Erasmus University Rotterdam
Erasmus School of Economics
Master Thesis Analytics and Operations Research in Logistics

A column generation approach for the electric truck routing problem with capacitated charging stations

Michaël Alexander Litjens (660334)

Supervisor:	Dr. Wilco van den Heuvel
Second assessor:	Dr. Twan Dollevoet
Supervisor TNO:	Ruben Fransen
Date final version:	July 17, 2024

This thesis was conducted at TNO at the department of Sustainable Transport & Logistics.

Abstract

This research addresses the single depot joint routing and charging scheduling problem for battery-electric trucks (BETs), accounting for range constraints, capacitated charging stations, partial recharging, and varying customer demand. Two column generation heuristics were developed: a truncated price-and-branch algorithm and a truncated column generation algorithm. To solve the pricing problem, a network structure was formulated in which the charging actions were integrated into the arcs and the state of charge values in the nodes.

Real world data was used, in which containers were transported from a depot to the customers. The largest used data instance contained 72 customer requests and 24 trucks. Both algorithms were able to generate feasible solutions within one hour of computation time and with an optimality gap smaller than 0.7% for this instance. These findings suggest that column generation is a promising approach for solving the joint routing and charging scheduling problem for BETs.

Contents

1 Introduction

The transport sector plays a crucial role in our day-to-day lives, enabling the interaction between people, businesses and resources and it is a driving factor behind our economic growth. However, the transport sector also contributes negatively to air, land and water quality, the ecosystem and human health. Research has shown that the transport sector in Europe is responsible for around a quarter of the greenhouse gas emissions in Europe (Ali, Socci, Pretaroli, & Severini, 2018). Road freight transport, including heavy-duty trucks, represents approximately 40% of these emissions and thus electrification of this fleet can lead to a significant decrease in greenhouse gas emissions (Kluschke, Gnann, Plötz, & Wietschel, 2019). Although the use of heavy-duty electric trucks can reduce the emission of greenhouse gases drastically, the use of these vehicles remains limited whereas the use of electric buses and passenger cars increases significantly.

There are several reasons for this disparity, among one is that electric trucks consume much more energy per kilometer compared to electric passenger cars and thus require larger batteries (Weiss, Cloos, & Helmers, 2020). Furthermore, according to Center (2020), trucks drive significantly further compared to passenger busses and thus require, also in combination with larger batteries, either longer recharging, more frequent recharging or more advanced charging infrastructure. Furthermore, transit buses are frequently charged at privately owned charging stations, electric trucks require due to their larger travel distance, publicly available charging stations, which remain limited at the moment (Bluekens, 2024). Publicly available charging stations require the need for a comprehensive charging planning which includes the modelling of the availability of chargers at any given moment in time.

According to the European Automobile Manufacturers Association, the amount of electric trucks in Europe needs to increase by a factor of 87 in order to achieve the EU's CO2 reduction targets (Zsófia, 2021). Although this will lead to a decrease in CO2 emissions, many practical aspects of the implementation of these electric trucks remain unknown, indicating the need for more research on the use of (battery) electric trucks (BETs). In April 2024, the European Parliament adopted even stricter CO2 emissions targets for trucks, making research on the practical implementation of BETs even more evident (Parliament, 2024).

Electric vehicles (EVs) require the need for charging management, whereas refuelling management for conventional vehicles is not necessary and thus the conventional vehicle routing problems (VRPs) require alterations. Many different variants of the VRP are studied, but mostly focused on fossil-fueled vehicles. The cruising range of EVs cannot compete with the cruising ranges of fossil-fueled vehicles and charging stations are not as widespread as gas stations. Furthermore, charging an electric truck takes much longer than refuelling a fossil-fueled truck. These problems lead to a new category of VRPs, namely the electric vehicle routing problem (EVRP). A common factor in research on EVRPs is that a maximum capacity at the charging stations is neglected. In Froger, Jabali, Mendoza, and Laporte (2022), however, a maximum capacity at the charging stations is considered and a new formulation is introduced. The problem is solved by using an iterated local search algorithm in combination with a branch-and-cut algorithm to build solutions satisfying the capacity constraints. The use of electric buses also requires the consideration of charging planning, which is considered in de Vos, van Lieshout, and Dollevoet (2022). The problem is solved by using a column generation algorithm in combination with a price-and-branch and truncated column generation algorithm. Battery electric trucks exhibit similarities to electric passenger cars and electric buses, but also display distinctive characteristics. Contrary to electric buses, battery electric trucks lack specific timetables and thus their schedules vary daily. However, similar to electric buses, battery electric trucks need a comprehensive charging planning. Operating heavy-duty battery electric trucks also involves specific considerations, such as tardiness and cargo capacity which add new constraints to the problem (Bragin, Ye, & Yu, 2024).

In this research, the EVRP is extended to the scheduling of battery-electric trucks while considering a maximum capacity at the charging stations. Data from Van Berkel Logistics is used which includes the demand of customers for container terminals. This dataset contains trips from a container terminal to customers over a period of three years including origin, destination and the delivery deadline.

In this research, a novel problem formulation is introduced, and the problem is solved by a column generation algorithm. The algorithm aims to find solutions to the problem within a reasonable time ($\leq 1h$), with a reasonable optimality gap ($\leq 5\%$), and with a practical implementation.

This research consists of the following sections: In Section 2, the characteristics of the problem are explained. In Section 3, the corresponding literature is discussed and in Section 4 and 5, the methodology of this research, including a mathematical formulation of the problem, is explained. The results from applying the methodology to the given dataset are presented in Section 7. The paper is concluded in Section 8 by summarizing the results, drawing a conclusion and suggestions for further research.

2 Problem description

In this section, an overview of the considered problem is given. The problem that is considered in this research is a single depot joint routing and charging scheduling of battery-electric trucks that includes range constraints, capacitated charging stations, partial recharging and varying customer demands.

Battery-electric trucks have different characteristics compared to conventional diesel-powered trucks, such as different driving ranges and a longer refuelling (recharging) time. Therefore, it is important that future planning algorithms combine transport and charging planning, taking into account the available charging capacity at charging stations. The aim of this research is to effectively schedule the transportation planning of battery electric trucks while considering the capacity at charging stations. In this research, container transport is considered, and for simplicity it is assumed that a truck can carry at most one container. After delivering a container to a customer, the truck is not able to visit another customer but always has to return back to the depot. Transporting containers from the customer back to the depot is not considered. The input is the demand of the customers for a specific container terminal. This demand consists of the amount of containers and the desired delivery times.

A trip is defined as a route from the depot to a customer and back to the depot while possibly visiting intermediate (public) charging stations, a route for a truck is built up by one or more of these trips. The location of the charging stations can be determined from the data from Bluekens (2024). Assumptions on the amount of chargers at charging stations may be used, since the current charging capacity at charging stations is now limited to a maximum of 2 chargers. The capacity of the chargers at the charging stations is known and can be retrieved from Bluekens (2024). It is furthermore assumed that all the trucks of Van Berkel Logistics are electric and an assumption on the amount of available trucks needs to be made, since currently only one electric truck is used by the company. TNO has intensively monitored the only BET of Van Berkel Logistics, which resulted in a heatmap of the vehicle location depicted in Figure 2.1. These locations correspond to some of the charging stations in Bluekens (2024) and thus these charging stations can potentially be used for this research.

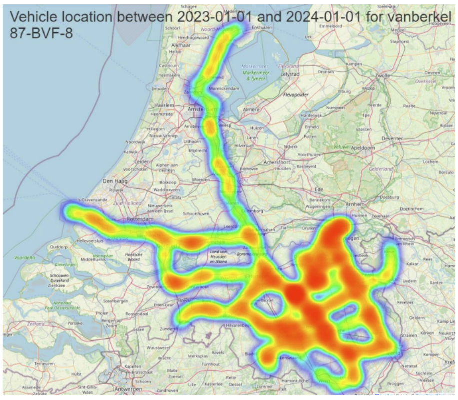

Figure 2.1: Heatmap of the vehicle location, obtained by TNO

The battery capacity of the trucks is 540 kWh and the average discharge rate per kilometre is 1.28 kWh/km. The discharge rate is assumed to be constant as is also done in previous research (Cui, Gao, Yu, Ma, & Najafi, 2023). The data on the BETs is made available to TNO by Van Berkel Logistics.

Additionally, assumptions on the depot's available charging infrastructure must be made. Obtained from interviews with Van Berkel Logistics by TNO, it is known that by the end of this year 6 Heliox chargers with each 90 kW capacity are present at the depot. In order to obtain a feasible schedule more chargers with higher capacities may be considered. Net congestion will not be considered in this research and thus there will be no limitation on the available energy supply at charging stations.

The battery capacity of a battery-electric truck will be modelled through its state of charge (SoC); this value needs to be strictly positive during a feasible trip. At the start of the planning horizon, all the BETs are assumed to be fully charged. Charging actions can be scheduled at charging stations with available capacity and partial charging is considered. It is assumed that if a truck starts travelling on a certain road segment, it needs to arrive at the end node of the road segment after a certain period. Since the modelling of continuous time blows up the number of decision variables, time blocks, as proposed in de Vos et al. (2022) will be used.

The objective function of our optimization problem focuses on minimizing the total transportation costs, including the driver's wage, charging costs and the costs associated with a positive tardiness.

This problem definition leads to the following research questions:

1. Which algorithm can be used to solve the joint routing and charging scheduling of battery electric trucks within reasonable time ($\leq 1h$) and with a reasonable optimality gap ($\leq 5\%$)?

The aim of this research is also to answer the following subquestions. These subquestions are relevant for practical implementations.

