

In this section, we validate the model. The purpose of the validation is to test whether the model indeed provides the best results for the problem without neglecting the constraints. To do this, we created a small toy problem for the SAVED use case, consisting of 5 container jobs. The jobs and its specifics are shown in Table 3-2

Table 3-2. List of Container Jobs Toy Problem

Number	Start	End	StartTW	EndTW	WF
1	CTT	Timberland	360	380	2
2	Timberland	CTT	580	600	2
3	CTT	Bleckmann	360	600	2
4	Timberland	CTT	360	400	2
5	CTT	Bolk	400	500	2

The time matrix between the 4 companies can be found in Table 3-3. Furthermore, we use one vehicle, with a battery capacity of 100 kWh and energy consumption per minute of 0.1 kWh. The vehicle starts with a full battery and has to end with a full battery. The charging speed is 500 W and the charging cost is €0.3 per kWh. In this toy problem, we only focus on the minimization of the charging costs. This means that the weights of the traveling time, and the number of vehicles are 0.

Table 3-3. Time Matrix Toy Problem

	CTT	Bolk	Bleckmann	Timberland
CTT	0	1	4	10
Bolk	1	0	3	9
Bleckmann	4	3	0	7
Timberland	10	9	7	0

Because of the difference in time windows between the 5 jobs, the best route is to start with job 1, then do job 4 immediately after since the vehicle is already at Timberland. Then it is best to do job 5 and drive back empty to CTT, then job 3, continue empty to Timberland to end with job 2. In total, this would lead to driving with a container for 35 minutes and without a container for 8 minutes. This leads to a total energy consumption of 7.8 kWh, which can be charged for €2.56. Table 3-4 shows that the model reaches the same order of jobs. It shows that the first job is 1 because the cell of row 0 and column 1 displays 1. Then it does 4, 5, 3, and 2, as indicated by the time window constraint. The corresponding objective value is indeed 2.56 Therefore it can be concluded that the mathematical model is valid for this small toy problem. In Chapter 4, we experiment with bigger instances to investigate how the model compares to our heuristic.

Table 3-4. Optimal Solution Toy Problem

	0	**1**	**2**	**3**	**4**	**5**
0	0	1	0	0	0	0
1	0	0	0	0	1	0
2	1	0	0	0	0	0
3	0	0	1	0	0	0
4	0	0	0	0	0	1
5	0	0	0	1	0	0

3.4 Heuristic

In this section, we explain the heuristic for solving the problem. We provide an explanation of the constructive heuristic in Section 3.4.1, then we explain the improvement heuristic in Section 3.4.2.

3.4.1 Constructive heuristic

The constructive heuristic is the starting point of the ALNS heuristic. Figure 3.4 shows the pseudocode of the constructive heuristic. The input for the algorithm is the set of trips that have to be executed and the set of available vehicles (1). The output is the *ConstructiveSolution* which consists of a set of vehicles each having a list of trips that they have to execute. Each trip has a start moment and an end moment. Furthermore, the solution consists of a list of charging moments, in which a vehicle is charged to a chosen percentage of the battery capacity. In the constructive heuristic, the chosen percentage is always 100%.

The algorithm starts with initializing the vehicles by starting them at the depot with battery level *bs* (3). Then the trips are sorted by the end of their time window in increasing order and then the duration in increasing order (4).

Then, we loop over all trips (5) and set a boolean *EnoughBattery* to false (6). We then find the vehicle that is available at the earliest moment (8-9). We then calculate the energy needed to reach the starting point of the trip, execute the trip, and go back to the depot (10). If the vehicle does not have enough battery left to meet the energy needed (11), the vehicle goes to the depot and charges to 100% (12-13). In this case, the boolean *EnoughBattery* remains false and we go back to line 8.

If this is not the case the trip is appended to the vehicle and the battery level and availability moment of the vehicle are updated (16-17). The pseudocodes for charging the vehicle and appending a trip to the vehicle are described later in Sections 3.4.3. This process is iterated until all trips are appended to a vehicle (20). If this is done, all vehicles are summoned back to the depot and charged if necessary to meet the threshold battery level for the end of the day (*be*) (21). Then the solution can be created. After that we check the feasibility of the solution, regarding battery levels and time windows and the statistics and objectives can be calculated.

Algorithm 1: Constructive Heuristic

```
Input: Trips, Vehicles
Output: ConstructiveSolution
Initialize Vehicles
Sort Trips on end of TW increasingly and duration increasingly
for Trip in Trips
    EnoughBattery ← false
    while EnoughBattery = false
        Sort Vehicles on Availabilitymoment increasingly
        ChosenVehicle ← Vehicles[0]
        Calculate EnergyNeeded for Trip
        if EnergyNeeded > Vehicle.BatteryLevel
            Charge ChosenVehicle to 100%
            Update ChosenVehicle.BatteryLevel, ChosenVehicle.AvailabilityMoment
        else
            EnoughBattery ← true
            Append Trip to Vehicle
            Update ChosenVehicle.BatteryLevel, ChosenVehicle.AvailabilityMoment
        end if
    loop
end for
Vehicles back to depot and charge until threshold
Create ConstructiveSolution of Vehicles and calculate Statistics
return ConstructiveSolution
```

Figure 3.4. Algorithm 1: ConstructiveHeuristic

3.4.2 Calculation of Objectives

The charging costs can be calculated by looping over all minutes in a day, checking per minute how many vehicles are charged, and checking the available solar energy. We assume that the available solar energy per minute is known at the start of the day. We can then calculate the energy needed from the grid to charge the vehicles that minute. This amount is then multiplied by the charging rate, and then we know the charging costs per minute. If we sum over all minutes, we know the total charging costs.
We can calculate the total traveling costs per vehicle by looping over all trips per vehicle and determining the traveling time between the end location of one trip and the start location of the next trip. If we sum over all vehicles, we know the total traveling costs. The penalty costs can be calculated by summing over all trips and calculating the difference between the end moment of each trip and the end of the time window of that trip and multiplying that by the costs of missing the time window per minute. The end of the day can be calculated by finding the trip with the latest end moment and adding the traveling time between the end location of that trip and the depot. The waiting time can be calculated by summing over all vehicles, calculating the difference between their end moment and start moment, and subtracting the number of minutes they are executing a trip, traveling between trips, and charging. Then we can calculate the total objective of a solution by adding the objectives multiplied by their weights.

3.4.3 Adaptive Large Neighborhood Search

Overall Heuristic

Figure 3.5 shows the pseudocode of the overall heuristic. The input for the heuristic is the Constructive Solution, the parameters, and the maximum number of worse solutions allowed. The output is then the overall solution of the heuristic. It starts with setting *CurrentSolution* equal to the *ConstructiveSolution* (3) and setting the *NRWorseSolutions* to 0 (4).

Then, while *NRWorseSolutions* is lower than the maximum number allowed (5), the *CurrentSolution* undergoes an improvement iteration, which returns *NewSolution* (6). The pseudocode for the improvement iteration can be found in Section 0. Then we compare the objective of *NewSolution* with the objective of *CurrentSolution.* If *NewSolution* is better than the *CurrentSolution* (7), the *NewSolution* becomes the *CurrentSolution* (8), the ALNS parameters are updated with the *UpdateParam* algorithm (9) (the pseudocode of the algorithm can be found in Section 0), and the *NRWorseSolutions* is set to 0 (10). Else (11), the *NRWorseSolutions* is incremented by 1 (12). The heuristic ends when a certain number of *NRWorseSolutions* is reached. This means that after a certain number of consecutive iterations without finding a better solution, the algorithm stops. This ensures that the algorithm does not run too long when the chance of finding improvements is low while keeping the algorithm running when the potential of finding improvements is higher.

Algorithm 2: Adaptive Large Neighborhood Search

```
1:   Input: ConstructiveSolution, MaxNRWorseSolutions, param
2:   Output: BestSolution
3:   CurrentSolution ← ConstructiveSolution
4:   NRWorseSolutions ← 0
5:   while NRWorseSolutions < MaxNRWorseSolutions
6:       NewSolution ← ImprovementIteration(CurrentSolution, param)
7:       if NewSolution.Objective < CurrentSolution.Objective
8:           CurrentSolution ← NewSolution
9:           UpdateParam(CurrentSolution, param)
10:          NRWorseSolutions ← 0
11:      else
12:          NRWorseSolutions ← NRWorseSolutions + 1
13:      end if
14:  end while
15:  BestSolution ← CurrentSolution
16:  return BestSolution
```

Figure 3.5. Algorithm 2: Adaptive Large Neighborhood Search

Improvement Iteration

Figure 3.6 shows the pseudocode for the improvement iteration. The input for the improvement iteration is the *CurrentSolutio*n and the parameters, while the output is the *Iterationsolution.* The first step of the iteration is to destroy a random subset of the Vehicles of the *CurrentSolution*. (3) In this case, destroying means that all trips of the *DestroyedVehicles* are unscheduled. This means that they do not have a start time anymore, however, they are still connected to the vehicle which executes the trip in the *CurrentSolution.* We also create a set *IntactVehicles* (4) and an empty list of Solutions (5). Because there is randomization involved in creating a new solution, we create multiple solutions (6), while we later pick the solution with the best objective value (22).

To start the reparation of the *DestroyedVehicles*, we first loop over all destroyed vehicles and their trips (7-8) and move the trips to another destroyed vehicle with a certain chance (*MovePercentage*) (9-10). The *MovePercentage* is one of the percentages that are adapted in the algorithm *UpdateParam* (Section 0). After that, we shuffle the destroyed vehicles (14) and repair each vehicle independently by using the *RepairOneVehicle* algorithm (16). The pseudocode of this algorithm can be found in Section 0. Then we append the repaired vehicle to the list *IntactVehicles* (17), and after each vehicle is repaired, we create a solution consisting of the *IntactVehicles* and we calculate the statistics and the objective of the solution (18). We also check whether the solution is feasible, and if so we append the solution to the list *Solutions* (19). After we have the complete list of *Solutions* for this iteration, we pick the solution with the lowest objective value and return it as the *IterationSolution.*

Algorithm 3: ImprovementIteration

```
1:   Input: CurrentSolution, param
2:   Output: IterationSolution
3:   DestroyedVehicles ← random subset of CurrentSolution.Vehicles
4:   IntactVehicles ← CurrentSolution.Vehicles – (DestroyedVehicles)
5:   Solutions ← ()
6:   for Solution in range (param.NrOfSolutions)
7:       for Vehicle in DestroyedVehicles
8:           for Trip in Vehicle.TripList
9:               if RandomNumber < param.MovePercentage
10:                  Move Trip to another random Vehicle in DestroyedVehicles
11:              end if
12:          next Trip
13:      next Vehicle
14:      Shuffle DestroyedVehicles
15:      for Vehicle in DestroyedVehicles
16:          Vehicle ← RepairOneVehicle(Vehicle, param)
17:          Append Vehicle to IntactVehicles
18:      next Vehicle
19:      Create Solution of IntactVehicles and calculate Statistics
20:      Append Solution to Solutions if feasible
21:  next Solution
22:  IterationSolution ← min(Solution.ObjectiveValue for Solution in Solutions)
23:  return IterationSolution
```

Figure 3.6. Algorithm 3: ImprovementIteration

Repair One Vehicle Algorithm

Figure 3.7 shows the pseudocode of the *RepairOneVehicle* algorithm. It has as input the Vehicle and the parameters, and the output is the vehicle but then repaired. It starts with initializing the *Vehicle* (3). This means that the location of the *Vehicle* is set to the depot, the battery level is set to the level at the start of the day, and the *TripList* is disconnected from the *Vehicle*. We also give the vehicle an empty list of *ChargingMoments* (4). Then we sort the *TripList* on the end of the time windows increasingly and then the *duration* increasingly (5). We then loop over the *TripList* and switch some trips with their successors with a random *SwitchPercentage* (6-10). The *SwitchPercentage* is one of the percentages that are adapted in the algorithm *UpdateParam* (Section 0). After that, we loop over all trips in the *TripList* (11). Per trip, we calculate the *EnergyNeeded* to go from the current location of the *Vehicle* to the starting location of the trip, then execute the trip, and then return to the depot (12). If the battery level of the vehicle is lower than *EnergyNeeded* (13), then the vehicle automatically has to charge (14). For this, we created the algorithm *ChargeVehicle* (Section 0). The algorithm returns the charged *Vehicle* and the *ChargingMoment,* which is then appended to the *Vehicle.* (15). If the *Vehicle* has enough battery to execute the trip, it still has to charge with a certain *RandomChargingPercentage* (16). The *RandomChargingPercentage* is also one of the percentages that are adapted in the algorithm *UpdateParam* (Section 0). After charging, or when it does not have to charge, it executes the trip with the algorithm *AppendTrip (20).* That algorithm is explained in Section 0. After all trips are appended, the vehicle goes back to the depot and charges until the threshold for the end of the day is met. At the end of the algorithm, the vehicle is returned as output.

Algorithm 4: RepairOneVehicle

```
1:  Input: Vehicle, param
2:  Output: Vehicle
3:  Initialize Vehicle
4:  Vehicle.ChargingMoments ← ()
5:  Sort TripList on end of TW increasingly and duration increasingly
6:  for Trip in range(length(TripList)-1)
7:      if RandomNumber < param.SwitchPercentage
8:          Switch TripList(Trip) with TripList(Trip+1)
9:      end if
10: next Trip
11: for Trip in TripList:
12:     Calculate EnergyNeeded for Trip
13:     if Vehicle.BatteryLevel < EnergyNeeded
14:         Vehicle, ChargingMoment ← ChargeVehicle(Vehicle, param, Trip)
15:         Append ChargingMoment to Vehicle.ChargingMoments
16:     elif RandomNumber < param.RandomChargingPercentage
17:         Vehicle, ChargingMoment ← ChargeVehicle(Vehicle, param, Trip)
18:         Append ChargingMoment to Vehicle.ChargingMoments
19:     end if
20:     Vehicle ← AppendTrip(Vehicle, param, Trip)
21: next Trip
22: Vehicle back to depot and charge until end of day threshold
23: return Vehicle
```

Figure 3.7. Algorithm 4: RepairOneVehicle

Charge Vehicle

Figure 3.8 shows the pseudocode of the *ChargeVehicle* algorithm. The input for the algorithm is the *Vehicle,* the parameters, and the trip that has to be executed after charging. The output is the *Vehicle* and the *ChargingMoment*. The algorithm starts with updating the *Availabilitytime* of the *Vehicle* by adding the time necessary to travel back to the depot (3). Also, the *BatteryLevel* is updated by subtracting the energy consumed to travel back to the depot (4). The *CurrentLocation* of the *Vehicle* is set to the depot (5). After that, we calculate the *MinimumThreshold* which is needed to execute the trip and come back to the depot. We then pick a random *Threshold* above the minimum with the *ThresholdDictionary* (7). The *ThresholdDictionary* is a dictionary that as keys has partial charging percentages to which the vehicle is charged, and as values has the chance percentages that the particular threshold is chosen. In case the vehicle is charged during the GreedySolution, the threshold is always 100%. The dictionary is updated by the *UpdateParam* algorithm (Section 0). We also pick a random time that the *Vehicle* has to wait before it charges with the *WaitingTimeDictionary* which as keys has minutes that the *Vehicle* has to wait and as values has the corresponding percentages that the particular waiting time is chosen (8). Then the time needed for charging is calculated (9) and a *ChargingMoment* is created. Then the start and end times of the charging moment are created (11-12) and the battery level and availability time of the *Vehicle* are updated (13-14). Then, the algorithm ends with returning the *Vehicle* and the *ChargingMoment*.

```
Algorithm 5: ChargeVehicle
Input: Vehicle, param, Trip
Output: Vehicle, ChargingMoment
Update Vehicle.AvailabilityTime
Update Vehicle.BatteryLevel
Vehicle.CurrentLocation ← Depot
MinimumThreshold ← BatteryLevelPercentage needed for executing Trip
Threshold ← random threshold above MinimumThreshold using
      param.ThresholdDictionary
WaitingTime ← random WaitingTime using param.WaitingTimeDictionary
ChargingTime ← (Threshold*Vehicle.BatteryCapacity)
      -BatteryLevel)/param.ChargingSpeed
Create ChargingMoment
ChargingMoment.StartTime ← Vehicle.AvailabilityTime + WaitingTime
ChargingMoment.EndTime ← ChargingMoment.StartTime + ChargingTime
Vehicle.BatteryLevel ← Threshold*Vehicle.BatteryCapacity
Vehicle.AvailabilityTime ← ChargingMoment.EndTime
return Vehicle, ChargingMoment
```
Figure 3.8. Algorithm 5: ChargeVehicle