1. How does the battery capacity influence the transportation and charging planning?

2. How does the available amount of charging stations influence the transportation and charging planning?

3 Literature Review

In this section, the relevant literature from previous research on (electric) vehicle routing problems will be discussed. Firstly, the theory on the electric vehicle routing problem (EVRP) will be analyzed, followed by the theory on VRPs with synchronization and the theory on EVRPs with charging management. Lastly, possible solution techniques such as Lagrangian relaxation, column generation and metaheuristics will be discussed.

3.1 Electric Vehicle Routing Problem (EVRP)

The EVRP is an extension of the VRP that considers electric vehicles instead of fossil-fueled vehicles. The aim of the EVRP is to find an optimal set of vehicle routes that minimizes a certain objective function. According to Kucukoglu, Dewil, and Cattrysse (2021), this objective function can consist of several components including total travel distance, total travel time, total recharging cost or recharging time and other operational costs. A route consists of a begin node, corresponding to the depot, an end node, corresponding to the same depot and intermediate customer nodes and charging nodes. In Kucukoglu et al. (2021), a basic set of assumptions for this problem is defined:

1. The begin and end node of a route correspond to the depot
2. Each customer node is visited exactly once by an electric vehicle
3. A charging station can be visited within a route for recharging
4. Charging stations have unlimited availability of chargers
5. The locations of the charging stations are known and thus the distances from any node to the charging stations are known
6. The battery level of an electric vehicle must be strictly greater than zero and smaller than or equal to the maximum battery capacity
7. A full recharging policy is used

This set of assumptions can be extended by adding vehicle capacity constraints, making it a CEVRP (Jia, Mei, & Zhang, 2021). Furthermore, time-related restrictions such as delivery windows can be added, corresponding to an EVRPTW (Raeesi & Zografos, 2020). As mentioned in Wen, Linde, Ropke, Mirchandani, and Larsen (2016), partial recharging can also be considered. Since the VRP is classified as an NP-hard problem, the EVRP will definitely be NP-hard as well (Kumar & Panneerselvam, 2012). For NP-hard problems, only small instances can be solved to optimality, and therefore, these problems rely on heuristics. The assumptions on the unlimited availability of chargers and the full recharging policy need to be changed to address the problem in this research.

3.2 VRP with synchronization

Vehicle routing problems with synchronization increase the complexity of a model. The interaction between vehicles to share resources is called synchronization (Lam, Desaulniers, & Stuckey, 2022). In El Hachemi, Gendreau, and Rousseau (2011) such a VRP with synchronization is researched: the joint routing and scheduling problem for trucks in the timber industry. In Jungwirth, Desaulniers, Frey, and Kolisch (2022), a vehicle routing problem with scheduling constraints is solved. An exact branch-price-and-cut is used to solve the problem. VRPs with synchronization are characterized by the fact that changes in a route affect other routes as well. This is in contrast with the conventional VRP where changes in one route do not affect changes in other routes in the solution. These interactions increase the complexity of a model. Since a limited capacity on the charging stations is assumed, the proposed problem in Section 2 contains synchronization. There are several variants of synchronization: task synchronization, operation synchronization, movement synchronization, load synchronization and resource synchronization (Labadie, Prins, & Yang, 2014). The problem in this research falls into the category of resource synchronization: only a limited amount of chargers are available for the BETs.

3.3 Charging management

Charging management has two components: choosing a charging policy and considering the charging station's capacity.

As defined in Kucukoglu et al. (2021), two charging policies can be defined: partial charging and full charging. If full charging is chosen, the battery is charged to 100% during every recharge stop. In case of partial charging, a new decision variable needs to be defined indicating the amount of energy charged during a recharge stop. In Keskin and Çatay (2016), it was found that allowing partial charging led to better solution qualities compared to a full charging policy. In this research, an Adaptive Large Neighborhood Search with Simulated Annealing was used to formulate solutions.

Furthermore, previous research neglected the capacity at charging stations which can lead to large delays according to Froger et al. (2022). A practical transportation planning algorithm incorporates this capacity. A new formulation is introduced which does not allow for violations of the capacity at the charging station with the assumption of privately owned charging stations. The research introduces a route generator in which the capacity at the charging stations is neglected and the problem is solved by an iterative local search.

Charging management is also considered in the scheduling of electric buses as described in de Vos et al. (2022). The difference with this research is that by scheduling trucks not a fixed set of scheduled trips is considered which increases the number of possible solutions.

The latest, and to our knowledge, the only research on the joint scheduling of routing and charging for electric (container) trucks with capacitated charging stations is done in Bragin et al. (2024). A novel mathematical formulation for container truck transport between a depot and port for battery electric vehicles is introduced which takes capacitated charging stations into account. The coupling constraints that link the different trucks are relaxed and the problem is solved by Surrogate Level-Based Lagrangian relaxation as introduced in Bragin and Tucker (2022). The mathematical formulation of the problem can be translated into our problem by considering trips between the depot and the customers instead of only considering trips between a depot and a port.

In Jia et al. (2021), a mathematical formulation is introduced that minimizes the total transportation distance. A bi-level ant colony optimization algorithm is used in which a capacitated vehicle routing problem is solved in the first subproblem, by relaxing the battery capacity constraint. In the second subproblem, the generated routes are used to solve a Fixed Route Vehicle Charging Problem (FRVCP), only allowing full recharging. The capacity of the charging stations is not considered in this research, but by considering the capacity and alternating the objective function, this formulation can be translated to our problem. The second subproblem then changes to a FRVCP with capacited charging stations. Charging management is also considered in Lam et al. (2022). An EVRP with time windows and capacitated recharging stations is solved by Dantzig-Wolfe decomposition in combination with Benders decomposition.

3.4 Solution techniques

In this subsection different solution techniques for the proposed problem will be discussed. Firstly, Lagrangian relaxation will be discussed, followed by a discussion of column generation and metaheuristics.

3.4.1 Lagrangian relaxation

Mixed-integer-programming problems (MIPs) with synchronization contain so-called coupling constraints and they are known for their combinatorial complexity. Lagrangian Relaxation has proven to be successful in resolving combinatorial issues (Bragin & Tucker, 2022). Lagrangian relaxation involves reallocating certain 'difficult' constraints, which complicate the MIP, into the objective function (Gaudioso, 2020). Lagrangian relaxation has been previously used in tackling the combinatorial complexity of EVRPs (Xia, He, Wang, Liu, & Zhang, 2024).

The optimal solution of the Lagrangian relaxation is a lower bound for the original problem. The process of finding the best lower bound is referred to as the Lagrangian dual (Gaudioso, 2020). The solution to the Lagrangian dual problem is smaller than or equal to the solution of the original problem (Gaudioso, 2020). In Bragin and Tucker (2022), Surrogate Level-Based Lagrangian relaxation is introduced and this solution technique is used for electric truck scheduling with capacitated charging stations in Bragin et al. (2024). Surrogate Level-Based Lagrangian relaxation (SLBLR) is compared to conventional Lagrangian relaxation, and it is found that SLBR performs better than conventional Lagrangian relaxation. In SLBR, different stepsizes are used in which an optimal solution to the subproblems is not needed in the optimization steps.

In Bragin et al. (2024), an instance corresponding to a demand of 3 at the port, a demand of 8 at the depot, and 5 available trucks results in 151,791 constraints, 28,312 binaries, 4,782,805 nonzeros and 65 time periods. CPLEX was not able to find a feasible solution to the problem within 1,000,000 seconds. The author states, however, the Surrogate Level-Based Lagrangian Relaxation method was able to find a solution with an optimality gap of less than 1% within 5 minutes. Larger instances were also analysed in which the battery characteristics and available charging stations were changed. For an instance corresponding to 49 trucks, feasible solutions could be obtained within 1,800 seconds. The computation time was, however, limited to 1,800 seconds and thus better solutions could be obtained by using a larger computation time.

Lagrangian relaxation was also successfully applied in Xia et al. (2024), in which an EVRP with capacitated charging stations was solved.

In order to apply Lagrangian relaxation, a node-based formulation that includes decision variables for charging needs to be used. A node-based formulation is likely to have more variables and constraints compared to an arc-based formulation.

3.4.2 Column generation

Column generation is used for solving linear programming problems including many variables (Lübbecke, 2010). Column generation is furthermore a common technique in order to solve VRPs in combination with a heuristic to ensure integral solutions. The problem is split into two problems; the Restricted Master Problem (RMP) and the Pricing Problem. The corresponding pricing problems for VRPs are often translated into a shortest path problem (Chabrier, 2006). In case of minimization, the pricing problem is solved and the route with the most negative reduced costs is added to the RMP. The RMP is solved and this process is repeated until no more variables with negative reduced costs can be found. The corresponding solution value gives a lower bound on the optimal value for the integer solution. For the EVRP, the shortest path problems changes in a resource-constrained shortest path problem.

A dynamic programming approach for a directed acyclic graph can be used to solve the (resource -constrained) shortest path problem. The resource constraints are modelled in the arcs; non-feasible arcs are not considered in the network.

A Multi Label Setting algorithm can also be used to solve the (resource-constrained) shortest path problem (Wu, Lin, Liu, & Jin, 2022). Resource constraints need not be modelled in the arcs, but are accounted for in the labels.

Furthermore, a branch-and-price-and-check algorithm can be used as is done in Lam and Hentenryck (2016) for a VRP with location congestion. The vehicle routing problem is solved by using Dantzig Wolfe decomposition and the scheduling problem is solved by using Benders decomposition. This problem can be translated to the problem solved in this research by modelling the charging stations as resources.

Truncated column generation can be used to obtain integer solutions. In every iteration of the algorithm, solution values above a certain threshold are set to one and the column generation algorithm is run again. This is done until all variables are integer (Ozturk, 2020). Another option is to use a branch-and-price algorithm to obtain feasible integer solutions (Barnhart, Johnson, Nemhauser, Savelsbergh, & Vance, 1998).

The initialization of the column generation algorithm can have a great impact on the speed of convergence of the algorithm. A feasible initial solution leads to meaningful dual values which can speed up the convergence of the algorithm. Furthermore, starting with a solution that is closer to the optimal value requires fewer iterations to convergence to the optimal solution. Thus, starting the algorithm with a feasible solution instead of an infeasible solution with very high costs can be computationally beneficial (Amor, Desrosiers, & Frangioni, 2009). The fact that in the first iterations irrelevant columns are generated due to poor information on the bounds and dual values is called the heading-in effect (Desaulniers, Desrosiers, & Solomon, 2006). A column generation algorithm is often initialized with artificial/infeasible solutions that are given very high costs, denoted by M, such that they are not chosen in an optimal solution. A high value of M can lead to an increase in the heading-in effect.

In de Vos et al. (2022), several instances were solved by using a truncated column generation algorithm. The instances are presented in Table 3.1. The number of trips represents the total number of trips that have to be performed in the dataset. Time blocks of 5 minutes were used. The outcome of the algorithm determines the amount of buses that will be used. With the use of a Truncated Column Generation algorithm, all the instances could be solved with a relatively small computation time (< 3h) and with a reasonable optimality gap ($< 1.319\%$).

Table 3.1: Summary of instances

Instance	Trips (#)
A	50
B	100
1	119
2	185
3	304
4	512
5	816

3.4.3 Metaheuristics

Metaheuristics are widely used in VRPs to tackle their combinatorial complexity and to generate solutions for practical applications (Elshaer & Awad, 2020).

In Froger et al. (2022), a metaheuristic to solve the EVRP with capacitated charging stations is introduced. A different mathematical formulation compared to our problem is used and multiple request locations can be visited on a route. Firstly, an iterated local search algorithm for the EVRP is used. In the variable neighbourhood descent algorithm, an FRVCP is used to generate (energy) feasible routes without considering the capacity at charging stations. Lastly, different subproblems are considered in order to account for the limited capacity at the charging stations. These subproblems lead to cuts that cut off infeasible selections of routes. Instead of using an iterated local search algorithm for the route generation, the proposed ant colony optimization algorithm in Jia et al. (2021) can also be used. The corresponding FRVCP can then be combined with the proposed subproblems in Froger et al. (2022) to account for the capacity at the charging stations.

Several other metaheuristics are used in the literature on EVRP, including genetic algorithms, tabu search, simulated annealing and large neighbourhood search (Bragin et al., 2024).

While metaheuristics have the ability to solve large instances in a relatively short computation time, optimality is not guaranteed. Furthermore, these algorithms lack the ability to improve the solution based on theory and these algorithm are often only applicable to a very specific case of the EVRP. (Bragin et al., 2024).

4 Mathematical formulation

In this section, the mathematical formulation of the problem is presented. To explain the mathematical formulation, some variables and parameters need to be defined, as will be done first. Secondly, the graph wich is used for the problem will be presented, followed by an analysis of the constraints and the objective function.

4.1 Variables and parameters

In this subsection, some variables and parameters of the mathematical formulation will be defined. Containers from a container terminal need to be transported to the customers by battery-electric trucks. After visiting a customer, the truck needs to return back to the depot; to possibly pick up a new container, for charging or when no more trips are scheduled for this truck. The mathematical formulation is based on the mathematical formulation in Bragin et al. (2024). Bragin et al. (2024) focuses on trips between a depot and a port, whereas now trips between a depot and customers are considered. Furthermore, in Bragin et al. (2024), a node-based formulation is used whereas in this research an arc-based formulation is presented. This leads to changes in the set of constraints. Furthermore, for every demand unit of a customer, a new customer location is created, such that each demand unit corresponds to a unique location. Each unit of demand is referred to as a request. A copy of each charging station is created to make a feasible graph as will be explained in Section 4.2.

Firstly, the used sets will be explained. These sets are given as parameters of the problem.

1. $\mathcal{K} = \{1, ..., K\}$: set of available battery electric trucks (BETs). For this set, the index k is used.
2. $\mathcal{N} = \{1, ..., N\}$: set of all the locations in the formulation. For this set, the index n is used.
3. $\mathcal{N}^{chrg} \subset \mathcal{N}$: set of locations that include a charging station, includes the copy of each charging station.
4. $\mathcal{N}^{chrg}_{org} \subset \mathcal{N}$: set of locations that include the original charging stations.
5. $\mathcal{N}^{chrg}_{cpy} \subset \mathcal{N}$: set of locations that include the copy of each charging station.
6. $N^{cstm} \subset \mathcal{N}$: set of locations that includes all the request locations.
7. $N^{art} \subset \mathcal{N}$: set of locations of the artificial source and sink node, as will be explained in Section 4.2.
8. $\mathcal{B} = \{1, ..., B\}$: set of time blocks. For this set, the index b is used.
9. $\mathcal{E} = \{1, ..., E\}$: set of arcs. For this set, the index e is used.
10. $\mathcal{S} = \{s \in \mathbb{R} \mid 0 < s \leq \text{CAP}\}$: set of state of charge values of the BETs. For this set, the index s is used.

Secondly, the parameters will be explained.

1. $T_{n,a}$: representing the travel time between location n and location a
2. $TB_{n,a}$: representing the travel time between location n and location a, expressed in integer amount of time blocks. $TB_{n,a} = \left\lceil \frac{T_{n,a}}{|b|} \right\rceil$.
3. $S_{n,a}$: representing the travel distance between location n and location a, with $n \neq a$.
4. C^{lbr}: representing the wage of a truck driver (per time unit)
5. C^{tard}_n: representing the costs for delivering request $n \in \mathcal{N}^{cstm}$ too late (per time unit)
6. C^{chrg}_n: representing the costs for charging at node $n \in \mathcal{N}^{charg}$ (per time unit)
7. $due_n \in \mathbb{Z}_+$: representing the due time of request $n \in \mathcal{N}^{cstm}$
8. Δdis^{chrg}: the average discharge rate of the BETs per kilometer
9. Δs^{chrg}_n: the average charge rate at node n per time block
10. CAP: representing the battery capacity of the BETs

Thirdly, some important indices will be explained

1. n^{depot}: representing the depot location
2. n^{chrg}_{copy}: representing the copy of a charging station node
3. $f(e)$: representing the head of arc e.
4. $s(e)$: representing the tail of arc e

Before the decision variables and some other important variables are defined, more context on the formulation of the graph is needed, which will be given in the following subsection.

4.2 Graph

In this subsection, the construction of the graph for the considered problem will be discussed. Firstly, an extra set $\mathcal{V}$ is introduced which defines the nodes in the graph. For this set, the index v is used. Secondly, the arcs in the graph will be discussed.

4.2.1 Nodes

A node is represented as a time-location combination containing the following characteristics.

1. The location
2. The time block

Let $N = \{n_1, n_2, \ldots, n_m\}$ be a set locations, and let $B = \{b_1, b_2, \ldots, b_k\}$ be a set of time blocks. Then the set of nodes $\mathcal{V}$, which is the Cartesian product of N and B, is defined as:

$$\mathcal{V} = N \times B = \{(n, b) \mid n \in N, b \in B\}$$

Here, $\mathcal{V}$ consists of all possible pairs (n, b) where n is an element of N and b is an element of B.

The location coordinates are used to calculate the distance between two nodes. A copy is made of every time-location combination of a charging station to make a feasible graph. An artificial source and sink node are introduced in the graph for flow conservation. Both source and sink node locations coincide with the same coordinates as the depot location, but still represent a different location in the set $\mathcal{N}$.

4.2.2 Arcs

Within the directed graph with a set of nodes $\mathcal{V}$ and a set of arcs $\mathcal{E}$. Let $TB_{u,v}$ be the travel time for arc (u, v), with $u, v \in \mathcal{V}$, and let B_u be the time block of the tail node u. The time block of the head node v, denoted as B_v, for each arc $(u, v) \in \mathcal{E}$ is defined by the following equation:

$$B_v = B_u + TB_{u,v} + 1 \tag{1}$$

where:

- B_u is the time block of the tail node of the arc.
- $TB_{u,v}$ is the travel time between nodes u and v expressed in integer amount of time blocks.

- B_v is the time block of the head node of the arc.

This formulation leads to the following arcs in the graph, for which it is important to note that an arc connects two nodes that both correspond to a time-location combination.

1. The artificial depot nodes, corresponding to the source and sink nodes, are connected to the depot nodes. The distance and travel time between these nodes are zero. These arcs are denoted by:

$$\{(u,v) \mid u=(n',b') \text{ with } n' \in \mathcal{N}^{art}, v=(n,b) \text{ with } n=n^{depot} \text{ and } b=b'\}$$

2. The depot nodes are connected to the original charging station nodes, the request nodes and the artificial depot nodes. These arcs are denoted by:

$$\{(u,v) \mid u=(n',b') \text{ with } n'=n^{depot}, v=(n,b) \text{ with } n \in \mathcal{N}^{cstm} \cup \mathcal{N}^{chrg}_{org} \cup \mathcal{N}^{art} \text{ and } b=b'+TB_{u,v}+1\}$$

3. The original charging station nodes are connected to the request nodes and subsequent charging station nodes. These arcs are denoted by:

$$\{(u,v) \mid u=(n',b') \text{ with } n' \in \mathcal{N}^{chrg}_{org}, v=(n,b) \text{ with } n \in \mathcal{N}^{chrg}_{org} \cup \mathcal{N}^{cstm} \text{ and } b=b'+TB_{u,v}+1\}$$

4. The request nodes are connected to the nodes corresponding to the copy of each charging station and the depot nodes. These arcs are denoted by:

$$\{(u,v) \mid u=(n',b') \text{ with } n' \in \mathcal{N}^{cstm}, v=(n,b) \text{ with } n \in \mathcal{N}^{chrg}_{cpy} \cup n^{depot} \text{ and } b=b'+TB_{u,v}+1\}$$

5. The nodes corresponding to the copy of each charging station are connected to the depot nodes and subsequent charging station nodes. These arcs are denoted by:

$$\{(u,v) \mid u=(n',b') \text{ with } n' \in \mathcal{N}^{chrg}_{cpy}, v=(n,b) \text{ with } n \in \mathcal{N}^{chrg}_{cpy} \cup n^{depot} \text{ and } b=b'+TB_{u,v}+1\}$$

If a truck is at a charging station and decides to 'travel' to the next time block at the same charging station location, it stays charging at the location for another time block.