Append Trip

Figure 3.9 shows the pseudocode of the algorithm *AppendTrip*. The input for the algorithm is the *Vehicle*, the parameters, and the *Trip*, while the output is the *Vehicle,* to which the *Trip* is appended. The first step of the algorithm is updating the *AvailabilityTime* of the *Vehicle*, by adding the travel time to the start location of the *Trip*. Then the *BatteryLevel* of the *Vehicle* is updated by subtracting the energy consumed to travel to the start location. Then the *StartTime* of the Trip is determined by taking the maximum of the *AvailabilityTime* of the *Vehicle*, and the start of the time window of the *Trip.* Then the *EndTime* of the Trip is calculated, and the *AvailabilityTime, BatteryLevel,* and *CurrentLocation* of the *Vehicle* are updated. Lastly, the *Trip* is appended to the *TripList* of the *Vehicle* and the *Vehicle* is returned

```
Algorithm 6: AppendTrip
Input: Vehicle, param, Trip,
Output: Vehicle
Update Vehicle.AvailabilityTime
Update Vehicle.BatteryLevel
Trip.StartTime ← max(Trip.StartTimeWindow, Vehicle.AvailabilityTime)
Trip.EndTime ← Trip.StartTime +
      param.TimeMatrix[Trip.StartLocation][Trip.EndLocation] + param.ServiceTime
Update Vehicle.BatteryLevel
Vehicle.AvailabilityTime ← Trip.EndTime
Vehicle.CurrentLocation ← Trip.EndLocation
Append Trip to Vehicle.TripList
return Vehicle
```
Figure 3.9. Algorithm 6: AppendTrip

Update Parameters

Figure 3.10 shows the pseudocode of the UpdateParam algorithm. This algorithm takes place at the end of one improvement iteration if a better solution is found. In this algorithm, we have 5 parameters for creating the solution. These parameters are either percentages or a dictionary with percentages. After a new solution is found, the statistics of the solutions are calculated. For the *Solution*.*MovePercentage* for example, the algorithm calculates what percentage of the trips are actually moved, while the *param.MovePercentage* is the predefined chance of moving each trip. Since the *Solution* is an improvement, we can update the parameters with the actual percentages (4-8). *AdaptivityPercentage* indicates the proportion of the new values derived from *Solution* that should influence the updated parameters. A higher *AdaptivityPercentage* means that the new solution's statistics have a greater impact on the updated parameters, while a lower *AdaptivityPercentage* means that the existing parameters retain more influence.

Algorithm 7: UpdateParam

```
Input: Solution, param
Output: param
Calculate Solution.ThresholdDictionary, Solution.WaitingTimeDictionary,
    Solution.MovePercentage, Solution.SwitchPercentage,
        Solution.RandomChargingPercentage
for ALNSParam in param
    param.ALNSParam ← Solution.ALNSParam*
        param.AdaptivityPercentage + (1-param.AdaptivityPercentage)
            param.ALNSParam
next ALNSParam
return param
```

Figure 3.10. Algorithm 7: UpdateParam

3.5 Conclusion

In this chapter, we first described our use cases and then created a problem formulation that is suitable for both use case problems. We model our EVRP, with the use of a conceptual graph in which the nodes are the trips and the edges represent the distances between the end location of one trip and the start location of the other trip. Then we explained the exact problem formulation for both use cases, including partial charging and solar charging. After the problem formulation, we stated a list of assumptions necessary for our model, such as the assumptions that all information including the weather forecast is known upfront and that all vehicles can be charged at the same time.

Then the solution methods are provided with their characteristics. A mathematical model is provided to model the problem as an EVRP, however without the use of solar charging. This mathematical model is validated with the use of a small toy problem. To include solar charging, a heuristic is developed including a constructive heuristic and an ALNS algorithm. The constructive heuristic only charges a vehicle when it is necessary to charge. It then charges automatically to 100% of the battery capacity. The ALNS heuristic then destroys parts of the solutions and rebuilds them by including moves between vehicles, switches in the order within a vehicle, random charging when not necessary, partial charging, and waiting time before charging.

4 Evaluation

In this chapter, we evaluate our solution approach by experimenting with our approach on different data instances representing both the industrial use case from SAVED and the urban use case based on the Campus of the University of Twente. We start by explaining our experimental design in Section 4.1, then we explain the data instances for both use cases in Section 4.2. We tune our parameters in Section 4.3 and evaluate the results of our experiments in Section 4.4.

4.1 Experimental Design

4.1.1 Experiments

The experimental design consists of 7 different experiments. Table 4-1 shows the experiments, together with a small explanation of the experiments. Each experiment is tested at either one of the use cases or both using the data instances which are explained in Section 4.2.

In Experiment 1 we test our ALNS heuristic against the mathematical model, both described in Chapter 3. In Experiment 2, we test the collaboration scenario against the non collaboration scenario. In Experiments 3 and 4, we test the outcome of our ALNS against a scenario in which the planning does not take solar charging into account. In that scenario, we solve the problem with the ALNS as if we have 0 panels, and save the policy part of the solution of that scenario. We then calculate the corresponding KPIs of that policy implemented in the scenario with solar panels. In Experiment 5, we test a scenario without battery usage against a scenario with battery usage and in Experiment 6, we consider soft time windows and test different weights of the objectives regarding penalty costs and charging costs.

Furthermore, each experiment except the first is executed for each of the four seasons, since the efficiency of solar panels depends on the season as can be seen in Figure 2.4. For season 1 (winter) we set the efficiency to 30%, for season 2 (spring) we set the efficiency to 70%, for season 3 (summer) we set the efficiency to 80%, and for season 4 (autumn) we set the efficiency to 60%. Also, each experiment is executed 5 times to minimize the variance between experiments and reduce the randomization of both the trip instances and the ALNS parameters.

Table 4-1. Explanation of the Experiments

ID	Name	Use case	Explanation
0	Parameter Tuning	Both	This experiment aims to identify the optimal settings for the ALNS to improve solving efficiency by using a subset of data instances. The parameters of the ALNS that are determined are the number of worse solutions allowed, the number of solutions created per iteration, the initial move percentage, the initial switch percentage, the initial random-charging percentage, the initial threshold dictionary, the initial waiting time dictionary, and the adaptivity percentage of all those parameters.
1	Exact vs Heuristic	SAVED	This experiment tests the ALNS algorithm against the exact optimization using the mathematical model. This experiment is executed without the use of solar panels since solar charging is not involved in the mathematical model.
2	Collaboration	SAVED	This experiment aims to evaluate the advantages of companies collaborating. In the non-collaborating scenario, import trips from the depot to one company cannot be combined with export trips from another company to the depot, while in the collaborating scenario, those two may be combined.
3	Variable Time Windows	Both	This experiment tests the influence of different time window distributions on the algorithm's solution and the solution's KPIs. We test a uniform distribution of time windows of the orders divided over the day against a distribution with most windows at the start of the day, middle of the day, and end of the day.
4	Variable Weather	Both	This experiment tests the influence of variable weather settings on the solution of the algorithm and its KPIs. We test an efficiency that is constant all over the day versus a situation in which the efficiency is higher in the morning, versus a situation in which it is higher around noon, versus a situation in which it is higher in the afternoon, versus a situation in which it is very low around noon and high in both the morning and afternoon.
5	Battery Usage	Campus	This experiment tests the influence of using a battery for charging. In this case, all solar energy that is not used for charging immediately is saved in a battery and can be used at the end of the day for charging the vehicles up to 100%.
6	Soft Time Windows	Both	This experiment tests the influence of using soft time windows on the solution performance. The idea of the experiment is to test with multiple weights for both the charging costs and the penalty costs, to see how the solution changes with these different weights.

4.1.2 Key Performance Indicators

The main Key Performance Indicator (KPI) of our experiments is cost. The main components of costs are charging costs and traveling costs. The charging costs consist of two different types of costs, namely the costs of using energy from the grid to charge the vehicles and the feed-in costs of delivering back energy from the solar panels to the grid. In this research, we focus on the average solar output during the day. Based on the works of Shazly, (1996), and Figure 2.3, we assume the distribution of solar output over the day to be a normal distribution with a mean of 750 minutes (12.30 pm) and a standard deviation of 150 minutes. Furthermore, we consider the traveling time in hours as a KPI to minimize. This combination covers both the routing aspect as well as the charging aspect for both use cases.

Except for Experiment 6, the charging costs and traveling time are the only objectives in the algorithm. Correspondingly to Woldring, (2024), we consider the costs for using energy from the grid to be €0.3 per kWh and the feed-in costs to be €0.1 per kWh. Furthermore, we have extra KPIs per experiment. For experiment 2 a KPI is the moment that all trips are executed and the vehicles are charged, so the end of the day. This is a relevant KPI because it shows what the advantage is time-wise of collaboration between companies at the XL business park. For the other experiments, relevant KPIs are the percentage of energy from solar panels as a fraction of the total energy needed for a day. The relevance of this KPI is that we can say how self-sustaining either the XL business park or the Campus is using the ALNS algorithm.

For Experiment 4 a relevant KPI is the percentage of energy from solar panels as a fraction of the total available solar energy. This last KPI is important because if for example, all solar energy is used, and the charging costs are still relatively high, this does not mean that the algorithm does not reach a good solution. It means that there are not enough solar panels, or the weather is not good enough, to charge the vehicles with the help of solar energy. Lastly, for experiment 6, a relevant KPI is the penalty costs for missing the time windows. Using that KPI combined with the charging costs, we can perform a sensitivity analysis on how the solution changes for different relative objective weights for both the charging costs and the penalty costs.

4.2 Data Instances

In this section, we explain the data instances that we use to execute the experiments. In Section 4.2.1 we provide details on the data instances of the industrial SAVED use case and in Section 4.2.2 the data instances of the urban Campus use case are explained.

4.2.1 SAVED Use Case

For this research, CTT made a table available with all shipped containers in the year 2022. This table consists of rows with their ID, Container type, Start location, End location, whether it was an import or export container, the time when it should be at the client (only relevant for import containers), departure moment from CTT in case of import container, Arrival moment back at CTT in case of an export container. An analysis of this data leads to a distribution of the number of containers (shown in Figure 4.1).

It can be seen that there are for example 29 containers transported in approximately 3.5% of days and 11 containers transported on approximately 2% of days. In total, there were 256 days in which containers were transported. The average number of containers transported per day is 28.4, however, there is a significant amount of variance in this distribution, since there is a significant number of days in which only 10 containers are transported, and the same holds for days with 50 containers transported. Important to note is that import and export trips are counted separately. Each container has two trips over the XL business park, one from CTT to a warehouse and one back.

Approximately half of all containers transported are either from CTT to Timberland or from Timberland to CTT. Bleckmann accounts for approximately 40% of container trips, while Bolk only has 10% of container trips.

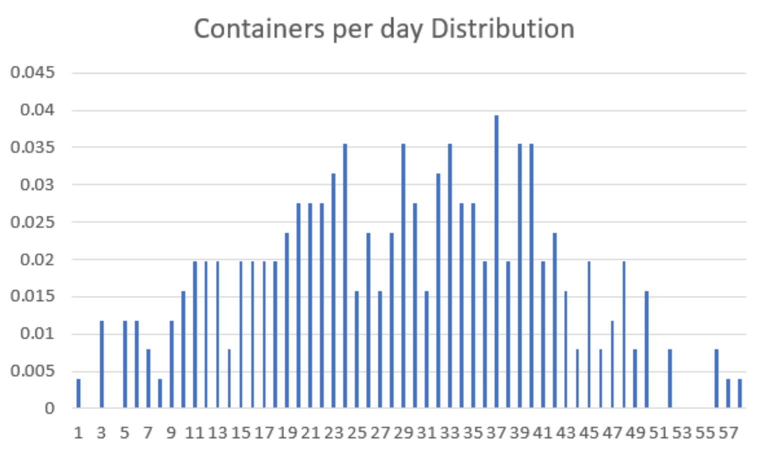

Figure 4.1. Graph of the number of containers distributed per day

The time windows of the trips are distributed as follows. The start of the time window is a whole hour between 6 AM and 10 AM, with the probabilities shown in Table 4-2. The earlier hours have a higher probability because there is a relatively high chance that the containers are already located at their start location at the start of the day. The duration of the time windows are either 3, 4, 5 or 6 hours, uniformly distributed.

Table 4-2. Distribution of start of time windows for the SAVED use case

Start Time Window	Probability
06:00	0.3
07:00	0.3
08:00	0.2
09:00	0.1
10:00	0.1

In terms of vehicles, the goal of the SAVED project is to let one electric AGV transport the containers between the depot and warehouses. In our algorithm, we assume that the average speed of the AGV is 20 km/h. The duration between the locations is then the Euclidean distance between those locations, multiplied by a distance factor of 1.2 and then divided by the speed. Furthermore, based on the vehicles of (Electric Vehicles, 2024), we assume that the battery capacity of the AGV is 236 kWh, while the energy consumption power is 12 kW and the charging power is 32 kW. Each container has a weight factor between 1 and 3. To experiment with our solution approach we also assume in most instances that the SAVED project consists of two vehicles, while also doubling the number of container trips. In our algorithm, each vehicle starts with a full battery and has to end with a full battery, enforced by the ALNS algorithm.

Since a trip on average costs 7 minutes and we need to account for travel time without carrying a container, we need approximately 2 kWh per trip. We assume that the SAVED use case uses solar panels with a 430-wattpeak (*Soly.Nl*, 2024), which each delivers approximately 1,2 kWh per day on an average day with 60-70% efficiency based on the calculation of *GlobalSolarAtlas.Nl* (2024). This means that we need approximately 1.7 solar panels per trip, however, to have more flexibility in the algorithm and account for days with lower solar panel efficiency, we use more solar panels in our data instances. We also experiment with different numbers of solar panels while keeping the number of container jobs constant. This all combined leads to the following 10 instance sizes (Table 4-3). During the experiments (except for experiment 0), we generate a specific instance with a set of container jobs randomly per replication. This means that if we do 5 replications for instance size S0, we generate 5 different instances with 10 container jobs, 1 truck, and 30 panels.