For the arcs connecting the depot with an artificial sink location, there is no travel time, but the travel time is still set to 1 to simulate the ending of a route.

The graph should only consider feasible arcs to create a feasible schedule for the BETs. Therefore the following restrictions have been taken into account when formulating the arcs.

1. After visiting a customer, a truck should return to the depot before visiting another customer
2. A charging station can be visited on the way to a customer and on the way back to the depot
3. Departures are assumed to be at the end of a time block
4. Arrivals are assumed to be at the beginning of a time block
5. Unloading of a container is done within the period of one time block

6. The charging amount depends on the time block and is constant

7. Nodes can only be connected if Constraint (1) holds.

Due to the first restriction, a copy of each charging station is made to ensure that the truck always returns to the depot after visiting a request location. If no copies were used, an extra constraint had to be added to the mathematical formulation enforcing the return to the depot after a request location had been visited.

In the graph and following sections, the following sets are used: $\mathcal{V}^{art}$, $\mathcal{V}^{depot}$, $\mathcal{V}^{chrg}_{org}$, $\mathcal{V}^{chrg}_{cpy}$, $\mathcal{V}^{cstm}$ which are represented as:

$$\mathcal{V}^{set} = \{(n, b) \mid n \in \mathcal{N}^{set},\ b \in \mathcal{B}\}$$

The subscript *set* corresponds to the label of the subset.

An example of a graph is partly depicted in Figure 4.1. The figure corresponds to a time-location graph of a potential network. The vertical axis represents the location of a node, $n \in \mathcal{N}$, and the horizontal represents the corresponding time block of a node, $b \in \mathcal{B}$. The first level (level 0) corresponds to the artificial source and sink nodes, $v \in \mathcal{V}^{art}$, the second level corresponds to the depot nodes, $v \in \mathcal{V}^{depot}$, the third level corresponds to the original charging station nodes, $v \in \mathcal{V}^{chrg}_{org}$, the fourth level corresponds to the nodes associated with a copy of each charging station, $v \in \mathcal{V}^{chrg}_{cpy}$, and the fifth level corresponds to the nodes of a request, $v_1 \in \mathcal{V}^{cstm}$ and the sixth level to another request, $v_2 \neq v_1 \in \mathcal{V}^{cstm}$. Travel times are assumed to be 1 between the nodes and zero between the source/sink nodes and the depot. For simplicity it is assumed, that the intermediate distance between nodes on the horizontal axis is one time block and the nodes in the network also represent one time block; if a truck departs at the end of time block 4, it arrives at the beginning of time block 6, if the travel time is equal to 1 time block.

The previously mentioned five different arcs are all depicted in Figure 4.1.

From an artificial source node, node 2, it is only possible to travel to a depot node, node 9; this arc corresponds to the purple arc in the figure. From a depot node, node 9, it is possible to travel to an original charging station node, node 17, and customer request nodes, node 31 and 38; these arcs are green in the figure. From an original charging station node, node 17, it is possible to travel to a request node, node 39 (an arc to node 32 is also possible), and a subsequent node that corresponds to the same original charging station, node 18; these arcs correspond to the pink arcs in the figure. From a request node, node 38, it is possible to travel to a copy of a charging station node, node 25, and a depot node, node 11; these arcs correspond to the red arcs in the figure. From a copy of a charging station node, node 25, it is possible to travel to a subsequent charging station node corresponding to the same location, node 26, and a depot node, node 12; these arcs correspond to the blue arcs in the figure.

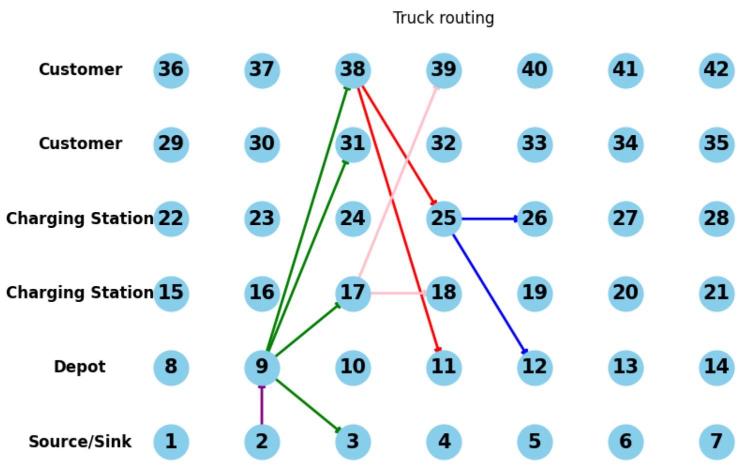

Figure 4.1: Example of a subset of the arcs in a graph

4.2.3 Potential route

A schematic representation of a possible route for a BET can be seen in Figure 4.2. For simplicity it is again assumed, that the intermediate distance on the horizontal axis between nodes equals one time block.

In the path, it can be seen that two customer requests are covered and that one charging station, node 24, is visited. If a charging node is visited, it means that the truck is charged for one time block. After visiting a customer, a truck always returns to the depot. Only for the (artificial) source/sink nodes, the inflow is not equal to the outflow. An extra time block is added when returning to the artificial sink node to simulate the ending of a route.

Arcs between the following levels are forbidden (for both ways).

- Level 0 (Source/Sink) and level 2, 3, 4 and 5
- Level 1 and level 3
- Level 4, 5 and level 2

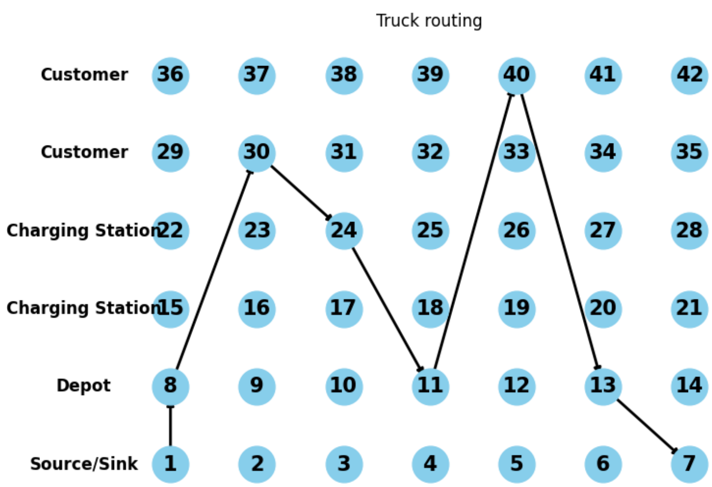

Figure 4.2: Time-location network of a feasible route of a truck

4.2.4 Decision variables

Using the information of the graph, the decision variables and some other important variables can be explained. The graph represents the time element of the problem, which is crucial in defining the nodes and some important variables of the mathematical formulation.

A decision variable is used which indicates whether a truck will travel along a certain arc in a route.

1. $x_k^e \in \{0,1\}$: representing whether truck k travels along arc $e = (u,v) \in \mathcal{E},\ u,v \in \mathcal{V}$.

Important variables to keep track of the charging and routing process are introduced below.

1. $a_{k,v} \in \mathbb{Z}_+$: arrival time of truck k to node v
2. $soc_{k,v} \in \mathcal{S}$: state of charge of the battery of truck k at node v
3. $tard_n \in \mathbb{Z}_+$: tardiness of delivering a container to request location $n \in \mathcal{N}^{cstm}$

4.3 Constraints

In this section, the constraints of the mathematical formulation of the problem will be discussed. It is important to note that both the set $\mathcal{N}$ and set $\mathcal{V}$ are used in the formulation, in which $\mathcal{N}$ corresponds to the set of locations and $\mathcal{V}$ to the set of nodes (time-location combinations) in the graph. If $\forall(e \in \mathcal{E} : s(e) = n)$ is used, it means the following subset is used:

$$\{e \in \mathcal{E} \mid s(e) = (n, b) \text{ with } b \in \mathcal{B},\ n = n\}$$

If $\forall(e \in \mathcal{E} : s(e) = v)$ is used, it means the following subset is used:

$$\{e \in \mathcal{E} \mid s(e) = (n, b) \text{ with } b = b.\ n = n\}$$

Furthermore, note that $f(e)$ is defined as the head node of arc e and $s(e)$ as the tail node of arc e.

4.3.1 Flow conservation

In Constraint (2), it is stated that the total inflow in a node must be equal to the outflow for each truck. This is done by setting the sum over the incoming arcs equal to the sum over the outgoing arcs in a node. This constraint holds for all the nodes in the graph, except for the artificial depot nodes, denoted by $\mathcal{V}^{art}$.

$$\sum_{u \in \mathcal{V}: e=(u,v) \in \mathcal{E}} x_k^e - \sum_{u \in \mathcal{V}: d=(v,u) \in \mathcal{E}} x_k^d = 0, \quad \forall(k \in \mathcal{K}, v \in \mathcal{V} \setminus \{\mathcal{V}^{art}\} \tag{2}$$

4.3.2 State of Charge

In Constraint (3), the state of charge of truck k at a specific node is modelled. If a truck travels along an arc that starts in a charging node, the state of charge in the head node is equal to the state of charge in the tail node minus the discharge along the arc, plus the charging amount at the charging station within a time block. Since Constraint (3) is a logical constraint, this constraint needs to be modelled as a big-M constraint. This is done in Constraint (4) and (5). The state of charge along arcs that do not start in a charging station is modelled in Constraint (6). Since this is also a logical constraint, this constraint is modelled as a big-M constraint in Constraint (7) and (8).

Charging is incorporated into the arcs, and to ensure the state of charge does not exceed battery capacity, Constraint (9) is added to the formulation. This constraint enforces that a time-location combination of a charging station, v, can only be visited if the state of charge allows for Δs_v^{chrg} kW extra charge. To allow for less charge, a constraint can be added that enforces the state of charge to be equal to the capacity of the BET if the extra charge of the charging station leads to a violation of the available capacity. This is not considered in this research.

Since Constraint (9) is also a logical constraint, this constraint is modelled as a Big-M constraint in (10).

Trucks are assumed to start their route fully charged, which is enforced by Constraint (11). The state of charge at the artificial depot nodes equals the battery capacity. This means that the trucks always start their route with a fully charged battery. This constraint can also be relaxed, enabling the state of charge value to be any value between the bounds.

$$x_k^e = 1 \implies soc_{k,f(e)} = soc_{k,v} - \Delta dis^{\text{chrg}} S_{v,f(e)} x_k^e + \Delta s_v^{\text{chrg}}, \tag{3}$$

$$\forall (k \in \mathcal{K}, e \in \mathcal{E} : s(e) = v, v \in \mathcal{V}^{chrg}).$$

$$soc_{k,f(e)} - (soc_{k,v} - \Delta dis^{\text{chrg}} S_{v,f(e)} x_k^e + \Delta s_v^{\text{chrg}}) \leq M \cdot (1 - x_k^e), \tag{4}$$

$$\forall (k \in \mathcal{K}, e \in \mathcal{E} : s(e) = v, v \in \mathcal{V}^{chrg},).$$

$$soc_{k,f(e)} - (soc_{k,v} - \Delta dis^{\text{chrg}} S_{v,f(e)} x_k^e + \Delta s_v^{\text{chrg}}) \geq -M \cdot (1 - x_k^e), \tag{5}$$

$$\forall (k \in \mathcal{K}, e \in \mathcal{E} : s(e) = v, v \in \mathcal{V}^{chrg},).$$

$$x_k^e = 1 \implies soc_{k,f(e)} = soc_{k,v} - \Delta dis^{\text{chrg}} S_{v,f(e)} x_k^e, \tag{6}$$

$$\forall (k \in \mathcal{K}, e \in \mathcal{E} : s(e) = v, v \in \mathcal{V} \setminus \mathcal{V}^{chrg}).$$

$$soc_{k,f(e)} - (soc_{k,v} - \Delta dis^{\text{chrg}} S_{v,f(e)} x_k^e) \leq M \cdot (1 - x_k^e), \tag{7}$$

$$\forall (k \in \mathcal{K}, e \in \mathcal{E} : s(e) = v, v \in \mathcal{V} \setminus \mathcal{V}^{chrg}).$$

$$soc_{k,f(e)} - (soc_{k,v} - \Delta dis^{\text{chrg}} S_{v,f(e)} x_k^e) \geq -M \cdot (1 - x_k^e), \tag{8}$$

$$\forall (k \in \mathcal{K}, e \in \mathcal{E} : s(e) = v, v \in \mathcal{V} \setminus \mathcal{V}^{chrg}).$$

$$x_k^e = 1 \implies soc_{k,v} + \Delta s_v^{\text{chrg}} \leq CAP, \quad \forall (k \in \mathcal{K}, e \in \mathcal{E} : s(e) = v, v \in \mathcal{V}^{chrg}). \tag{9}$$

$$soc_{k,v} + \Delta s_v^{\text{chrg}} \leq CAP + M \cdot (1 - x_k^e), \quad \forall (k \in \mathcal{K}, e \in \mathcal{E} : s(e) = v, v \in \mathcal{V}^{chrg}). \tag{10}$$

$$soc_{k,v} = CAP, \quad \forall (k \in \mathcal{K}, v \in \mathcal{V}^{art}). \tag{11}$$

4.3.3 Demand satisfaction

To satisfy the demand, the sum over all the nodes that correspond to a specific request location should be equal to one to ensure demand satisfaction. However, a set covering formulation, instead of a set partitioning formulation, is used since this formulation has proven to be numerically more stable (Barnhart et al., 1998). This change replaces the equality sign with a greater-than- or-equal-to sign, ensuring that every request location is visited at least once.

Due to Constraint (2), only the outgoing arcs are considered in Constraint (12). In this equation, $\mathcal{N}^{cstm}$ is used instead of $\mathcal{V}^{cstm}$, since we want each request location to be visited more than once, rather than ensuring each time-location combination v of a request is visited more than once.

$$\sum_{k \in \mathcal{K}} x_k^e \geq 1, \quad \forall (e \in \mathcal{E} : s(e) = n, \ n \in \mathcal{N}^{cstm}). \tag{12}$$

4.3.4 Charging station capacity

The capacity at the charging station is modelled through Constraint (13). In this constraint, v_1 corresponds to the original charging station node and v_2 corresponds to its copy. To link these nodes, the following mapping function is introduced:

$$\phi : V_{\text{cpy}}^{\text{chrg}} \to V_{\text{org}}^{\text{chrg}}$$

which links the copy node v_2 to its original node v_1.

Since v_1 and v_2 correspond to the same location, both nodes have the same capacity that corresponds to the original capacity of the charging station. Therefore, to accurately reflect the real capacity, the constraint only needs to consider the capacity of one of the nodes.

$$\sum_{k \in \mathcal{K}} x_k^e \leq C_{v_1} \tag{13}$$

$$\forall (e \in \mathcal{E} : s(e) = v_1, v_2, \ v_1 \in \mathcal{V}_{org}^{\text{chrg}}, v_2 \in \mathcal{V}_{cpy}^{\text{chrg}}, \ \phi(v_2) = v_1)$$

4.3.5 Artifical depot nodes

In this section, the constraints regarding the artificial depot nodes are discussed. Each truck must have one arrival and one departure from these nodes. Therefore, the sum over all departures is set to one for each truck, as specified in Constraint (14). Similarly, the sum over all arrivals is set to one for each truck, as outlined in Constraint (15). Although Constraint (15) is already enforced by Constraint (14) and Constraint (2), it is included for clarification purposes.

For the same reason as in Constraint (12), the set $\mathcal{N}^{art}$ instead of the set $\mathcal{V}^{art}$ is used.

$$\sum_{n \in \mathcal{N}^{art}, e \in \mathcal{E}: s(e)=n} x_k^e = 1, \quad \forall (k \in \mathcal{K}). \tag{14}$$

$$\sum_{n \in \mathcal{N}^{art}, e \in \mathcal{E}: f(e)=n} x_k^e = 1, \quad \forall (k \in \mathcal{K}). \tag{15}$$

4.3.6 Arrival times

The arrival times are modelled in Constraint (16). The arrival of truck k at node v is equal to the corresponding time block of a node multiplied by the decision variables which capture whether or not the truck travels along arc e with end node v. If a node is not visited by truck k, the arrival time is set to 0. The arrival time are needed to calculate the tardiness costs.

$$\sum_{e \in \mathcal{E}: f(e)=v} B_{f(e)} \cdot x_k^e = a_{k,v}, \quad \forall (v \in \mathcal{V}, k \in \mathcal{K}) \tag{16}$$

4.3.7 Tardiness

The tardiness of request n is modelled in Constraint (17). The tardiness is added to the objective function and therefore the tardiness must be greater than or equal to the latest arrival at a request node.

$$tard_n \geq a_{k,v} - due_n, \quad \forall (k \in \mathcal{K},\ v = (n,b) \in \mathcal{V}^{cstm}). \tag{17}$$

4.4 Objective function

In this section, the objective function of the mathematical formulation of the described problem will be discussed.

The mathematical program aims to minimize the total transportation costs, including labour, charging, and tardiness costs. The labour costs are determined by the total duration of a route. In Objective Function (18), the first part consists of the labour costs. These costs are determined by the total duration of the route of a truck. This is modelled by the time block of the head of an arc, $B_{f(e)}$ minus the time block of the tail of an arc, B_v summed over all the arcs and trucks. The second sum corresponds to the charging costs; these are incorporated into the arcs when the tail node of an arc corresponds to a charging node. The third sum consists of the tardiness costs.

$$\begin{aligned} \min \Bigg\{ & \sum_{k \in \mathcal{K}, e \in \mathcal{E}: s(e)=v, v \in \mathcal{V}} C^{\text{lbr}} \cdot (B_{f(e)} - B_v) \cdot x_k^e \\ & + \sum_{k \in \mathcal{K}, e \in \mathcal{E}: s(e)=v, v \in \mathcal{V}^{\text{chrg}}} C_v^{\text{chrg}} \cdot x_k^e \\ & + \sum_{n \in \mathcal{N}^{cstm}} C_n^{\text{tard}} \cdot \text{tard}_n \Bigg\}. \end{aligned} \tag{18}$$

5 Methodology

In this section, the solution method for the described problem in Section 2 is explained. To give an overview of the methodology, the algorithm to obtain an integer solution is summarized in Figure 5.1. Firstly, the algorithm is analyzed with a set of routes, as explained in Section 5.1.3. Secondly, the Restricted Master Problem (RMP) is solved, as explained in Section 5.1.1. Within the column generation algorithm, a pricing problem, as explained in Section 5.1.2, is used. The pricing problem and the RMP are subsequently solved until no more routes with negative reduced costs can be found. The final integer solutions are obtained by a truncated column generation algorithm and a truncated price-and-branch algorithm and the difference in model performance is evaluated, as discussed in Section 5.2.