Table 4-3. Data instances industrial/SAVED use case

ID	Nr of Container Jobs	Nr of Trucks	Nr of Solar Panels
S0	10	1	50
S1	20	1	50
S2	30	2	100
S3	40	2	100
S4	50	2	100
S5	50	2	150
S6	60	2	120
S7	60	2	150
S8	70	2	150
S9	80	2	150

4.2.2 Campus Use Case

Since the Campus use case is a hypothetical use case, we have to come up with artificial data. To come up with artificial data, we pick 20 customer locations on the campus. These locations are either faculties, student houses, or sports associations. These locations are shown on the map in Figure 4.2. The red dot shows the depot and the blue dots show the customer locations.

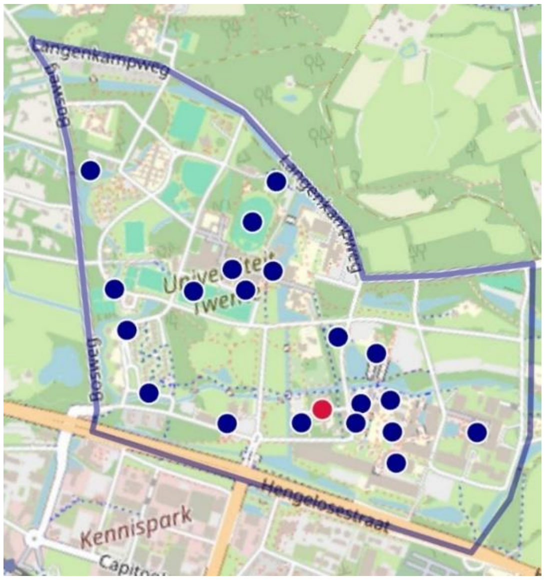

Figure 4.2. Map of the University with artificial customer locations

We assume that the number of trips from the depot to each location or back is uniformly distributed per location. The start of the time windows of the trips are whole hours between 7 AM and 3 PM and are uniformly distributed while the duration of time windows can either be 30 minutes, 1 hour, 1.5 hours, 2 hours, 3 hours, or 4 hours.

In this hypothetical use case, we can use two vehicle types, namely drones and street robots. We assume that drones fly with an average speed of 20 km/h, while their distance factor is 1.2. Their battery capacity is 0.75 kWh based on the drones used during the research of (Figliozzi, 2017), while their energy consumption power is 500 W when empty. This means that they can fly without a package for 90 minutes. The street robot has an average speed of 10 km/h, while its distance factor is 1.5. The battery capacity is 0.5 kWh, while its energy consumption power is 250 W when empty. The charging power for both vehicles is 500 W and each trip has a weight factor between 2 and 3. Each vehicle starts with a full battery and also has to end with a full battery. Since we do not use heavy electric trucks, but lighter drones, we need fewer solar panels. We also experiment with different fleet sizes for the same number of trips. This all combined leads to the following 10 data instances. (Table 4-4). Identically to the SAVED use case, we generate an exact instance per replication of the experiments.

Table 4-4. Data instances urban/Campus use case

ID	Nr of Jobs	Nr of Drones	Nr of Street Robots	Nr of Solar Panels
C0	50	3	0	5
C1	75	4	0	8
C2	75	3	1	8
C3	100	5	0	10
C4	100	4	1	10
C5	100	3	2	10
C6	200	9	3	20
C7	200	8	4	20
C8	200	7	5	20
C9	400	18	6	40

4.3 Parameter Tuning

4.3.1 Experiment Design

This experiment is dedicated to tuning the parameters of the ALNS method. This is crucial for balancing the accuracy and efficiency of our solution approach. The goal of the experiment is to identify the optimal settings for the ALNS to ensure that a near-optimal solution is reached within a reasonable computation time. The tuned parameters are the number of worse solutions allowed, the number of solutions created per iteration, the initial move percentage, the initial switch percentage, the initial random-charging percentage, the initial threshold dictionary, the initial waiting time dictionary, and the adaptivity percentage of all those parameters.

The first parameter that is optimized is the initial value of the battery threshold probabilities for partial charging. We test three different configurations as Table 4-5 shows, a uniform distribution, a distribution that focuses more on lower thresholds, and a distribution with more emphasis on higher thresholds. For example, if Configuration 2 is chosen, and a vehicle has to charge, the probability that the algorithm chooses 0.4 as the battery threshold is 0.3, except when the minimum threshold is higher than 0.4, then the probability is 0.

Table 4-5. Configurations for battery threshold parameter

Thresholds	Configuration 1	Configuration 2	Configuration 3
0.4	1/7	0.30	0.05
0.5	1/7	0.20	0.05
0.6	1/7	0.20	0.10
0.7	1/7	0.10	0.10
0.8	1/7	0.10	0.20
0.9	1/7	0.05	0.20
1	1/7	0.05	0.30

The second parameter is the initial probability distribution of the length of voluntary waiting times before charging (Table 4-6). We test a uniform distribution against a distribution that has higher probabilities for the lower waiting times.

Table 4-6. Configurations for waiting times parameter

Waiting times	Configuration 1	Configuration 2
0	1/5	0.4
10	1/5	0.2
20	1/5	0.2
30	1/5	0.1
60	1/5	0.1

The other parameters are shown in Table 4-7, and are the initial probabilities of moving a trip between vehicles, switching a trip with the next trip in the order within a vehicle, and the probability of going to a charging station while it is not necessary to charge (Move/Switch/Random Charging percentage). These three probabilities start on the same initial value, however, they will be adapted separately in the updating phase of the ALNS. The next parameter is the adaptivity percentage of all parameters described in this section. Per accepted solution, we calculate the actual percentages of for example the move percentage and update the parameter with the calculated actual percentage. The adaptivity percentage is then the weight we give to the actual percentage related to the parameter percentage.

The next parameter is the number of solutions created per iteration and the last parameter is the number of worse solutions. This is the termination criterion of our heuristic. If for example, the value is 5, this means that the algorithm stops after there are 5 successive iterations in which no better solution is found. All these parameters and values are tested using the full factorial method for six instances (S2, S5, S8, C2, C5, C8) with 5 replications per experiment so that we can come up with the best parameter settings for both use cases independently. However each replication is for the exact same data instance, so the exact same jobs/trips. In total this means we have 3*2*3*3*2*4 = 432 different parameter settings. To reduce the computation time, we can combine the values for the number of worse solutions in one experiment, since we can save our KPIs the first time 3 successive worse solutions are found and continue with the experiment.

Table 4-7. Configurations for remaining parameters

ID	Name	Values
0	Move/Switch/RandomCharging Percentage	(0.1, 0.2, 0.3)
1	Adaptivitypercentage	(0.2, 0.3, 0.5)
2	NrSolutionsPerIteration	(50,100)
3	NrWorseSolutions	(3; 5; 8; 10)

4.3.2 Experiment Results

SAVED Use Case

After running for all 432 different parameter settings, we know the objective values and the running times. The next step is to determine which setting is suitable for our research. For this we filter the settings, in order to only show the solution settings which are not dominated by another setting. Dominated settings are outperformed by another setting in both the objective value and the running time, and are therefore not suitable for the thesis. (Ngatchou et al., 2005) Table 4-8 shows the 17 solutions that are not dominated by any other solution. The table shows per parameter setting the configuration for the battery threshold (BT), the configuration for the voluntary waiting times (WT), the Move/Switch/RandomCharging

Percentage (MSRC), the Adaptivity Percentage (AP), the NrSolutionsPerIteration (NRS), the NrWorseSolutions (NRW) the average of the best objective per instance, the average objective per instance, the average of the worst objective per instance, and the average running time (T(s)). The table is increasingly sorted on the average objective, and because it only shows non-dominated solutions, it is also decreasingly sorted on the running time

Table 4-8. Non-Dominated Solutions for Parameter Tuning SAVED use case

ID	**Parameter Settings**						**Experimental Results**			
	BT	WT	MSRC	AP	NRS	NRW	Best Obj	Average Obj	Worst Obj	t(s)
1	3	2	0.1	0.2	100	10	22.59	24.28	27.00	18.43
2	3	2	0.1	0.2	100	8	22.61	25.04	27.61	13.62
3	3	2	0.1	0.2	100	5	23.55	25.97	31.03	10.8
4	1	2	0.1	0.2	100	5	24.52	26.56	30.21	9.93
5	1	2	0.1	0.2	50	3	25.13	27.29	32.43	8.78
6	1	1	0.1	0.5	100	3	26.61	28.66	33.98	7.46
7	3	2	0.1	0.2	100	3	27.74	30.19	35.44	5.97
8	3	2	0.1	0.5	50	3	30.54	32.83	36.45	3.43
9	1	2	0.1	0.3	50	3	32.18	35.33	41.49	3.28
10	1	1	0.1	0.2	50	3	33.64	37.24	42.24	2.78
11	1	1	0.1	0.3	50	3	36.23	39.90	44.13	2.45
12	1	2	0.1	0.5	50	3	37.82	41.92	48.34	2.34
13	1	1	0.2	0.5	50	3	39.49	43.85	50.69	2.09
14	2	2	0.2	0.2	50	3	40.30	44.78	49.73	2.02
15	3	1	0.2	0.5	50	3	42.59	46.67	53.24	1.76
16	3	1	0.2	0.5	50	3	44.81	49.09	56.12	1.35
17	3	1	0.3	0.2	50	3	46.42	51.39	57.03	1.23

It can be seen that the parameter setting which leads to the best objectives, achieves this goal in 18.43 seconds. This is a relatively short computation time, so it can be considered the best parameter setting for this algorithm in the SAVED use case. However, in this thesis, we have 6 experiments for multiple efficiency settings, various instances, and numerous replications, so the emphasis for the experiments lies on reaching reasonably good solutions with a very short computation time. This means, that for the rest of this thesis, we chose the parameter setting with ID 8. This leads to a 35.2% worse objective than the setting that leads to the best objective, however, it reaches this solution in only 18.6% of the running time of the earlier setting.

Campus Use Case

For the parameter selection at the instances at the Campus use case, we use the same procedure as for the SAVED use case. The first step is to determine the list of settings that are not dominated. These solutions are shown in Table 4-9.

Table 4-9. Non-Dominated Solutions for Parameter Tuning Campus use case

ID	Parameter Settings						Experimental Results			
	BT	WT	MSRC	AP	NRS	NRW	Best Obj	Average Obj	Worst Obj	t(s)
1	1	1	0.1	0.3	100	10	1.90	1.95	2.07	37.27
2	2	2	0.2	0.2	100	10	1.86	1.97	2.08	29.66
3	1	1	0.2	0.3	100	10	1.81	1.99	2.27	28.68
4	1	2	0.1	0.2	50	10	1.91	2.00	2.07	17.07
5	1	2	0.1	0.2	50	8	1.98	2.01	2.08	13.84
6	2	2	0.2	0.2	100	3	1.93	2.03	2.35	10.07
7	1	2	0.1	0.3	50	5	1.90	2.04	2.21	9.28
8	2	2	0.2	0.2	50	5	1.90	2.06	2.25	8.62
9	1	2	0.1	0.2	50	5	2.01	2.08	2.29	8.53
10	1	1	0.2	0.3	100	3	1.90	2.09	2.37	7.12
11	2	2	0.2	0.2	50	3	1.98	2.12	2.40	3.69
12	2	2	0.2	0.3	50	3	2.06	2.15	2.25	3.61
13	1	2	0.1	0.2	50	3	2.01	2.18	2.35	3.33
14	2	2	0.1	0.3	50	3	2.10	2.20	2.34	3.13
15	3	1	0.1	0.2	50	3	2.12	2.22	2.31	2.24
16	2	1	0.3	0.5	50	3	2.01	2.31	2.85	2.03

It can be seen that the parameter setting which leads to the best objectives, achieves this goal in 37.27 seconds. This is again a relatively short computation time, so it can be considered the best parameter setting for this algorithm in the Campus use case. However, we chose a setting that is more focused on efficiency. In this case that is the setting with ID 11. This leads to a 8.7% worse objective than the setting that leads to the best objective, however, it reaches this solution in only 10.0% of the running time of the earlier setting.

4.4 Scenario Evaluation

4.4.1 Exact versus Heuristic

In this experiment, we test the performance of our developed ALNS against the mathematical model. Both the model and the ALNS are described in Chapter 3. The goal is to measure how well our ALNS performs. This experiment is performed for the SAVED use case however, the data instance sizes are changed. We create smaller instances, to have a smaller computation time for the Gurobi solver. Table 4-10 shows the 7 instance sizes which are used in this experiment. The maximum running time for the exact optimization method is set to 10 minutes. Per instance size, we perform 5 replications. The instances also have 0 solar panels, because the mathematical model does not account for solar charging. Therefore all energy is taken from the grid.

Table 4-10. Data Instances Exact vs Heuristic Experiment

ID	Nr of Container Jobs	Nr of Trucks
0	5	1
1	10	1
2	20	1
3	20	2
4	30	2

Table 4-11 shows the results of this experiment. It can be seen that the objective value of our heuristic is relatively close to the objective value of the exact optimization for the smallest two instances (the average difference is 5.6%), however, the running time for instance 1 is already 461 seconds, while the heuristic reaches its solution in 1.82 seconds. From instance 2 the heuristic outperforms the exact optimization in both the objective and running time. The reason for this is that the exact optimization did not come to the optimal solution within the maximum running time.

Table 4-11. Experimental Results Exact vs Heuristic

	Exact			**Heuristic**	
ID	**Obj**	**Gap (%)**	**t(s)**	**Obj**	**t(s)**
0	5.77.	0	0.16	6.01	0.77
1	6.70	0	461	7.15	1.82
2	22.83	90.2	600	16.26	2.07
3	20.87	92.3	600	15.95	2.28
4	-	-	600	23.28	2.47

From this experiment, we can conclude that the objective value of the ALNS is very close to the objective value of the exact optimization, while the computation time of the heuristic is much shorter than the computation time of the exact optimization. However, it should be noted that there is no solar charging involved in this experiment and the instances are very small with also only 3 different warehouses, so we cannot draw significant conclusions about the performance of our ALNS when solar charging is included and the instance sizes increase.

4.4.2 Collaboration

In the collaboration experiment, we test the influence of companies collaborating in the SAVED use case. The purpose of this experiment is to see how companies in the XL business park can reduce their charging costs and operating time by collaborating. Therefore, we compare the scenarios in which the companies do not collaborate with the scenarios in which they do collaborate.

In the scenarios in which the companies do not collaborate, it means that the vehicle cannot deliver a container from the depot to one company and then pick up another container at a different company and deliver it back to the depot. Instead, the vehicle first delivers a container from the depot to the company, then drives back to the depot empty, then drives to the second company and picks up the container to deliver it at the depot. However, the vehicle is allowed to deliver a container to one company and immediately pick up another container at the same company if there is a container ready to be picked up.

In the scenario in which the companies do collaborate, the vehicle is allowed to deliver a container to one company and immediately drive to another company to pick up a container there. The difference between these scenarios is modeled in the time matrix between companies. In the collaborating scenario, the time matrix is the normal matrix. In contrast, in the non-collaboration scenario, the time matrix values between the companies are equal to the travel time between the first company and the depot plus the travel time between the depot and the second company.

The experiment is executed for all data instances in the SAVED use case, with five replications and 4 seasons per instance size. Per replication, we generate a new set of container jobs and solve the instance for both the non collaboration scenario and the collaborating scenario.