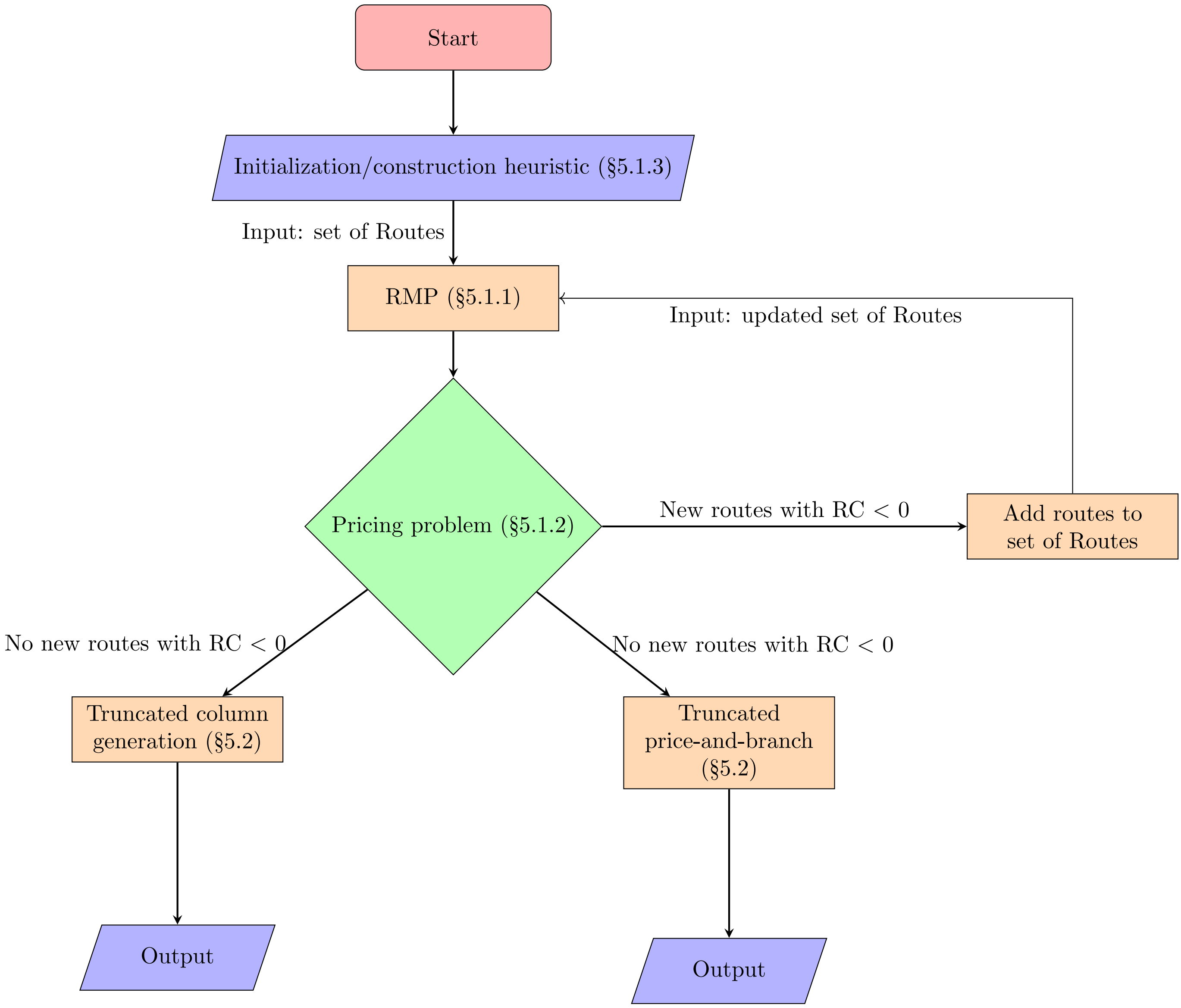

Figure 5.1: Flowchart illustrating the algorithm to obtain integer solutions. RC stands for reduced costs.

5.1 Column generation

In order to solve the problem with column generation, a standard set covering formulation will be used. The formulation of the standard set covering formulation is given below.

5.1.1 (Restricted) Master Problem

$$\min \sum_{z \in \mathcal{Z}} c_z x_z \tag{19}$$

Such that:

$$\sum_{z \in \mathcal{Z}} a_{n,z} x_z \geq 1 \quad \forall n \in \mathcal{N}^{cstm} \tag{20}$$

$$\sum_{z \in \mathcal{Z}} u_{n,z,b} x_z \leq C_n \quad \forall (n \in \mathcal{N}^{chrg}, b \in \mathcal{B}) \tag{21}$$

$$\sum_{z \in \mathcal{Z}} x_z \leq K \tag{22}$$

$$x_z \in \{0, 1\} \quad \forall z \in \mathcal{Z} \tag{23}$$

The objective function (19) minimizes the total costs of the route selection, and Constraint (20) makes sure that every request location is visited at least once. In Constraint (20), $a_{n,z}$ is equal to 1 if location n is in route z and 0 otherwise. Constraint (21) ensures the capacity of a charging station is not exceeded. In Constraint (21), $u_{n,z,p}$ is equal to 1 if charging station location n is occupied by route z in time block b and 0 otherwise. Constraint (22) ensures that for the scheduling horizon of one day, no more than K BETs are used. The set $\mathcal{Z}$ represents all the feasible routes. Constraint (21) can also be changed to a formulation where instead of set $\mathcal{N}^{chrg}$ and set $\mathcal{B}$, only the set $\mathcal{V}^{chrg}$ is used which contains the same information.

The definition of a route ensures constraints such as a maximum capacity of one container for the BETs. The LP relaxation of the given mathematical formulation is called the Master Problem (MP). In a column generation algorithm, the LP relaxation is solved for a subset of the variables. This problem is called the Restricted Master Problem (RMP). In every algorithm iteration, columns/routes are added to the RMP and the RMP is solved again.

A route consists of all the connected trips of a BET, mentioned in Section 2.

5.1.2 Pricing problem

The solution method for the pricing problem is based on the method described in Wu et al. (2022). For the network structure of the pricing problem, one depot and a set of customers and charging stations are considered. A truck is only able to transport one container at a time and thus after visiting a customer, the truck has to return to the depot. The nodes in the network correspond to the depot, customers, charging station locations and their corresponding time block, both incorporated in the set $\mathcal{V}$. A more detailed explanation of the graph and corresponding nodes and arcs can be found in Section 4.2.

The pricing problem aims to add promising routes to the set of routes in the RMP. Let $\mathcal{Z}$ be the set of all routes, and each route $z \in \mathcal{Z}$ consists of arcs. The cost of an arc (u, v) in the path is defined as $c_{u,v}$

with $u, v \in \mathcal{V}$. The costs are defined based on the type of the ending node v:

- If v is a request node, the cost $c_{u,v}$ includes:

$$c_{u,v} = C^{lbr}_{u,v} + C^{tard}_{v} \tag{24}$$

  where $C^{lbr}_{u,v}$ is the operational cost (driver's wage) of travelling from node u to node v, and C^{tard}_{v} represents the tardiness cost for delivering to request node v.

- If v is a charging node, the cost $c_{u,v}$ includes:

$$c_{u,v} = C^{chrg}_{v} + C^{lbr}_{u,v} \tag{25}$$

  where C^{chrg}_{v} represents the charging cost at node v, and $C^{lbr}_{u,v}$ is again the operational cost of travelling from node u to node v.

- If v is a depot node, the cost $c_{u,v}$ is:

$$c_{u,v} = C^{lbr}_{u,v} \tag{26}$$

  where $C^{lbr}_{u,v}$ is simply the operational cost of travelling from node u to node v.

Given these costs, the reduced costs can be calculated in the algorithm for each route. The column generation algorithm aims to find negative reduced costs and the pricing problem can be modelled as a resource-constrained shortest path problem. In Equation (27), σ_n, $\gamma_{n,b}$ and β correspond to the dual values of respectively Constraint (20), (21) and (22), which are all positive values.

$$\min \quad c_z - \sum_{n \in \mathcal{N}^{cstm}} a_{n,z}\sigma_n + \sum_{n \in \mathcal{N}^{chrg}, b \in \mathcal{B}} u_{n,z,p}\gamma_{n,b} + \beta \tag{27}$$

The pricing problem continues to generate routes as long as routes with negative reduced costs can be found. The reduced costs of an arc are defined as follows.

- If u is an artificial depot node, the reduced costs of the arc is:

$$rc_{u,v} = \beta \tag{28}$$

- If v is a request node, the reduced costs are calculated as follows:

$$rc_{u,v} = c_{u,v} - \sigma_n \tag{29}$$

  where σ_n is the dual value related to request n, with $v = (n, b)$.

- If v is a charging node, the reduced costs are calculated as follows:

$$rc_{u,v} = c_{u,v} + \gamma_{n,b} \tag{30}$$

  where $\gamma_{n,b}$ is the dual value related to charging station n at time block b.

- If v is a depot node, the reduced cost of the arc is equal to $c_{u,v}$.

In Figure 4.2, the representation of a path in the time-location network can be seen. The reduced costs are determined in the following steps.