Table 4-12. Experimental results Collaboration Experiment per instance size

	Non Collaboration					**Collaboration**				
ID	**Min**	**Obj**	**Max**	**EOD**	**t(s)**	**Min**	**Obj**	**Max**	**EOD**	**t(s)**
S0	7.47	8.23	9.12	12:43	1.39	6.41	7.40	8.70	12:42	1.63
S1	16.53	16.34	19.95	14:06	1.78	15.73	16.21	18.69	14:04	1.75
S2	13.17	15.21	18.25	14:30	2.74	12.39	14.11	14.37	14:28	2.67
S3	20.45	24.07	24.22	14:41	1.92	18.36	22.53	25.34	14:33	2.51
S4	33.00	34.88	36.35	15:19	2.89	30.29	32.36	34.00	15:15	2.34
S5	26.37	26.91	28.72	15:39	3.05	22.41	24.89	27.31	15:24	3.02
S6	39.96	42.78	45.83	16:12	2.84	36.37	41.42	43.77	15:59	2.81
S7	38.40	39.41	43.36	16:21	2.79	35.72	37.40	43.50	16:13	2.85
S8	50.70	56.34	66.64	17:17	2.21	46.25	54.67	61.98	17:01	3.04
S9	76.84	83.24	86.85	18:13	1.72	77.05	81.35	85.92	18:06	1.63
AVG	**32.29**	**34.74**	**37.95**	**15:30**	**2.33**	**30.10**	**32.01**	**36.36**	**15:21**	**2.42**

Table 4-12 shows the experimental results of both scenarios for all instance sizes. It shows per scenario the minimum objective (Min), the average objective value (Obj), the maximum objective (Max), the makespan of the operation, and the running time of the complete algorithm. It can be concluded that there is relatively little difference in terms of the overall average charging costs. The average difference is 2.94, which is 8.5% of the objective in the non collaboration scenario. The difference in time is only 10 minutes. The difference in objective value is low for the small instances. The reason for this might be that the vehicle is done operating at the start of the afternoon, and can start charging to get its battery level back to 100%. Charging around this time is very cheap since there is much solar energy available.

4.4.3 Variable Time Windows

In this section, we test the influence of different time window weights on the quality of the ALNS algorithm against a situation where solar charging is not taken into account in the algorithm. This means that we test two policies in this experiment. The 'Solar Panels' policy is just the standard policy of our ALNS algorithm, while in the 'No Solar Panels' policy, the algorithm solves the scenario as if there were 0 solar panels, then saves the policy and calculates the objective as if there are the standard number of solar panels.

The purpose of this experiment is to investigate the robustness of the algorithm for different experiment settings regarding time windows. We perform the experiment for both use cases. The variable that we experiment with is the probabilities of the start moment of the time window of each order. We experiment with one setting in which the probabilities are the highest for the earliest hours of the use case, one in which the probabilities are the highest for hours in the middle of the use case, one where the probabilities are the highest for hours at the end of the use case, and one where the probabilities are uniform. The exact probabilities per scenario can be found in Table 4-13. With these probabilities, we cannot compare the exact same job/trip instances per scenario. The consequence of this is that we generate new instances per scenario. This leads to more randomization between experiments, so we do 10 replications per experiment to reduce the variance of randomization.

Table 4-13. Probability Distribution Start Time Window per Experiment

SAVED use case					**Campus use case**				
StartTime	Start	Middle	End	Uniform	StartTime	Start	Middle	End	Uniform
06:00	0.3	0.1	0.1	0.2	07:00	0.3	0.05	0.05	0.11
07:00	0.3	0.2	0.1	0.2	08:00	0.2	0.1	0.05	0.11
08:00	0.2	0.4	0.2	0.2	09:00	0.1	0.1	0.05	0.11
09:00	0.1	0.2	0.3	0.2	10:00	0.1	0.15	0.05	0.11
10:00	0.1	0.1	0.3	0.2	11:00	0.1	0.2	0.1	0.11
					12:00	0.05	0.15	0.1	0.11
					13:00	0.05	0.1	0.1	0.11
					14:00	0.05	0.1	0.2	0.11
					15:00	0.05	0.05	0.3	0.11

Table 4-14 and Table 4-15 show the experimental results for the 4 different time window scenarios in the SAVED use case. It shows per time window scenario and instance size and per policy the average objective (Obj), the average percentage of solar energy used relative to the total amount of energy used (PSE) and the average running time (t(s)). The extended results including the minimum and maximum per instance can be found in Appendix A. It also shows the relative difference between the two policies per time window scenario. It can be seen that the average ALNS objective is the lowest when in the "Start" scenario. The reason for this is that, if the time windows are all at the start of the use case, the vehicle completes its operations earlier, and can charge in the middle of the day, when there is the most solar energy available.

It can also be seen that for all scenarios, the difference between the objectives of the policies is relatively the same. The difference is the largest when the time windows are uniformly distributed, namely €11.08 or 28.5% compared to the 'No Solar Panels' policy, while the relative difference is the lowest in the "Middle" scenario, with 24.2%.

Table 4-14. Experimental Results Time Windows SAVED use case 1/2

	Start							**Middle**						
	No Solar Panels			**Solar Panels**			**Dif**	**No Solar Panels**			**Solar Panels**			**Dif**
ID	**Obj**	**PSE**	**t(s)**	**Obj**	**PSE**	**t(s)**		**Obj**	**PSE**	**t(s)**	**Obj**	**PSE**	**t(s)**	
S0	9.65	0.33	0.76	8.27	0.38	1.44	-14.3%	8.64	0.34	0.72	7.13	0.37	1.41	-17.5%
S1	15.40	0.39	0.75	14.09	0.39	1.29	-8.5%	18.23	0.35	0.73	16.66	0.36	1.29	-8.6%
S2	22.99	0.45	0.79	13.52	0.72	1.54	-41.2%	22.54	0.43	0.74	12.80	0.73	1.61	-43.2%
S3	32.05	0.43	0.76	20.92	0.68	1.18	-34.7%	32.53	0.42	0.74	21.07	0.67	1.41	-35.2%
S4	40.38	0.39	0.77	27.73	0.62	1.30	-31.3%	43.90	0.32	0.73	30.45	0.57	1.51	-30.6%
S5	32.96	0.56	0.76	18.45	0.82	1.57	-44.0%	33.67	0.53	0.74	18.85	0.80	1.55	-44.0%
S6	52.52	0.34	0.76	36.96	0.56	1.76	-29.6%	54.57	0.28	0.75	41.90	0.47	1.76	-23.2%
S7	44.95	0.45	0.74	29.76	0.68	1.58	-33.8%	50.60	0.35	0.71	38.29	0.52	1.61	-24.3%
S8	61.46	0.31	0.71	49.48	0.45	1.51	-19.5%	62.92	0.26	0.64	58.04	0.33	0.92	-7.8%
S9	81.73	0.18	0.61	75.37	0.23	0.91	-7.8%	84.56	0.13	0.50	78.47	0.21	0.50	-7.2%
AVG	**39.41**	**0.38**	**0.74**	**29.46**	**0.55**	**1.41**	**-26.5%**	**41.22**	**0.34**	**0.70**	**32.37**	**0.50**	**1.36**	**-24.2%**

Table 4-15. Experimental Results Time Windows SAVED use case 2/2

	End							**Uniform**						
	No Solar Panels			**Solar Panels**			**Dif**	**No Solar Panels**			**Solar Panels**			**Dif**
ID	**Obj**	**PSE**	**t(s)**	**Obj**	**PSE**	**t(s)**		**Obj**	**PSE**	**t(s)**	**Obj**	**PSE**	**t(s)**	
S0	9.02	0.37	0.73	7.63	0.39	1.32	-15.5%	9.20	0.35	0.76	7.66	0.39	1.32	-16.7%
S1	16.46	0.36	0.75	14.99	0.38	1.36	-8.9%	17.92	0.36	0.74	15.57	0.38	1.44	-13.1%
S2	23.69	0.46	0.78	14.56	0.72	1.28	-38.5%	23.74	0.46	0.75	13.75	0.73	1.58	-42.1%
S3	35.61	0.37	0.74	23.55	0.64	1.65	-33.9%	32.35	0.41	0.75	20.41	0.68	1.54	-36.9%
S4	44.61	0.30	0.74	29.73	0.58	1.48	-33.4%	41.40	0.35	0.78	27.34	0.61	1.65	-34.0%
S5	38.43	0.43	0.74	18.43	0.80	1.56	-52.0%	34.64	0.51	0.76	16.62	0.85	1.77	-52.0%
S6	61.20	0.22	0.75	43.66	0.46	1.58	-28.7%	54.80	0.30	0.76	38.53	0.54	1.67	-29.7%
S7	60.17	0.25	0.72	43.99	0.48	1.65	-26.9%	49.74	0.39	0.75	33.94	0.62	1.44	-31.8%
S8	70.52	0.19	0.66	60.63	0.31	1.10	-14.0%	67.73	0.25	0.72	52.50	0.43	1.44	-22.5%
S9	82.13	0.15	0.59	76.84	0.20	0.84	-6.4%	87.38	0.13	0.67	81.80	0.17	0.96	-6.4%
AVG	**44.19**	**0.31**	**0.72**	**33.40**	**0.50**	**1.38**	**-25.8%**	**41.89**	**0.35**	**0.74**	**30.81**	**0.54**	**1.48**	**-28.5%**

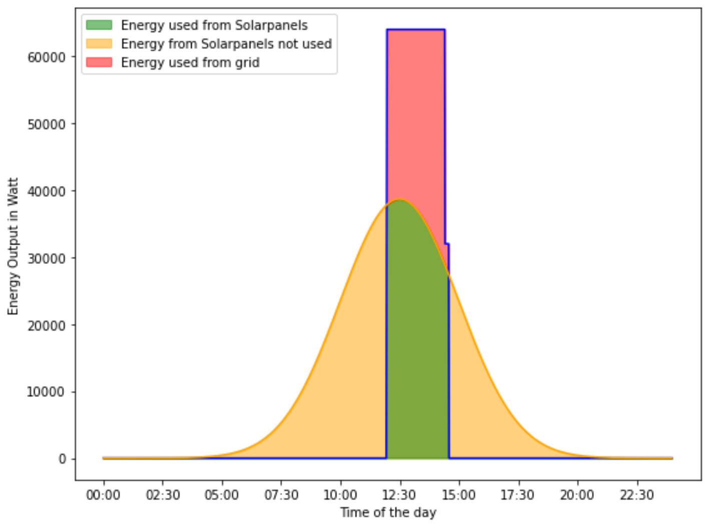

Figure 4.3. Example of Energy Graph No Solar Panel Policy SAVED use case

Figure 4.3 shows the solar output of the 'No Solar Panel' policy from an instance of the category S5. The green area is the solar energy used to charge a vehicle, while the orange area shows the energy not used. The red area shows the energy used from the grid. It can be seen that the strategy for the vehicles is to only charge one time, at the end of the operation. They are both at the same time done with operating (approx. 12:20) and go both to the charging station. This means that a relative high amount of energy needs to be used from the grid. Figure 4.4 shows the solar output of the 'Solar Panel' policy for the same instance. It can be seen that the first vehicle already starts charging at approximately 10:00. There is also only one small moment in which the two vehicles charge at the same time. This policy leads to a much smaller read area and therefore less energy used from the grid.

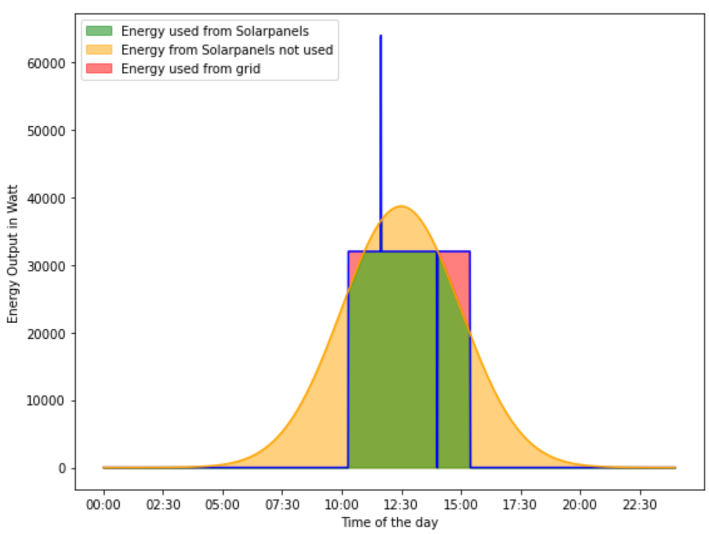

Figure 4.4. Example of Energy Graph Solar Panel Policy SAVED use case

Table 4-16 and Table 4-17 show the experimental results for the 4 different scenarios in the Campus use case. Again, the extended results can be found in Appendix A. It can be seen that the charging costs are the lowest in the "Start" scenario, and the percentage of solar energy used is the highest in the "Start" scenario, with 84%. This is logical, because when most trips have to be executed in the morning, most charging can take place after those trips when there is the most solar power available. The biggest relative difference between the objectives of the two policies is in the “Start” scenario and in the “Uniform” scenario with 25.2% and 24.8% respectively, while the difference is the lowest in the “End” scenario with only 16.8%. The reason for this is most probably that in the “End” scenario, the vehicles are done operating relatively late, and still have to charge for the next day. The algorithm can do nothing to prevent that from happening, so therefore the algorithm cannot make a difference.