- Arc 1 $\rightarrow$ 8 is associated with the dual value β
- Arc 8 $\rightarrow$ 30 is associated with a cost for travelling that arc and the dual value σ for visiting that customer
- Arc 30 $\rightarrow$ 24 is associated with a cost for travelling that arc
- Arc 24 $\rightarrow$ 11 is associated with a cost for charging at the charging station, travelling to node 24 and the dual value for visiting node 24, γ.
- Arc 11 $\rightarrow$ 30 is associated with a cost for travelling that arc and the dual value, σ, for visiting the customer
- Arc 40 $\rightarrow$ 13 is only associated with the costs for travelling that arc
- Arc 13 $\rightarrow$ 7 is only associated with the costs for travelling that arc

The pricing problem can be solved by a Multi Label Setting Algorithm as described in Wu et al. (2022). The algorithm aims at finding routes with minimal reduced costs while ensuring the state of charge, SoC, remains positive throughout the whole route. Since multiple routes can be formulated from the source to the sink, each node contains multiple labels. In the algorithm, the label l_v^m corresponds to the m^{th} label of node v. Each label consists of the SoC value. the original cost value and the reduced cost value, denoted by (SoC_v^m, c_v^m, rc_v^m). The original cost value corresponds to the real costs of a route without including the dual values. The labels are updated following the scheme denoted in (31). The first update rules states that if a charging station is visited, there needs to be sufficient room for extra charge and the second update rule states that energy infeasible routes cannot be chosen.

$$l_{f(e)}^m = \begin{cases} \text{None} & \text{if } SoC_{s(e)}^m + \Delta s_{s(e)}^{\text{chrg}} > CAP \\ & \quad \forall e \in \mathcal{E} : s(e) = (n,b), n \in \mathcal{N}^{chrg} \\ \text{None} & \text{if } SoC_{s(e)}^m - S_{s(e),f(e)} \leq 0 \quad \forall e \in \mathcal{E} \\ (rc_{s(e)}^m + rc_{s(e),f(e)}, SoC_{s(e)}^m - S_{s(e),f(e)}, c_{s(e)}^m + c_{s(e),f(e)}) & \text{else if} \\ & \forall e \in \mathcal{E} : s(e) = (n,b), n \in \mathcal{N} \setminus N^{chrg} \\ (rc_{s(e)}^m + rc_{s(e),f(e)}, SoC_{s(e)}^m - S_{s(e),f(e)} + \Delta s_{s(e)}^{\text{chrg}}, c_{s(e)}^m + c_{s(e),f(e)}) & \text{else if} \quad \forall s(e) = (n,b), n \in \mathcal{N}^{chrg} \end{cases} \tag{31}$$

Dominance rules are introduced to favour labels that will for sure contribute more to a better solution value and thus limit the search space without losing the potential for finding an optimal solution. These rules are used to compare different labels and determine which ones can be discarded.

A label is dominated by another label if the reduced costs are higher, the original costs are higher and the state of charge is lower as represented in (32): label k is dominated by label m. All conditions in (32) have to be satisfied to achieve dominance.

$$SoC_v^m \geq SoC_v^k,\ c_v^m \leq c_v^k \ and \ rc_v^m \leq rc_v^k \tag{32}$$

This means that if there is a route to a node which is more cost-effective, due to lower costs, more energy efficient, due to a higher state of charge value and has lower reduced costs, this route (corresponding to label l_v^m) will always be favoured over a route with higher costs, lower state of charge value and higher reduced costs (corresponding to label l_v^k). In other words, label l_v^k can never lead to a better solution value than l_v^m and thus, label l_v^k can be discarded without loss of generality. Also note that the labels are linked to nodes which correspond to a time-location combination in the graph.

Routes are generated by backtracking in which lower reduced costs are favoured.

After solving the pricing problem, the selected columns are added to the RMP. Columns are iteratively added to the RMP until no more columns with negative reduced costs can be found. When the LP relaxation is solved to optimality, a lower bound on the objective value for the integer solution is found.

5.1.3 Initialization

The RMP is initialized with a route consisting of all the customers. This route is not a feasible route and the cost is set to 1,000,000. Given the corresponding cost values of the described problem, this route can never be part of an optimal solution unless the problem is infeasible due to time, truck availability or state of charge constraints. If this route is part of the optimal solution, the problem is infeasible.

Another approach is to use a greedy/construction heuristic to generate feasible start solutions. By initializing the RMP with feasible routes, the heading-in effect can be reduced. Routes are iteratively built up by continuously adding feasible nodes to the path in a smart way. The following restrictions are taken into account and can also be found in Algorithm 1.

- Routes start at the source node and end at the sink node (line 4 and line 21-28)
- After returning to the depot, it is first checked whether a request node can be added to the route while taking into account Constraint (33) and (34). In these constraints u, corresponds to a time-location combination of the depot location: $u \in \mathcal{V}^{depot}$.

$$SoC_u \geq 2 \cdot S_{u,v} \cdot \Delta dis^{\text{chrg}} \tag{33}$$

$$B_u + 2 \cdot TB_{u,v} \leq ET \tag{34}$$

  In Constraint (34), ET is defined as the latest time at which the trucks need to be back at the depot. Constraint (33) and (34) ensure that the infeasible paths, due to state of charge or time limitations, are not explored. This step is modelled in line 7.

- Request nodes have priority over charging stations nodes and the sink nodes (line 6)
- Charging station nodes have priority over the sink nodes (line 6)
- If no request node can be added, it is first checked whether a charging station can be visited, and subsequently, another charging station or request node (line 6)

- If no request location or charging station location can be visited, the route is terminated by returning to one of the artificial depot nodes $v \in \mathcal{V}^{art}$ (line 21 - 28)
- The maximum amount of routes that can be generated equals the amount of available trucks (line 3)
- If the capacity of a charging station is reached at a certain time block, the node (representing the location and corresponding time block) is removed from the graph and cannot be visited anymore (line 13 - 18)
- If a request has been fulfilled, the corresponding request nodes are removed from the graph and cannot be visited anymore (line 10 - 12)
- After generating the initial set of feasible routes, the algorithm checks if all customer requests have been served. If some requests remain unserved, the RMP is initialized by the previously mentioned infeasible route (line 32 - 36)
- Furthermore, the constraints mentioned in Section 2 also apply here.

In this algorithm, nodes are iteratively added to the route until the route can no longer be extended and is terminated at the artificial depot node. The algorithm prioritizes different nodes based on their importance, focusing on serving all requests and extending the route as much as possible.

The algorithm aims to cover as many request nodes as possible within a route by prioritising the extension of routes and ensuring all requests are served. However, this focus on maximizing route length and node coverage means that the initial solution generated by this heuristic is likely to have an objective value that is far from optimal. The primary goal at this stage is feasibility rather than optimality. The algorithm does not guarantee feasible solutions. If no feasible solution can be found, the algorithm is initialized with the previously mentioned infeasible route, as is modelled in line 32 - 36 of the algorithm. The costs of the routes are determined in the same way as in the pricing problem. The algorithm can be found in Algorithm 1. Feasibility means Constraint (33) and (34) and the charging station capacity restriction are satisfied. Furthermore, only arcs from the graph mentioned in Section 4.2 are considered.

Algorithm 1 Generate Feasible Solutions

```
Input: Nodes 𝒩, Arcs ℰ, Source Node source, Sink Node sink, Maximum Routes M, Capacity
  charging station capacity
Initialize routes ← ∅
while not all requests served and |routes| < M do
  Initialize route at source
  while route not terminated do
    for each Node in prioritized order (Requests > Charging Stations > Depot) do
      if Route is feasible and Node ≠ sink then
        Add Node to Route
        Extend Route
        if Node = request then
          Delete corresponding request nodes
        end if
        if Node = charging station then
          Current capacity charging station ++
          if Current capacity = capacity then
            Delete charging station node
          end if
        end if
        break
      end if
      if Route is feasible and Node = sink then
        Add Node to Route
        break
      end if
    end for
    if !Extend Route then
      Terminate Route
    end if
  end while
  Add route to routes
end while
if not all requests served then
  routes ← ∅
  Use initial infeasible route
  Add initial infeasible route to routes
end if
Output: routes
```

5.1.4 Reducing computation time

Primal degeneracy in column generation can lead to the so-called tail-off effect, resulting in slow convergence (Amor et al., 2009). To avoid this slow convergence, the pricing problem can be stopped after no improvement in objective value can be seen for a certain amount of observations. This means that the LP relaxation is not solved to optimality since routes with negative reduced costs can still be found.

Moreover, using a feasible solution as the initial solution is also expected to decrease the computation time compared to an initial infeasible solution with very high costs.

In order to keep the RMP to a manageable size, column management can be used. After a certain amount of iterations, routes with positive reduced costs can be deleted and stored in a pool. These routes are then not used in solving the RMP, but for every iteration of the algorithm the pool of stored routes can be checked for routes with negative reduced costs. If routes with negative reduced costs are found, these columns are again added to the RMP.

5.2 Obtaining integer solutions

As suggested by de Vos et al. (2022), truncated column generation can be used to find integral solutions for the LP relaxation. In this heuristic, a column generation phase and fixing phase are iteratively performed. After an early stopping criterion is met for the column generation phase, variables above a certain threshold are fixed and set to 1 (if the path is not part of the initial solution) and the column generation phase is restarted. This process is repeated until an integer solution is found. In this research, all the variables that are one and the highest value among the fractional values are set to one.

Next to the truncated column generation algorithm, a truncated price-and-branch algorithm is used to obtain integer solutions. This algorithm consists of applying an MIP solver when the Master Problem has been solved to optimality. All the routes generated in the column generation algorithm are then used to obtain an integer solution.

For both heuristics, the optimality gap compared to the lower bound is computed to evaluate the model performance.