Table 4-16. Experimental results Time Windows Campus use case 1/2

	Start							Middle						
	No Solar Panels			Solar Panels			Dif	No Solar Panels			Solar Panels			Dif
ID	Obj	PSE	t(s)	Obj	PSE	t(s)		Obj	PSE	t(s)	Obj	PSE	t(s)	
C0	0.72	0.60	1.46	0.48	0.88	2.26	-33.9%	0.75	0.65	1.88	0.54	0.83	2.14	-29.1%
C1	0.95	0.64	2.18	0.69	0.89	2.38	-27.7%	1.10	0.65	1.07	0.83	0.80	2.01	-24.2%
C2	1.14	0.64	1.96	0.83	0.89	3.63	-27.0%	1.16	0.70	0.94	0.92	0.84	1.35	-20.7%
C3	1.41	0.59	1.70	1.09	0.81	2.44	-22.9%	1.39	0.69	1.19	1.07	0.83	1.84	-22.8%
C4	1.52	0.62	1.54	1.16	0.84	2.19	-23.3%	1.82	0.62	1.02	1.53	0.71	0.93	-15.8%
C5	1.88	0.58	1.48	1.40	0.74	1.60	-25.4%	2.04	0.62	1.05	1.73	0.70	1.12	-15.1%
C6	2.93	0.66	3.54	2.30	0.82	2.18	-21.7%	3.53	0.66	1.51	3.02	0.74	1.47	-14.4%
C7	3.17	0.64	2.06	2.37	0.84	2.16	-25.4%	3.51	0.68	1.33	2.91	0.76	1.46	-17.2%
C8	3.07	0.67	2.26	2.31	0.86	2.48	-24.7%	3.72	0.66	1.84	3.08	0.75	1.39	-17.2%
C9	6.02	0.65	2.89	4.81	0.81	1.77	-20.2%	6.75	0.68	1.55	5.84	0.76	1.81	-13.4%
AVG	**2.28**	**0.63**	**2.11**	**1.74**	**0.84**	**2.31**	**-25.2%**	**2.58**	**0.66**	**1.34**	**2.15**	**0.77**	**1.55**	**-19.0%**

Table 4-17. Experimental Results Time Windows Campus use case 2/2

	End							Uniform						
	No Solar Panels			Solar Panels			Dif	No Solar Panels			Solar Panels			Dif
ID	Obj	PSE	t(s)	Obj	PSE	t(s)		Obj	PSE	t(s)	Obj	PSE	t(s)	
C0	1.10	0.45	1.10	0.84	0.59	1.77	-23.8%	0.84	0.59	2.39	0.58	0.82	3.87	-31.0%
C1	1.63	0.47	1.35	1.39	0.52	1.18	-14.5%	1.23	0.62	2.06	0.89	0.82	3.53	-27.4%
C2	1.87	0.52	0.98	1.55	0.59	1.15	-17.2%	1.34	0.65	1.82	0.99	0.80	2.88	-26.1%
C3	2.04	0.49	1.23	1.48	0.68	1.67	-27.7%	1.51	0.67	2.57	1.09	0.79	2.77	-27.6%
C4	2.21	0.52	1.49	1.95	0.54	1.60	-11.8%	1.80	0.64	2.02	1.35	0.76	2.27	-24.7%
C5	2.31	0.56	1.28	1.96	0.62	1.73	-15.3%	1.92	0.67	1.53	1.46	0.78	2.16	-24.0%
C6	4.95	0.45	1.74	4.33	0.45	1.60	-12.6%	3.65	0.65	2.70	2.80	0.76	2.56	-23.2%
C7	5.17	0.45	1.25	4.40	0.52	1.64	-14.9%	3.90	0.64	2.31	3.04	0.73	2.53	-22.0%
C8	5.25	0.44	1.23	4.54	0.49	1.50	-13.5%	4.09	0.64	2.38	3.19	0.73	2.48	-21.9%
C9	10.52	0.44	1.28	8.76	0.44	1.32	-16.7%	7.03	0.67	1.45	5.61	0.75	1.71	-20.2%
AVG	**3.71**	**0.48**	**1.29**	**3.12**	**0.54**	**1.52**	**-16.8%**	**2.73**	**0.64**	**2.12**	**2.10**	**0.77**	**2.68**	**-24.8%**

From this experiment, we can conclude that for the SAVED use case, the ALNS algorithm leads to relatively even improvements for all tested scenarios regarding the variability in the start of the time windows, while the overall charging costs are the lowest if in the "Start" scenario. For the Campus use case, we can conclude that the charging costs are the lowest if most trips have to be scheduled at the start of the day. The ALNS algorithm, however, works best in both the "Start" and "Uniform" scenario.

4.4.4 Variable Weather

In this experiment, we test with different weather settings. In the first experiments, we assumed the solar energy curve to be a normal distribution curve. However, in the real world, this is not the case. That is why we have to experiment with multiple weather types. In this experiment, we first investigate the influence of the overall efficiency of the solar panels on the charging costs. We experiment with 4 different overall efficiencies (0.3, 0.7, 0.8, 0.6) representing the 4 seasons.

In this section, we also experiment with different efficiencies during the day, since the solar power output can also vary during the day. We experiment with 5 different scenarios that reflect 5 different types of daily weather to see the quality of our algorithm compared to the 'No Solar Panel' policy. The efficiencies for the five experiments can be found in Table 4-18. The numbers in for example the 'Afternoon' scenario mean that in the afternoon the efficiency is 0.5 of the total daily efficiency. Figure 4.5 shows how the daily solar output looks in the 'Afternoon' scenario.

Table 4-18. Variable Weather Efficiencies per Hour per Experiment

Hour	Overall	Morning	Noon	Afternoon	SunnyNoon
6-7	1	0.5	1	1	0.5
7-8	1	0.5	1	1	0.5
8-9	1	0.5	1	1	0.5
9-10	1	0.5	1	1	0.5
10-11	1	0.5	1	1	0.5
11-12	1	0.5	0.3	1	1
12-13	1	0.5	0.3	0.5	1
13-14	1	1	0.3	0.5	1
14-15	1	1	1	0.5	0.5
15-16	1	1	1	0.5	0.5
16-17	1	1	1	0.5	0.5
17-18	1	1	1	0.5	0.5
18-19	1	1	1	0.5	0.5

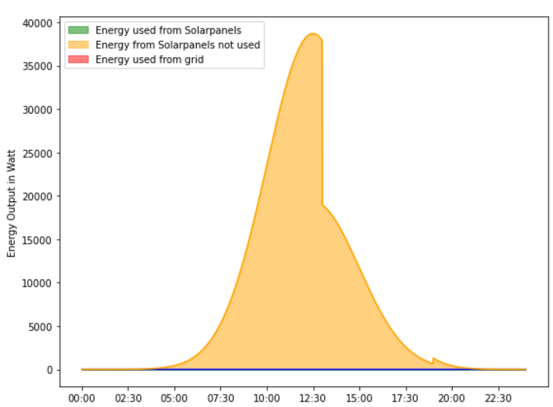

Figure 4.5. Example Solar Output Curve Afternoon Scenario

Saved Use Case

Table 4-19 shows the experimental results per efficiency setting for the SAVED use case. The table shows the averages of all instance sizes combined for both policies. It shows the average objective value of the ALNS (Obj), the percentage of solar energy relative to the total energy needed (PSE), the percentage of solar energy relative to the total energy available (PSA), and the running time of the algorithm (RT). The table shows that the efficiency and the objective of our algorithm are negatively correlated, which is logical. A higher efficiency leads to more solar energy available and therefore less use of energy from the grid.

The table shows that our algorithm performs relatively better as the efficiencies are higher. This makes sense, since the higher the efficiency, the more solar energy is available, which makes charging during the operation cheaper, and therefore the 'Solar Panels' policy, with its random charging, works relatively well.

Table 4-19. Experimental Results per Efficiency SAVED use case

	No Solar Panels				**Solar Panels**				**Dif(%)**
Efficiency	**Obj**	**PSE**	**PSA**	**t(s)**	**Obj**	**PSE**	**PSA**	**t(s)**	
0.3	60.84	0.23	0.41	1.02	50.52	0.37	0.64	2.58	**-17.0**
0.6	44.83	0.46	0.42	0.99	34.82	0.66	0.58	2.94	**-22.0**
0.7	40.2	0.50	0.37	1.13	30.2	0.76	0.56	3.07	**-27.4**
0.8	37.8	0.54	0.34	1.45	24.1	0.80	0.49	3.15	**-36.2**

Table 4-20 shows the experimental results for the different weather types. It can be seen that both policies perform best when there is the most solar energy available ('Overall'), which makes sense because then less energy from the grid is needed. It also shows that in that scenario, our algorithm performs significantly better than the 'No Solar Panel' policy, while that difference is lower in the other weather scenarios. This is logical since the more energy there is available, charging during the operation is an increasingly better strategy and therefore the difference between the two policies is higher.

Table 4-20. Experimental Results per Weather Type SAVED use case

	No Solar Panels				**Solar Panels**				**Dif(%)**
	Obj	**PSE**	**PSA**	**RT**	**Obj**	**PSE**	**PSA**	**RT**	
Afternoon	43.35	0.25	0.36	0.99	34.35	0.41	0.42	1.92	**-20.7**
Morning	38.95	0.30	0.43	1.02	31.30	0.43	0.46	2.00	**-19.6**
Noon	43.22	0.22	0.37	0.90	34.53	0.38	0.42	1.75	**-20.1**
Overall	39.17	0.40	0.45	0.74	29.03	0.58	0.49	1.41	**-25.8**
SunnyNoon	38.80	0.33	0.44	1.15	30.85	0.49	0.47	2.18	**-20.5**

Campus Use Case

Table 4-21 shows the experimental results per efficiency for the Campus use case. It can be seen that the algorithm relatively performs best (the relative difference is higher) when the efficiency is lower. The reason for this is that the 'No Solar panels' policy already delivers good results when the efficiency is higher. The average solar percentage is already 83%, which makes it more difficult to improve the solution. When the efficiency is lower there is more room for improvement, therefore our ALNS algorithm works relatively better when the efficiency is lower.

Table 4-21. Experimental Results per Efficiency Campus use case

Efficiency	No Solar Panels				Solar Panels				Dif(%)
	Obj	PSE	PSA	t(s)	Obj	PSE	PSA	t(s)	
0.3	2.43	0.61	0.72	1.03	1.73	0.77	0.83	2.26	**-28.8%**
0.6	2.02	0.71	0.42	1.08	1.48	0.87	0.50	2.45	**-26.7%**
0.7	1.75	0.8	0.41	1.11	1.36	0.88	0.45	3.35	**-22.2%**
0.8	1.63	0.83	0.35	1.07	1.32	0.94	0.40	2.56	**-19.0%**

Table 4-22 shows the experimental results per weather type for the Campus use case. It shows that the objective of the ALNS is the highest in the "Afternoon" and in the "SunnyNoon" scenario, while the relative difference between the two policies is also significantly lower in those scenarios. The reason for this is that both scenarios have less solar energy available in the afternoon, when most charging takes place, so, therefore, the operation has to use energy from the grid, regardless of the policy. The relative difference between the two policies is in the "Noon" scenario, mostly because in that scenario, the ALNS leads to a percentage of solar energy use of 75%, therefore having low feed-in costs.

Table 4-22. Experimental Results per Weather Type Campus use case

	No Solar Panels				Solar Panels				Dif(%)
	Obj	PSE	PSA	t(s)	Obj	PSE	PSA	t(s)	
Afternoon	2.21	0.71	0.51	2.08	2.02	0.77	0.57	2.98	**-8.5**
Morning	2.25	0.67	0.55	2.07	1.68	0.83	0.65	3.05	**-25.3**
Noon	2.43	0.63	0.56	2.11	1.75	0.83	0.75	4.09	**-28.0**
Overall	2.03	0.76	0.45	2.15	1.58	0.88	0.53	3.75	**-22.2**
SunnyNoon	2.44	0.64	0.48	1.99	2.21	0.72	0.55	3.12	**-9.4**

From this experiment in the SAVED use case, we can conclude that the ALNS algorithm performs relatively best when the efficiency is the highest, while in the Campus use case the algorithm performs best when the efficiency is the lowest. We can also conclude that the algorithm in the SAVED use case, performs relatively the same for the tested weather types, while for the Campus use case, the algorithm performs not so well when there is little energy available in the afternoon compared to the other scenarios.

4.4.5 Battery Usage

In experiment 5 we modify our solution approach to include the option of using a battery to store the energy not directly used. This battery can be used at the end of the day to charge the vehicles to their battery threshold to prepare them for the next day. The purpose of this experiment is to see whether we can reach the same performance with fewer solar panels since we can save the energy we do not use, so we can use all solar energy available.

This experiment is executed only for the Campus use case. The reason for this is that if a battery is used in the SAVED use case, there is enough battery capacity to execute all trips without intermediate charging, so with the use of a battery, all solar energy is stored and therefore running the ALNS algorithm has no purpose. This means that if the XL Businesspark wants to use a battery on a daily basis, it needs a battery of 236 kWh and enough solar panels to generate 236 kWh, which is on average 196 panels.

In the scenario that a battery is used, the vehicles in the ALNS algorithm do not have to be charged to 100% at the end of the day. All solar energy not used immediately during the day, is stored in a battery and is used after the algorithm to charge the vehicles to 100%. If there is not enough energy in the battery, energy from the grid is used, and if there is energy left in the battery, it is delivered back to the grid. We test this scenario against the scenario in which no battery is used, therefore the vehicles have to be charged to 100% in the algorithm. Since this experiment purely focuses on the difference in charging costs between the two scenarios, the only objective are the charging costs and not traveling time.

With the use of a battery, fewer solar panels are needed to charge the vehicle, since all energy from the solar panels can be used if necessary. Therefore, we also experiment with half the number of solar panels per instance size and three-quarters of the number of solar panels per instance size.

Table 4-23 shows the experimental results for the scenarios with and without the use of the battery when the normal number of solar panels is used. It shows the objective of the ALNS (Obj) and the running time t(s). For the "Battery" scenario, it also shows the percentage of instances in which the Campus is self-sustaining (PSS), which means that no energy from the grid has to be used at the end of the day. It can be seen that the objective of the algorithm with the use of a battery is on average €0.35 or 23.2% better compared to the scenario where no battery is used. It can also be seen that on average in 98% of instances, the campus is self-sustaining.

Table 4-23. Experimental Results Battery use Campus use case

	No Battery		**Battery**		
ID	**Obj**	**t(s)**	**Obj**	**PSS**	**t(s)**
C0	0.56	2.70	0.26	1.00	1.64
C1	0.81	3.49	0.59	1.00	2.62
C2	0.86	2.83	0.60	1.00	2.80
C3	1.04	1.55	0.89	1.00	1.60
C4	1.06	3.08	0.67	1.00	2.32
C5	1.07	3.17	0.76	1.00	1.72
C6	1.76	1.83	1.45	0.85	2.10
C7	2.08	3.76	1.52	0.9	2.64
C8	2.07	4.10	1.53	1.00	3.21
C9	3.85	0.71	3.31	1.00	1.37
AVG	**1.51**	**2.72**	**1.16**	**0.98**	**2.20**

Table 4-24 shows the average objective value, the average self-sustaining percentage, and the average running time for different percentages of solar panels used in combination with a battery. The values are the averages of all instance sizes. It can be seen that the objective value is the lowest when three-quarters of the solar panels are used. While the self-sustaining percentage is lower than when all panels are used, leading to more costs for energy used from the grid, less solar energy is delivered back to the grid, so overall it results in lower charging costs.

Table 4-24. Experimental Results Battery use Percentage Solar Panels Campus use case

Solar Panels	Obj	PSS	RT
50%	1.75	0.16	2.71
75%	0.83	0.68	2.29
100%	1.16	0.98	2.20

From this experiment, it can be concluded that the use of a battery would lead to a 23.2% improvement in costs if the same number of solar panels is used. Also in 98% of experiments, the Campus would be self-sustaining if a battery is used. However, because the costs without battery usage are already relatively low, buying a battery is not rewarding. It can also be concluded that the overall objective is lower when only 75% of solar panels are used. The reason for this is that there are on average fewer feed-in costs.

4.4.6 Soft Time Windows

In this experiment, we test the influence of soft time windows on the outcome of the solution. This means that time windows can be violated, leading to penalty costs. This means that our algorithm will have two objectives, namely the charging costs and the penalty costs per minute of time window violation. Again, the traveling time is not considered as an objective. Table 4-25 and Table 4-26 show the weights of both costs in the experiments in both use cases. The reason that the weights for the charging costs are much higher in the Campus use case than for the penalty costs is that if we want the costs of charging one minute at the Campus use case to equal the costs per minute of missing the time window, we already need to set the weight for the charging costs to 400 since charging for a minute costs 1/400 of a euro. In the SAVED use case, charging for a minute costs 1/6.25 of a euro.

Table 4-25. Objective Weights for Charging Costs and Penalty Costs per Scenario SAVED use case

ID	Weight Charging Costs	Weight Penalty Costs
ES0	0	1
ES1	1	1
ES2	2	1
ES3	5	1
ES4	6.25	1
ES5	10	1
ES6	1	0

Table 4-26. Objective Weights for Charging Costs and Penalty Costs per Scenario Campus use case

ID	Weight Charging Costs	Weight Penalty Costs
EC0	0	1
EC1	1	1
EC2	10	1
EC3	100	1
EC4	400	1
EC5	1000	1
EC6	1	0

Table 4-27 and Table 4-28 show the average penalty costs, the average charging costs, and the average running time per experiment. It shows, logically, that the bigger the relative weight is for the charging costs, the lower the charging costs and the higher the penalty costs. It can also be seen that the difference in charging costs when the objective weight of the charging costs is 1, is €17.14 or 44.3% in the SAVED use case €0.22 or 12.5% in the Campus use case of the charging costs when the objective weight is 0. The reason for this significant change is that when the objective for the charging costs is 0, the algorithm stops when the penalty costs become 0 (after all, the total objective is then 0 and cannot be improved). When the objective weight is 1, the algorithm continues after the penalty costs become 0, to find improvement in the charging costs while keeping the penalty costs 0. This can also be concluded from the difference in running time between the two experiments, where the experiment in which the charging cost weight is 1, runs for 46% in the SAVED use case and 42% in the Campus use case longer than when the objective weight is 0. It can also be concluded that the penalty costs only start to rise significantly when the weight of the charging costs reaches the point where one minute of charging from the grid is as expensive as missing the time window by one minute.

Table 4-27. Experimental Results Soft Time Windows Experiment SAVED use case

ID	Charging Costs	Penalty Costs	t(s)
ES0	38.62	0	2.43
ES1	21.48	0.05	4.44
ES2	21.18	0.08	4.98
ES3	18.29	0.23	5.22
ES4	15.18	7.04	6.34
ES5	14.83	7.32	6.78
ES6	12.92	392.62	8.31

Table 4-28. Experimental Results Soft Time Windows Experiment Campus use case

ID	Charging Costs	Penalty Costs	t(s)
EC0	1.76	0.22	2.66
EC1	1.54	0.22	3.79
EC2	1.54	1.30	2.64
EC3	1.51	1.41	2.67
EC4	1.43	11.18	3.34
EC5	1.41	39.24	3.60
EC6	1.36	293.90	7.01

Figure 4.6 and Figure 4.7 show two different usage graphs for different objective weights for instance C10 when the efficiency is 0.8. In Figure 4.6, the weights of Experiment EC0 are used, while in Figure 4.7 the weights of Experiment EC5 are used. It can be seen in Figure 4.6 that there is a significant energy peak at the end of the day, which almost completely has to be energy used from the grid. This is because the emphasis lies on meeting the time windows, and therefore a significant amount of charging is still required after the last trips. In contrast, when the emphasis only lies on the charging costs, there is no peak at the end of the day. At the end of the day, the energy usage line tries to follow the energy available line as close as possible to minimize the charging costs, and therefore accept the consequential penalty costs.

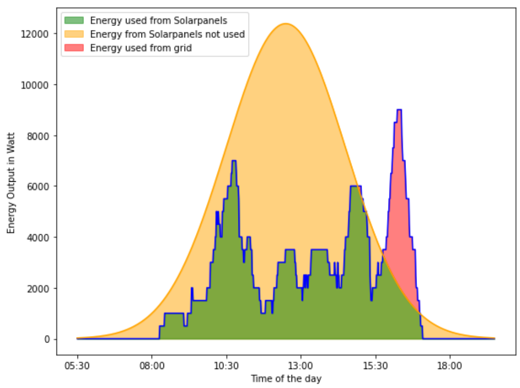

Figure 4.6. Example of Energy Graph High Penalty Costs Weight Campus use case

From these experiments, it can be concluded that the relatively different weights for the objectives charging costs and penalty costs have a significant impact on the outcome of the algorithm. The differences are the highest when the costs of one minute of charging with energy from the grid exceed the costs of missing the time window by one minute because if that is the case, the algorithm focuses more on the minimization of the charging costs than on meeting the time windows.

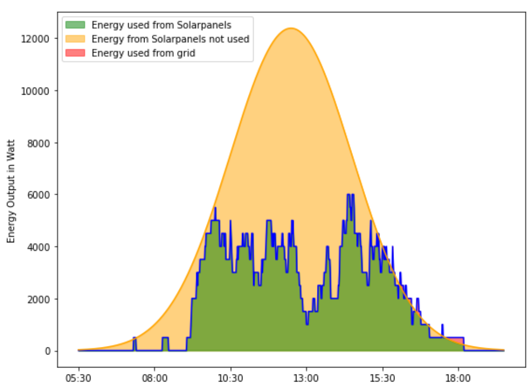

Figure 4.7. Example of Energy Graph High Charging Costs Weight Campus use case

4.5 Conclusion

In this chapter, we first defined 7 different experiments for one or both the industrial and urban use cases. The goal of these experiments was to test how well our solution approach performed in multiple scenarios. Then we parametrized our industrial use case by doing a context analysis, to come up with multiple instance sizes with either 1 or 2 vehicles. For the urban use case, artificial data instances were created representing the Campus area.

The first experiment was dedicated to tuning the parameters of our ALNS algorithm to be as efficient as possible. The goal was to balance between the quality of our solution and the computation time reaching that solution. For both use cases, we did a full factorial experiment regarding the parameter settings and found a setting that comes up with reasonably good solutions in a very short computation time.

In the second experiment, we tested our algorithm performance against the mathematical model. The goal was to see how close our algorithm could come to the optimal solution for small instances in the SAVED use case without the use of solar energy. It showed that in those instances, the performance of the ALNS is slightly worse than the exact optimization, while the computation time of the algorithm is significantly shorter. However, this cannot be used to draw conclusions about the ALNS performance when solar energy is used or when the instance sizes get as large as in the Campus use case.

In the third experiment, we tested for the SAVED use case the benefits in terms of charging costs and total duration of collaboration between the companies at the XL business park. The benefits are only significant in the large instance sizes.

In the fourth experiment, we tested the robustness of the algorithm by varying the time window distribution. We tested our ALNS algorithm against a “No Solar Panel” policy in which solar charging was not taken into account. For the SAVED use case, we conclude that ALNS leads to relatively even improvements for all tested scenarios compared with the “No Solar Panel” policy, while for the Campus use case, the conclusion is that the ALNS works best when the trips have a higher probability of being scheduled in the morning or uniformly.

In the fifth experiment, we tested what the influence of the weather was on the solution outcome of the ALNS. We can conclude for the SAVED use case, that a higher efficiency leads to more relative improvement of the ALNS in comparison to the “No Solar Panel” policy, while for the Campus use case, the ALNS method performs relatively better for lower efficiencies. We can also conclude that in the SAVED use case, the algorithm performs relatively the same for all tested weather types, while for the Campus use case, the algorithm performs relatively worse when there is little solar energy available in the afternoon.

In the sixth experiment, we tested how the use of a battery influenced the solution performance in the Campus use case. The conclusion is that a battery does have an influence on lowering the charging costs, however, it still leads to a significant amount of feed-in costs at the end of the day. This could be improved by saving energy for other days with low efficiencies. In the last experiment, we performed a sensitivity analysis on the influence of soft time windows and the relative weights between the penalty costs and the charging costs for both use cases. The conclusion is that the relative weights have a significant influence, but only after the weight of charging one minute from the grid exceeds the weight of missing the time windows by one minute.

5 Conclusion

This chapter ends this thesis by drawing conclusions regarding this research, together with recommendations, contributions, limitations, and ideas for future research. In Section 5.1, we summarize the outcome of this research by providing insights based on the outcome of the experiments with the ALNS in both the industrial and the urban use cases and answering the main research question. In Section 5.2, we discuss the contributions of this thesis to both the practice and theory. In Section 5.3, recommendations are provided based on our conclusions, while in Section 5.4 we acknowledge the limitations of our research and provide ideas for further research.

5.1 Conclusions

This research started with an introduction to the use cases, a research goal, and corresponding research questions and research methodology. Then the first step was an extensive literature review of different routing concepts and problems. Concepts that were discussed were classic VRPs, autonomous VRPs, Container Drayage Operations, and EVRP. In the world of EVRP, we reviewed different charging strategies, mixed fleets, and time dependencies. Then, we discussed the closest papers to our research and looked at possible solution methods. From the literature search, we conclude that much research had been done on these different concepts, but not on the combinations of those concepts for our specific use cases. Furthermore using the closest papers, the conclusion was that an ALNS algorithm would be best suitable as a metaheuristic for the solution approach to include solar charging in routing problems.

The next step was to design the solution approach. First, insights on the use cases were provided together with a conceptual graph that models the container jobs and trips as the nodes and the distances between the end location of one trip and the start location of another trip as the edges. This was the basis for the exact problem formulation which fits both our use cases. Also, a list with assumptions necessary to model our problem was stated. Then a mathematical model was provided for the problem without the use of solar charging and validated using a small toy problem representing the SAVED use case. To include solar charging in the solution approach, a Constructive Solution together with an ALNS was introduced. The ALNS iteratively destroys parts of the solution and rebuilds it using move operations, switch operations, partial charging, random charging when it is not necessary, and waiting before charging.

Then, multiple experiments were performed with our solution approach in the industrial area and urban area use cases. We first defined the parameter settings for both use cases, to ensure that the approach works as efficiently as possible, balancing between good solutions and computation time. After this, multiple experiments were executed to test the ALNS approach against either the mathematical formulation or the "No Solar Panel" policy.

From the experiments in the industrial area use case, it can be concluded that the ALNS delivers slightly worse solutions compared to the mathematical model in the small instances, in a shorter computation time, if solar charging is neglected. However, for the larger instances, the heuristic did outperform the exact optimization in both running time and objective value, because the exact optimization did not reach the optimal solution within the time limit.

In the comparison between the "No Solar Panel" policy and our heuristic, our heuristic performed on average 25.2% better in terms of costs than the "No Solar Panel" policy, while for some instance sizes, the number increases to 44%. number was for some instance sizes. Another conclusion is that collaboration between the companies at the XL business park leads to an improvement in terms of charging costs and the total duration of the logistical operation. In the smaller instances, the improvement is relatively small, while in the bigger instances, the improvements are more significant. The conclusion is also that the algorithm is robust for multiple scenarios regarding time windows, with each scenario delivering the approximate same relative improvement. The ALNS is also robust for different weather setting experiments, because the improvement is relatively equal for all those tests, while it performs relatively best in high efficiencies.

From the experiments in the Campus use case, it can also be concluded that the ALNS leads to significant improvements compared to the ‘No Solar Panel’ policy. On average, when the time windows are uniformly distributed, 77% of the energy needed for executing the trips, comes from solar panels, while for some instances this goes up to 82%, while the ‘No Solar Panel’ policy only reaches 64%. The relative improvements of the ALNS are the lowest when most trips have to be executed at the end of the day because there is less solar energy available at that moment to charge the vehicles. Our heuristic works better when the efficiency is the lowest, while in terms of weather types, it performs relatively worse if there is less solar energy available in the afternoon. Another conclusion is that the usage of a battery does lead to lower charging costs and a higher self-sustaining percentage since all available energy can be used, however, this also leads to higher feed-in costs. Since the charging costs in the Campus use case are low without using a battery, buying a battery is not rewarding. The last conclusion for both use cases is that when time windows are soft, choosing the relative weights of the penalty costs and the charging costs is very important because a change in those weights can significantly alter the solution.

To answer our main research question, sustainable charging can be integrated into electric vehicle routing problems in industrial and urban use cases using a metaheuristic approach. In this thesis, the chosen metaheuristic was a constructive heuristic combined with an ALNS. In the ALNS, sustainable charging is included with the use of partial charging, random charging when it is not strictly necessary, or waiting before charging. This leads to a sustainable solution, in which the emphasis lies on minimizing charging costs, which consists of using energy from the grid or delivering energy back to the grid. The solution approach is tested for multiple scenarios in both an industrial use case and an urban use case and leads to significantly better solutions compared to the “No Solar Panel” policy in a short computation time.

5.2 Contributions

5.2.1 Contribution To Practice

The main contribution to the practice of our research lies in the SAVED use case. The research has shown that the XL business park can reduce its charging costs with the help of the ALNS algorithm, and charge during the operation instead of after the operation. This is the difference between the ‘No Solar Panel’ policy and the ALNS algorithm. The results show that on average the XL-businesspark can save 25.2% of its costs with the ALNS algorithm compared to the “No Solar Panel” policy. It also shows that the companies can save charging costs by collaborating, mainly if many containers need to be transported during the day.

The other contribution to the practice is to algorithm itself and the generalizability to other use cases. The method can be easily adapted to other use cases, by changing the parameters of the locations, container jobs/trips, vehicles, and/or solar panels. In the research, we experimented solely with autonomous vehicles, however this is not necessary. The only two conditions for the vehicles are that they should be electric vehicles and that they have single-unit capacity constraints, the algorithm can be used to make tactical and operational decisions with the logistical schedule of when to charge and when to transport, corresponding with the chosen objective weights.

5.2.2 Contribution To Theory

The first contribution to theory is an extensive literature review, which maps the concepts in the world of EVRP, combined with other VRP problems such as autonomous VRP and Container Drayage Operations. Furthermore, to the best of our knowledge, the exact problem of our research, which is a combination of EVRP with solar charging and Container Drayage operations, is new in the literature. Therefore, the created mathematical and the solution approach in the form of a constructive heuristic combined with an ALNS contribute to the theory. Also, a detailed analysis of the ALNS for different scenarios and a parameter tuning experiment show the quality of the solution approach in both the objective value and the computation time.

5.3 Limitations

The main limitations of this research lie in the assumptions. The first assumption that limits the scope of the research is that the vehicles have a single-unit capacity. This fits with the use cases in this thesis since the trucks at the XL business park can only transport one container at a time, and the drones and street robots at the hypothetical Campus use case can also only transport one package at a time. However, this limits the generalizability to other routing problems in which vehicles can transport more packages at a time, such as delivery vans. An idea for further research would be to include this option in the solution approach, to research how the solution approach would impact these routing problems.

Another assumption that limits this research is the assumption that all vehicles can be charged at the same time. In the larger Campus use case instances, this could lead to solutions in which more than 10 vehicles charge at the same time, to use the available solar energy as effectively as possible. In most real-world instances, there is a limit to the number of charging stations, which has a significant impact on the charging schedule of the vehicles. We also assumed that the vehicles would charge linearly. In future research, an idea is to use non-linear charging, in which the vehicles can charge relatively quickly to 80% and then the charging speed slows down for the last 20%.

5.4 Recommendations and Further Research

Based on the outcome of the experiments regarding the industrial use case, the recommendation to the XL business park is to use this solution approach daily to use the available solar energy as effectively as possible and reduce the charging costs. Although only one vehicle is used to transport the containers over the park, an efficient combination of charging and transporting can lead to a significant reduction of costs. Furthermore, it is a recommendation for the companies at the XL business park to collaborate to reduce operating time and charging costs. This can be done by the warehouses communicating, to the planners at CTT at the start of the day about the number of containers ready to be picked up and the status of other containers. The planners at CTT can then use that information to create a schedule to operate as efficiently as possible.

The next recommendation is to use our heuristic as a starting point and expand it with for example real-world data. Using real-world data such as accurate weather forecasts or travel routes would further improve the accuracy of the approach. In this research, we only used average efficiencies of solar panels per day or per hour as input for the experiments, while in terms of distances, we used the Euclidean distance between locations combined with a distance factor. Changing these two parameters to real-world data would improve the approach and therefore lead to more accurate results.

Another idea for further research is to use this approach to make strategic decisions for both use cases in terms of the number of vehicles and the number of solar panels needed to minimize overall costs. For the number of solar panels, there exists an optimum that minimizes the total costs of using energy from the grid and feed-in energy to the grid. Furthermore, it would be interesting how other parameter settings, which lead to better solutions in a longer computation time, would influence the solution's objective value in the experiments. The last idea for further research is to test this solution approach on other but similar use cases, to further test the robustness of the approach.

Bibliography

Abualigah, L., Hanandeh, E. S., Zitar, R. A., Thanh, C. Le, Khatir, S., & Gandomi, A. H. (2023). Revolutionizing sustainable supply chain management: A review of metaheuristics. *Engineering Applications of Artificial Intelligence*, *126*, 106839. https://doi.org/10.1016/J.ENGAPPAI.2023.106839

Afroditi, A., Boile, M., Theofanis, S., Sdoukopoulos, E., & Margaritis, D. (2014). Electric Vehicle Routing Problem with Industry Constraints: Trends and Insights for Future Research. *Transportation Research Procedia*, *3*, 452–459. https://doi.org/10.1016/J.TRPRO.2014.10.026

Agarwal, Y., Mathur, K., & Salkin, H. M. (1989). A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, *19*(7), 731–749. https://doi.org/10.1002/net.3230190702

Cheikhrouhou, O., & Khoufi, I. (2021). A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy. *Computer Science Review*, *40*, 100369. https://doi.org/10.1016/J.COSREV.2021.100369

Cheng, C., Adulyasak, Y., & Rousseau, L. M. (2020). Drone routing with energy function: Formulation and exact algorithm. *Transportation Research Part B: Methodological*, *139*, 364–387. https://doi.org/10.1016/J.TRB.2020.06.011

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

Conrad, R. G., & Figliozzi, M. A. (2011). *The Recharging Vehicle Routing Problem*.

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

Desaulniers, G., Errico, F., Irnich, S., & Schneider, M. (2016). Exact Algorithms for Electric Vehicle-Routing Problems with Time Windows. *Operations Research*, *64*(6), 1388–1405. https://doi.org/10.1287/opre.2016.1535

Dessouky, A., & Yao, M. (2023). *Routing of Battery Electric Heavy Duty-Trucks for Drayage Operations*. https://doi.org/10.7922/G27M068R

Dorling, K., Heinrichs, J., Messier, G. G., & Magierowski, S. (2017). Vehicle Routing Problems for Drone Delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *47*(1), 70–85. https://doi.org/10.1109/TSMC.2016.2582745

Eksioglu, B., Vural, A. V., & Reisman, A. (2009). The vehicle routing problem: A taxonomic review. *Computers and Industrial Engineering*, *57*(4), 1472–1483. https://doi.org/10.1016/j.cie.2009.05.009

Elatar, S., Abouelmehdi, K., & Riffi, M. E. (2023). The vehicle routing problem in the last decade: variants, taxonomy and metaheuristics. *Procedia Computer Science*, *220*, 398–404. https://doi.org/10.1016/J.PROCS.2023.03.051

Erdoĝan, 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

Fan, L., Liu, C., Dai, B., Li, J., Wu, Z., & Guo, Y. (2023). Electric vehicle routing problem considering energy differences of charging stations. *Journal of Cleaner Production*, *418*, 138184. https://doi.org/10.1016/J.JCLEPRO.2023.138184

Felipe, Á., Ortuño, M. T., Righini, G., & Tirado, G. (2014). A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transportation Research Part E: Logistics and Transportation Review*, *71*, 111–128. https://doi.org/10.1016/j.tre.2014.09.003

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

Ferro, G., Paolucci, M., & Robba, M. (2018). *An Optimization Model For Electrical Vehicles Routing with time of use energy pricing And partial Recharging*. *51*(9), 212–217. https://doi.org/10.1016/J.IFACOL.2018.07.035

Figliozzi, M. A. (2017). Lifecycle modeling and assessment of unmanned aerial vehicles (Drones) CO 2 e emissions. *Transportation Research Part D: Transport and Environment*, *57*, 251–261. https://doi.org/10.1016/j.trd.2017.09.011

Ghorbani, E., Alinaghian, M., Gharehpetian, Gevork. B., Mohammadi, S., & Perboli, G. (2020). A Survey on Environmentally Friendly Vehicle Routing Problem and a Proposal of Its Classification. *Sustainability*, *12*(21), 9079. https://doi.org/10.3390/su12219079

Giuliano, G., Dessouky, M., Dexter, S., Fang, J., Hu, S., Steimetz, S., Miller, M., & Fulton, L. (2020). *Developing Markets for Zero Emission Vehicles in Short Haul Goods Movement*. https://doi.org/10.7922/G2DN43BT

GlobalSolarAtlas.nl. (2024). https://globalsolaratlas.info/map?c=52.311837,6.634712,12&s=52.320059,6.631966&m=site

Gnegel, F., Schaudt, S., Clausen, U., & Fügenschuh, A. (2021). *A 2D Layered Graph Approach for Scheduling Delivery Robots*. https://doi.org/10.26127/btuopen-5493

Goeke, D., & Schneider, M. (2015). Routing a mixed fleet of electric and conventional vehicles. *European Journal of Operational Research*, *245*(1), 81–99. https://doi.org/10.1016/J.EJOR.2015.01.049

Ham, A., & Park, M.-J. (2016). Electric Vehicle Route Optimization Under Time-of-Use Electricity Pricing. *IEEE Access*, *9*, 37220–37228. https://doi.org/10.1109/ACCESS.2021.3063316

Heilig, L., Lalla-Ruiz, E., & Voß, S. (2017). port-IO: an integrative mobile cloud platform for real-time inter-terminal truck routing optimization. *Flexible Services and Manufacturing Journal*, *29*(3–4), 504–534. https://doi.org/10.1007/s10696-017-9280-z

Hiermann, G., Puchinger, J., Ropke, S., & Hartl, R. F. (2016). The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations. *European Journal of Operational Research*, *252*(3), 995–1018. https://doi.org/10.1016/j.ejor.2016.01.038

Hossain, Md. S., Jahid, A., Islam, K. Z., & Rahman, Md. F. (2020). Solar PV and Biomass Resources-Based Sustainable Energy Supply for Off-Grid Cellular Base Stations. *IEEE Access*, *8*, 53817–53840. https://doi.org/10.1109/ACCESS.2020.2978121

How much electricity do solar panels produce? (2022). Nea.Org. https://www.nea.org.uk/who-we-are/innovation-technical-evaluation/solarpv/how-much-electricity-solar-produce/

Ileri, Y., Bazaraa, M., Gifford, T., Nemhauser, G., Sokol, J., & Wikum, E. (2006). An optimization approach for planning daily drayage operations. *Central European Journal of Operations Research*, *14*(2), 141–156. https://doi.org/10.1007/s10100-006-0165-6

Jula, H., Dessouky, M., Ioannou, P., & Chassiakos, A. (2005). Container movement by trucks in metropolitan networks: modeling and optimization. *Transportation Research Part E: Logistics and Transportation Review*, *41*(3), 235–259. https://doi.org/10.1016/J.TRE.2004.03.003

Kallehauge, B., Larsen, J., Madsen, O. B. G., & Solomon, M. M. (2005). Vehicle Routing Problem with Time Windows. In *Column Generation* (pp. 67–98). Springer-Verlag. https://doi.org/10.1007/0-387-25486-2_3

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

Koç, Ç., & Laporte, G. (2018). Vehicle routing with backhauls: Review and research perspectives. *Computers & Operations Research*, *91*, 79–91. https://doi.org/10.1016/J.COR.2017.11.003

Kumar, A., Aggarwal, A., Rani, A., Bedi, J., & Kumar, R. (2023). A mat-heuristics approach for electric vehicle route optimization under multiple recharging options and time-of-use energy prices. *Concurrency and Computation: Practice and Experience*, *35*(27). https://doi.org/10.1002/cpe.7854

Laporte, G., Nobert Yves, & Desrochers, M. (1985). OPTIMAL ROUTING UNDER CAPACITY AND DISTANCE RESTRICTIONS. *Operations Research*, *33*(5), 1050–1073. https://doi.org/10.1287/OPRE.33.5.1050

Le Colleter, T., Dumez, D., Lehuédé, F., & Péton, O. (2023). Small and large neighborhood search for the park-and-loop routing problem with parking selection. *European Journal of Operational Research*, *308*(3), 1233–1248. https://doi.org/10.1016/J.EJOR.2023.01.007

Lebeau, P., De Cauwer, C., Van Mierlo, J., Macharis, C., Verbeke, W., & Coosemans, T. (2015). Conventional, Hybrid, or Electric Vehicles: Which Technology for an Urban Distribution Centre? *The Scientific World Journal*, *2015*, 1–11. https://doi.org/10.1155/2015/302867

Liang, Y.-J., & Luo, Z.-X. (2022). A Survey of Truck–Drone Routing Problem: Literature Review and Research Prospects. *Journal of the Operations Research Society of China*, *10*(2), 343–377. https://doi.org/10.1007/s40305-021-00383-4

Liu, H., Yin, W., Yuan, X., & Niu, M. (2018). Reserving Charging Decision-Making Model and Route Plan for Electric Vehicles Considering Information of Traffic and Charging Station. *Sustainability*, *10*(5), 1324. https://doi.org/10.3390/su10051324

Lu, J., Chen, Y., Hao, J. K., & He, R. (2020). The Time-dependent Electric Vehicle Routing Problem: Model and solution. *Expert Systems with Applications*, *161*, 113593. https://doi.org/10.1016/J.ESWA.2020.113593

Lugo-Laguna, D., Arcos-Vargas, A., & Nuñez-Hernandez, F. (2021). A European Assessment of the Solar Energy Cost: Key Factors and Optimal Technology. *Sustainability*, *13*(6), 3238. https://doi.org/10.3390/su13063238

Macrina, G., Di Puglia Pugliese, L., Guerriero, F., & Laporte, G. (2019). The green mixed fleet vehicle routing problem with partial battery recharging and time windows. *Computers & Operations Research*, *101*, 183–199. https://doi.org/10.1016/J.COR.2018.07.012

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

Meng, M., & Ma, Y. (2020). Route Optimization of Electric Vehicle considering Soft Time Windows and Two Ways of Power Replenishment. *Advances in Operations Research*, *2020*, 1–10. https://doi.org/10.1155/2020/5612872

Montoya, A., Guéret, C., Mendoza, J. E., & Villegas, J. G. (2017). The electric vehicle routing problem with nonlinear charging function. *Transportation Research Part B: Methodological*, *103*, 87–110. https://doi.org/10.1016/J.TRB.2017.02.004

Montoya, A., Guéret, C., Mendoza, J. E., Villegas, J., Montoya, «alejandro, Villegas», J. G., & Villegas, J. G. (2015). *The electric vehicle routing problem with partial charging and nonlinear charging function*. https://hal.science/hal-01245232v2

Munari, P., Dollevoet, T., & Spliet, R. (2016). *A generalized formulation for vehicle routing problems*. http://arxiv.org/abs/1606.01935

Ngatchou, P., Zarei, A., & El-Sharkawi, A. (2005). Pareto Multi Objective Optimization. *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*, 84–91. https://doi.org/10.1109/ISAP.2005.1599245

Qin, H., Su, X., Ren, T., & Luo, Z. (2021). A review on the electric vehicle routing problems: Variants and algorithms. *Frontiers of Engineering Management*, *8*(3), 370–389. https://doi.org/10.1007/s42524-021-0157-1

Reed, S., Campbell, A. M., & Thomas, B. W. (2022). The Value of Autonomous Vehicles for Last-Mile Deliveries in Urban Environments. *Management Science*, *68*(1), 280–299. https://doi.org/10.1287/mnsc.2020.3917

Sassi, O., Cherif, W. R., & Oulamara, A. (2014). *Vehicle routing problem with mixed fleet of convential and heterogenous electric vehicles and time dependent charging costs*.

Sassi, O., Cherif-Khettaf, W. R., & Oulamara, A. (2015). Iterated tabu search for the mix fleet vehicle routing problem with heterogenous electric vehicles. *Advances in Intelligent Systems and Computing*, *359*, 57–68. https://doi.org/10.1007/978-3-319-18161-5_6

Schneider, M., Stenger, A., & Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science*, *48*(4), 500–520. https://doi.org/10.1287/TRSC.2013.0490

Schonfeld, P., & Choi, Y. (2017). *Optimization of Multi-package Drone Deliveries Considering Battery Capacity*. https://www.researchgate.net/publication/347485538

Schulte, F., Lalla-Ruiz, E., González-Ramírez, R. G., & Voß, S. (2017). Reducing port-related empty truck emissions: A mathematical approach for truck appointments with collaboration. *Transportation Research Part E: Logistics and Transportation Review*, *105*, 195–212. https://doi.org/10.1016/J.TRE.2017.03.008

Shao, S., Guan, W., Ran, B., He, Z., & Bi, J. (2017). Electric Vehicle Routing Problem with Charging Time and Variable Travel Time. *Mathematical Problems in Engineering*, *2017*, 1–13. https://doi.org/10.1155/2017/5098183

Shazly, S. M. El. (1996). Estimation of hourly and daily global solar radiation at clear days using an approach based on modified version of gaussian distribution. *Advances in Atmospheric Sciences*, *13*(3), 349–358. https://doi.org/10.1007/BF02656852

Solar Photovoltaic Panel Sizes: A Complete Guide. (2023). Skills Training Group. https://www.skillstg.co.uk/blog/solar-photovoltaic-panel-sizes/#:~:text=A%20monocrystalline%20residential%20solar%20panel,of%20around%20340%2D400%20watts.

Soly.nl. (2024). https://soly.nl/zonnepanelen/wattpiek/430-wp/#:~:text=Het%20formaat%20van%20430%20Wp%20zonnepanelen%20is%20176,2%C3%97113,4%C3%973,0

Sonneberg, M.-O., Leyerer, M., Kleinschmidt, A., Knigge, F., & Breitner, M. H. (2019). *Autonomous Unmanned Ground Vehicles for Urban Logistics: Optimization of Last Mile Delivery Operations*. https://doi.org/10.24251/HICSS.2019.186

Srinivas, S., Ramachandiran, S., & Rajendran, S. (2022). Autonomous robot-driven deliveries: A review of recent developments and future directions. *Transportation Research Part E: Logistics and Transportation Review*, *165*, 102834. https://doi.org/10.1016/J.TRE.2022.102834

TerBerg Electric Vehicles. (2024). Https://Www.Terbergspecialvehicles.Com/En/Development/Electric/.

Theurich, F., Fischer, A., & Scheithauer, G. (2021). A branch-and-bound approach for a Vehicle Routing Problem with Customer Costs. *EURO Journal on Computational Optimization*, *9*, 100003. https://doi.org/10.1016/J.EJCO.2020.100003

Toth, P., & Vigo, D. (2002). 1. An Overview of Vehicle Routing Problems. In *The Vehicle Routing Problem* (pp. 1–26). Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898718515.ch1

Ulmer, M. W., & Streng, S. (2019). Same-Day delivery with pickup stations and autonomous vehicles. *Computers & Operations Research*, *108*, 1–19. https://doi.org/10.1016/J.COR.2019.03.017

Verma, A. (2018). Electric vehicle routing problem with time windows, recharging stations and battery swapping stations. *EURO Journal on Transportation and Logistics*, *7*(4), 415–451. https://doi.org/10.1007/S13676-018-0136-9

Vidović, M., Popović, D., Ratković, B., & Radivojević, G. (2017). Generalized mixed integer and VNS heuristic approach to solving the multisize containers drayage problem. *International Transactions in Operational Research*, *24*(3), 583–614. https://doi.org/10.1111/itor.12264

Wang, X., & Regan, A. C. (2002). Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B: Methodological*, *36*(2), 97–112. https://doi.org/10.1016/S0965-8564(00)00037-9

Wang, Y., & Han, Z. (2021). Ant colony optimization for traveling salesman problem based on parameters optimization. *Applied Soft Computing*, *107*, 107439. https://doi.org/10.1016/J.ASOC.2021.107439

Windras Mara, S. T., Norcahyo, R., Jodiawan, P., Lusiantoro, L., & Rifai, A. P. (2022). A survey of adaptive large neighborhood search algorithms and applications. *Computers & Operations Research*, *146*, 105903. https://doi.org/10.1016/J.COR.2022.105903

Woldring, B. (2024, July 19). *Wat is de stroomprijs per kWh?* Www.Gaslicht.Com. https://www.gaslicht.com/energienota/stroomprijs?t_campaignid=710427050&t_adgroupid=1254544082947275&t_feeditemid=&t_extensionid={extensionid}&t_targetid=kwd-78409366940191:loc-2485&t_loc_interest_ms=&t_loc_physical_ms=151804&t_matchtype=p&t_network=o&t_device=c&t_devicemodel=&t_gclid=&t_creative=&t_keyword=kwh%20prijs&t_placement=&t_target=&t_random=&t_adposition=&t_issearch=true&msclkid=3edcb772f67e1222814bd1d226286bc4&utm_source=bing&utm_medium=cpc&utm_campaign=Gaslicht.com%20*Search*%20%5BpMAX-Scale-Down%5D%20%5BMT%3AWG%2BEX%5D&utm_term=kwh%20prijs&utm_content=kWh%20stroomprijs

Xue, Z., Zhang, C., Lin, W. H., Miao, L., & Yang, P. (2014). A tabu search heuristic for the local container drayage problem under a new operation mode. *Transportation Research Part E: Logistics and Transportation Review*, *62*, 136–150. https://doi.org/10.1016/J.TRE.2013.12.007

Yang, J., & Sun, H. (2015). Battery swap station location-routing problem with capacitated electric vehicles. *Computers & Operations Research*, *55*, 217–232. https://doi.org/10.1016/J.COR.2014.07.003

Zhang, R., Guo, J., & Wang, J. (2022). A Time-Dependent Electric Vehicle Routing Problem With Congestion Tolls. *IEEE Transactions on Engineering Management*, *69*(4), 861–873. https://doi.org/10.1109/TEM.2019.2959701

Zhang, R., Yun, W. Y., & Kopfer, H. (2015). Multi-size container transportation by truck: modeling and optimization. *Flexible Services and Manufacturing Journal*, *27*(2–3), 403–430. https://doi.org/10.1007/s10696-013-9184-5

Appendix A. Experimental results time windows

Table 5-1. Extended Experimental Results Time Windows SAVED use case 1/2

	Start											Middle										
	No Solar Panels					Solar Panels					Dif	No Solar Panels					Solar Panels					Dif
ID	**Min**	**Obj**	**Max**	**PSE**	**t(s)**	**Min**	**Obj**	**Max**	**PSE**	**t(s)**		**Min**	**Obj**	**Max**	**PSE**	**t(s)**	**Min**	**Obj**	**Max**	**PSE**	**t(s)**	
S0	7.90	9.65	11.40	0.33	0.76	7.90	8.27	10.44	0.38	1.44	-14.3%	7.25	8.64	10.27	0.34	0.72	5.82	7.13	8.09	0.37	1.41	-17.5%
S1	12.44	15.40	17.49	0.39	0.75	12.44	14.09	15.90	0.39	1.29	-8.5%	12.44	18.23	23.07	0.35	0.73	11.33	16.66	21.83	0.36	1.29	-8.6%
S2	19.21	22.99	26.41	0.45	0.79	19.21	13.52	16.33	0.72	1.54	-41.2%	18.19	22.54	24.48	0.43	0.74	9.30	12.80	15.66	0.73	1.61	-43.2%
S3	29.29	32.05	35.23	0.43	0.76	29.29	20.92	25.02	0.68	1.18	-34.7%	24.41	32.53	43.22	0.42	0.74	15.90	21.07	27.98	0.67	1.41	-35.2%
S4	35.20	40.38	50.61	0.39	0.77	35.20	27.73	41.29	0.62	1.30	-31.3%	35.18	43.90	51.20	0.32	0.73	22.00	30.45	40.35	0.57	1.51	-30.6%
S5	23.56	32.96	42.57	0.56	0.76	23.56	18.45	24.56	0.82	1.57	-44.0%	27.55	33.67	46.99	0.53	0.74	10.85	18.85	29.20	0.80	1.55	-44.0%
S6	45.07	52.52	65.61	0.34	0.76	45.07	36.96	56.91	0.56	1.76	-29.6%	46.16	54.57	66.26	0.28	0.75	27.86	41.90	56.77	0.47	1.76	-23.2%
S7	37.65	44.95	51.81	0.45	0.74	37.65	29.76	39.33	0.68	1.58	-33.8%	43.16	50.60	61.89	0.35	0.71	30.81	38.29	51.05	0.52	1.61	-24.3%
S8	40.52	61.46	72.89	0.31	0.71	40.52	49.48	65.23	0.45	1.51	-19.5%	55.79	62.92	73.84	0.26	0.64	44.05	58.04	72.95	0.33	0.92	-7.8%
S9	68.96	81.73	96.46	0.18	0.61	68.96	75.37	85.62	0.23	0.91	-7.8%	72.38	84.56	104.24	0.13	0.50	72.38	78.47	96.29	0.21	0.50	-7.2%
AVG	**31.98**	**39.41**	**47.05**	**0.38**	**0.74**	**31.98**	**29.46**	**38.06**	**0.55**	**1.41**	**-26.5%**	**34.25**	**41.22**	**50.55**	**0.34**	**0.70**	**25.03**	**32.37**	**42.02**	**0.50**	**1.36**	**-24.2%**

Table 5-2. Extended Experimental Results Time Windows SAVED use case 2/2

	End											Uniform										
	No Solar Panels					Solar Panels					Dif	No Solar Panels					Solar Panels					Dif
ID	**Min**	**Obj**	**Max**	**PSE**	**t(s)**	**Min**	**Obj**	**Max**	**PSE**	**t(s)**		**Min**	**Obj**	**Max**	**PSE**	**t(s)**	**Min**	**Obj**	**Max**	**PSE**	**t(s)**	
S0	6.81	9.02	11.76	0.37	0.73	5.41	7.63	9.80	0.39	1.32	-15.5%	8.57	9.20	9.82	0.35	0.76	6.98	7.66	8.49	0.39	1.32	-16.7%
S1	13.22	16.46	19.33	0.36	0.75	12.25	14.99	17.71	0.38	1.36	-8.9%	13.30	17.92	25.51	0.36	0.74	11.63	15.57	19.10	0.38	1.44	-13.1%
S2	20.47	23.69	26.94	0.46	0.78	11.92	14.56	19.23	0.72	1.28	-38.5%	18.66	23.74	28.43	0.46	0.75	11.42	13.75	16.35	0.73	1.58	-42.1%
S3	29.13	35.61	51.72	0.37	0.74	18.18	23.55	37.74	0.64	1.65	-33.9%	26.98	32.35	36.20	0.41	0.75	17.07	20.41	22.91	0.68	1.54	-36.9%
S4	37.58	44.61	57.99	0.30	0.74	23.86	29.73	37.00	0.58	1.48	-33.4%	33.56	41.40	48.21	0.35	0.78	21.48	27.34	32.87	0.61	1.65	-34.0%
S5	26.16	38.43	52.26	0.43	0.74	11.10	18.43	24.80	0.80	1.56	-52.0%	25.06	34.64	40.63	0.51	0.76	10.24	16.62	22.70	0.85	1.77	-52.0%
S6	52.29	61.20	67.54	0.22	0.75	28.99	43.66	54.86	0.46	1.58	-28.7%	40.63	54.80	66.66	0.30	0.76	27.97	38.53	52.73	0.54	1.67	-29.7%
S7	47.13	60.17	77.19	0.25	0.72	26.64	43.99	63.69	0.48	1.65	-26.9%	38.04	49.74	61.07	0.39	0.75	25.53	33.94	50.04	0.62	1.44	-31.8%
S8	61.32	70.52	78.41	0.19	0.66	44.67	60.63	71.40	0.31	1.10	-14.0%	50.89	67.73	76.50	0.25	0.72	34.27	52.50	65.35	0.43	1.44	-22.5%
S9	71.22	82.13	93.37	0.15	0.59	62.35	76.84	91.95	0.20	0.84	-6.4%	79.72	87.38	91.71	0.13	0.67	76.91	81.80	86.43	0.17	0.96	-6.4%
AVG	**36.53**	**44.19**	**53.65**	**0.31**	**0.72**	**24.54**	**33.40**	**42.82**	**0.50**	**1.38**	**-25.8%**	**33.54**	**41.89**	**48.48**	**0.35**	**0.74**	**24.35**	**30.81**	**37.70**	**0.54**	**1.48**	**-28.5%**

Table 5-3. Extended Experimental Results Time Windows Campus use case 1/2

	Start											**Middle**										
	No Solar Panels					**Solar Panels**					**Dif**	**No Solar Panels**					**Solar Panels**					**Dif**
ID	**Min**	**Obj**	**Max**	**PSE**	**t(s)**	**Min**	**Obj**	**Max**	**PSE**	**t(s)**		**Min**	**Obj**	**Max**	**PSE**	**t(s)**	**Min**	**Obj**	**Max**	**PSE**	**t(s)**	
C0	0.53	0.72	0.85	0.60	1.46	0.34	0.48	0.54	0.88	2.26	-33.9%	0.57	0.75	0.98	0.65	1.88	0.41	0.54	0.63	0.83	2.14	-29.1%
C1	0.55	0.95	1.08	0.64	2.18	0.47	0.69	0.90	0.89	2.38	-27.7%	0.93	1.10	1.41	0.65	1.07	0.68	0.83	1.14	0.80	2.01	-24.2%
C2	0.89	1.14	1.39	0.64	1.96	0.74	0.83	1.06	0.89	3.63	-27.0%	1.09	1.16	1.23	0.70	0.94	0.86	0.92	0.99	0.84	1.35	-20.7%
C3	1.00	1.41	1.79	0.59	1.70	0.81	1.09	1.41	0.81	2.44	-22.9%	1.31	1.39	1.52	0.69	1.19	0.87	1.07	1.21	0.83	1.84	-22.8%
C4	1.18	1.52	2.05	0.62	1.54	1.00	1.16	1.28	0.84	2.19	-23.3%	1.71	1.82	1.93	0.62	1.02	1.48	1.53	1.58	0.71	0.93	-15.8%
C5	1.46	1.88	2.27	0.58	1.48	1.31	1.40	1.79	0.74	1.60	-25.4%	1.87	2.04	2.21	0.62	1.05	1.62	1.73	1.88	0.70	1.12	-15.1%
C6	2.41	2.93	3.67	0.66	3.54	2.10	2.30	3.07	0.82	2.18	-21.7%	3.18	3.53	3.86	0.66	1.51	2.71	3.02	3.34	0.74	1.47	-14.4%
C7	2.47	3.17	3.82	0.64	2.06	2.08	2.37	3.45	0.84	2.16	-25.4%	3.26	3.51	3.85	0.68	1.33	2.83	2.91	3.21	0.76	1.46	-17.2%
C8	2.55	3.07	4.16	0.67	2.26	2.22	2.31	3.00	0.86	2.48	-24.7%	3.37	3.72	3.92	0.66	1.84	2.91	3.08	3.45	0.75	1.39	-17.2%
C9	4.53	6.02	6.84	0.65	2.89	3.95	4.81	5.90	0.81	1.77	-20.2%	6.55	6.75	7.17	0.68	1.55	5.67	5.84	6.20	0.76	1.81	-13.4%
AVG	**1.76**	**2.28**	**2.79**	**0.63**	**2.11**	**1.50**	**1.74**	**2.24**	**0.84**	**2.31**	**-25.2%**	**2.38**	**2.58**	**2.81**	**0.66**	**1.34**	**2.00**	**2.15**	**2.36**	**0.77**	**1.55**	**-19.0%**

Table 5-4. Extended Experimental Results Time Windows Campus use case 2/2

	End											**Uniform**										
	No Solar Panels					**Solar Panels**					**Dif**	**No Solar Panels**					**Solar Panels**					**Dif**
ID	**Min**	**Obj**	**Max**	**PSE**	**t(s)**	**Min**	**Obj**	**Max**	**PSE**	**t(s)**		**Min**	**Obj**	**Max**	**PSE**	**t(s)**	**Min**	**Obj**	**Max**	**PSE**	**t(s)**	
S0	1.01	1.10	1.27	0.45	1.10	0.72	0.84	1.00	0.59	1.77	-23.8%	0.68	0.84	0.95	0.59	2.39	0.41	0.58	0.67	0.82	3.87	-31.0%
S1	1.61	1.63	1.65	0.47	1.35	1.30	1.39	1.49	0.52	1.18	-14.5%	1.02	1.23	1.47	0.62	2.06	0.77	0.89	1.01	0.82	3.53	-27.4%
S2	1.87	1.87	1.87	0.52	0.98	1.55	1.55	1.55	0.59	1.15	-17.2%	1.04	1.34	1.68	0.65	1.82	0.85	0.99	1.12	0.80	2.88	-26.1%
S3	2.04	2.04	2.04	0.49	1.23	1.48	1.48	1.48	0.68	1.67	-27.7%	1.33	1.51	1.91	0.67	2.57	0.91	1.09	1.52	0.79	2.77	-27.6%
S4	2.09	2.21	2.33	0.52	1.49	1.86	1.95	1.99	0.54	1.60	-11.8%	1.61	1.80	1.94	0.64	2.02	1.19	1.35	1.49	0.76	2.27	-24.7%
S5	2.17	2.31	2.40	0.56	1.28	1.82	1.96	2.11	0.62	1.73	-15.3%	1.72	1.92	2.29	0.67	1.53	1.34	1.46	1.61	0.78	2.16	-24.0%
S6	4.71	4.95	5.29	0.45	1.74	4.25	4.33	4.76	0.45	1.60	-12.6%	3.30	3.65	4.03	0.65	2.70	2.62	2.80	3.04	0.76	2.56	-23.2%
S7	5.04	5.17	5.29	0.45	1.25	4.29	4.40	4.52	0.52	1.64	-14.9%	3.44	3.90	4.21	0.64	2.31	2.62	3.04	3.37	0.73	2.53	-22.0%
S8	5.12	5.25	5.38	0.44	1.23	4.48	4.54	4.61	0.49	1.50	-13.5%	3.65	4.09	4.42	0.64	2.38	2.85	3.19	3.51	0.73	2.48	-21.9%
S9	9.74	10.52	11.37	0.44	1.28	8.76	8.76	8.76	0.44	1.32	-16.7%	6.48	7.03	7.58	0.67	1.45	5.18	5.61	6.06	0.75	1.71	-20.2%
AVG	**3.54**	**3.71**	**3.89**	**0.48**	**1.29**	**3.05**	**3.12**	**3.23**	**0.54**	**1.52**	**-16.8%**	**2.43**	**2.73**	**3.05**	**0.64**	**2.12**	**1.87**	**2.10**	**2.34**	**0.77**	**2.68**	**-24.8%**