5.2.1 Reducing computation time

The reductions in computation time for the column generation algorithm also apply to the truncated column generation algorithm. However, in the truncated column generation algorithm it is not needed to solve the LP relaxation to optimality in every iteration. And thus, if after 12 iterations, the objective value has not decreased by 0.1 or more in absolute value, the generation phase is terminated and the relevant variables are set to one. To further speed up the pricing problem, redundant arcs and nodes can be excluded from the formulation. If within the fixed solutions, the charging station capacity is obtained at a certain time point, this node can be excluded from the pricing problem. If a request has been satisfied within the fixed solution values, the corresponding request nodes can be excluded from the pricing problem. These changes in the set of nodes are applied to the pricing problem to speed up the algorithm.

6 Data

In this section, the data will be discussed. First, the data cleaning will be briefly discussed, followed by an explanation of the procedure to calculate the travel times and distances between the different locations.

6.1 Data cleaning

A realistic test case is needed to evaluate the mathematical formulation and solution method. Data from Van Berkel Logistics was provided to TNO, including data on the shipment of containers via barges and trucks for 2017, 2018, and 2019. Only the shipment via trucks is considered in this research. Van Berkel is a company responsible for the shipment of containers from the harbour in Rotterdam to inland terminals, corresponding to the depots, and ultimately to the customer. Across those 3 years, over 150,000 orders were documented.

Van Berkel Logistics uses a system which is not always accurate, and sometimes values are filled in by hand, leading to missing values. Since this is a test case, the missing values are filled with logical and conservative replacements and the exact values are not needed.

The procedure to fill in the missing values for the deadlines can be found in Figure B.1. In this procedure, variables are linked to the deadline with a certain value of Λ. For missing deadline values, several other values can be linked to the missing value: the time at which the container arrived at the depot via a barge, the time at which the container can be picked up from the depot, the time at which the container was processed at the depot, the time at which the container was picked up by the barge to be transported to the depot, the time at which the container left the depot. All these values are before the deadline of the container and via these values an approximation of the real deadline can be made.

To evaluate the model performance, trips to customers from 10-07-2019 were used, resulting in a total of 72 request locations. An overview of all the one-way distances from the depot to the locations within the dataset is given in Figure 6.1. Since there are multiple requests that coincide at the same location, the amount of unique distances is not equal to the amount of requests. Furthermore, some request locations are very close to each other and thus their distances fall into the same bar chart.

The unique values of the travel times between a location within the dataset and the depot are given in Figure 6.2. Travel times vary between 81 minutes and 0 minutes, corresponding to the distance between the depot and the charging station at the depot. Once again, several requests correspond to the same location, resulting in the same travel time to the depot.

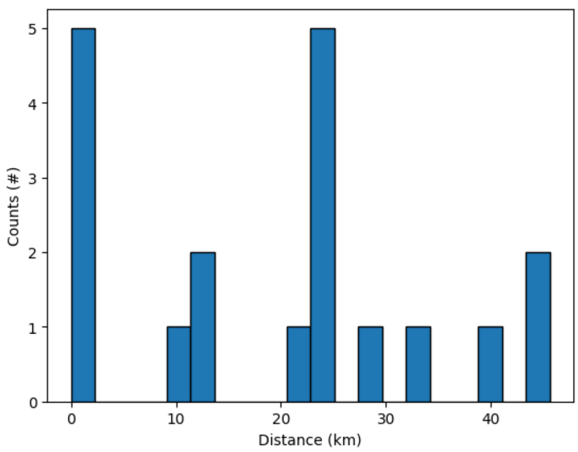

Figure 6.1: One-way distance to the depot

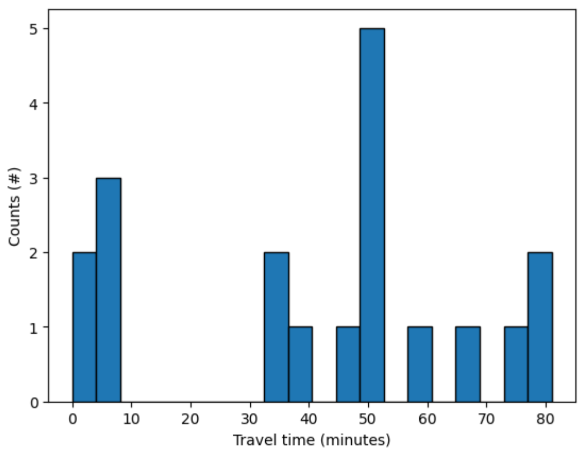

Figure 6.2: One-way travel time to the depot

6.2 Travel time and distance

The Haversine function is used to calculate the travel time between nodes. The Haversine function has been previously used in VRPs to determine the distance between two nodes in a network (Gunawan, Susyanto, & Bahar, 2019). This function calculates the great circle distance between two nodes. In this research, this distance is then used to calculate the travel time between two nodes. The travel time is calculated following the procedure in Pingen et al. (2022). The Haversine function calculates the distance in kilometres, and the algorithm returns the travel time in hours. The algorithm can be found in Algorithm 2. In this algorithm, a speed of $20km/h$ in urban areas, a speed of $50km/h$ in rural areas and a speed of $70km/h$ in highway parts is used.

In the algorithm, s corresponds to the travel time, d corresponds to the remaining distance in the route and t corresponds to the part of the distance of the route corresponding to either the urban area, the rural area or the highway area.

The first 10 kilometres are assumed to be in urban areas, so the first 10 kilometres of a route are driven at a speed of $20km/h$. The following 35 kilometres are assumed to be in rural areas, and the remaining kilometres in a route are assumed to be driven on a highway. Assume a route has a length of 30 kilometres; the first 10 kilometres are then driven at a speed of $20km/h$ and the remaining 20 kilometres at a speed of $50km/h$, resulting in a travel time of 54 minutes, i.e. 6 time blocks.

Algorithm 2 Travel time

```
INPUT = distance
d = 1.2 · distance
s = 0
if d > 0 then
    t = min(10, d)
    s = s + t/20
    d = d − t
end if
if d > 0 then
    t = min(35, d)
    s = s + t/50
    d = d − t
end if
if d > 0 then
    t = d
    s = s + t/70
    d = d − t
end if
OUTPUT = s
```

7 Results

This section discusses the findings of this research. First, the data instances utilized in the analysis will be examined. Following this, the results from applying the mathematical formulation in CPLEX will be presented. Thirdly, the outcomes from the proposed methodology outlined in Section 5 will be discussed. The section is concluded by a small sensitivity analysis in which the effect of the amount of charging stations and battery capacity on the outcome is analysed.

7.1 Instances

The characteristics of the solved instances are given in Table 7.1. The chargers are located at the depot. A battery-electric truck with a capacity of 95 kW is used and the discharge is assumed to be constant at 1 kW/km. This assumption is in line with the real average discharge rate, discussed in Section 2. The charging power of a charger is assumed to be 120 kW, leading to a constant charge of 0.033 kW/s. It is assumed that the trucks cannot leave before 06:00 from the depot and have to be back at the depot before 18:00, resulting in 72 time blocks of 10 minutes. Since this research is used to prove a working implementation of the algorithm, cost values are used which may not represent the true costs. For all the individual costs, including tardiness costs, driver's wage and charging costs, a cost of €1 per time block is used, which aligns with earlier model evaluation methods by TNO. In reality, these cost values will be higher.

Table 7.1: Characteristics of the instances

Instance	**Customers (#)**	**Trucks (#)**	**Chargers (#)**	**Time blocks (#)**
A	4	2	2	72
B	5	2	2	72
C	5	3	2	72
1	10	4	2	72
2	61	19	2	72
3	61	19	4	72
4	72	24	4	72
5	72	19	4	72

Given the distances in Figure 6.1, the capacity of the truck is set to 95 kW to ensure multiple visits to the charging station are needed and thus the charging station capacity is a limiting factor in the optimization of the objective value.

7.2 Exact formulation

The exact formulation, formulated in Section 2, was implemented in CPLEX 22.11 in JAVA. This section aims to emphasize the need for efficient algorithms to solve the proposed problem. The running time was set to an hour and the results for instance A, B and C are shown in Table 7.2. The optimality gap returned by CPLEX is the relative optimality gap between the best-found solution so far and the best-known bound. The results indicate that even though the amount of trucks and customers are relatively low, CPLEX fails to find the optimal solution value within a reasonable amount of computation time.

Table 7.2: Results for the exact formulation after 1 hour of solving

Instance	Constraints (#)	Variables (#)	Opt. gap (%)	Best found solution (€)
A	10209	5288	27.81	171
B	11470	6108	50.98	186
C	17169	9161	30.33	132

7.3 Column generation

In this section, the results from applying column generation in combination with a Multi Label Setting shortest path algorithm to the LP relaxation are discussed.

The results are displayed in Table 7.3. For all instances, the LP relaxation was solved to optimality within an hour. The total computation time was mostly determined by the pricing problem. For Instance 3, out of the total time required to solve the LP relaxation to optimality, approximately 5.7% of the computation time goes into solving the RMP, while the remaining 94.3% goes into solving the pricing problem. An increase in customers leads to an increase in computation time. More difficult instances, due to fewer charging stations or trucks, also exhibit larger computation times.

The change in objective value in the column generation algorithm for the LP relaxation for instance 2 can be seen in Figure 7.1. The RMP is initialized with an infeasible route with a very high cost: 1,000,000. In each iteration, a maximum of 2000 routes with negative reduced costs are added to the RMP. It can be seen that in the first 60 iterations, the column generation algorithm fails to find a feasible optimal solution since the infeasible route remains part of the optimal solution.

Table 7.3: Results of the instances for solving the LP relaxation

Instance	Iterations (#)	Comp. time Pricing (s)	Comp. time RMP (s)	LP solution (€)
A	9	2	0	132
B	12	5	0	179.5
C	9	2	0	132
1	20	36	3	285
2	78	1049	119	1228
3	72	745	45	1220.67
4	90	910	101	1858
5	98	2001	197	2201.56

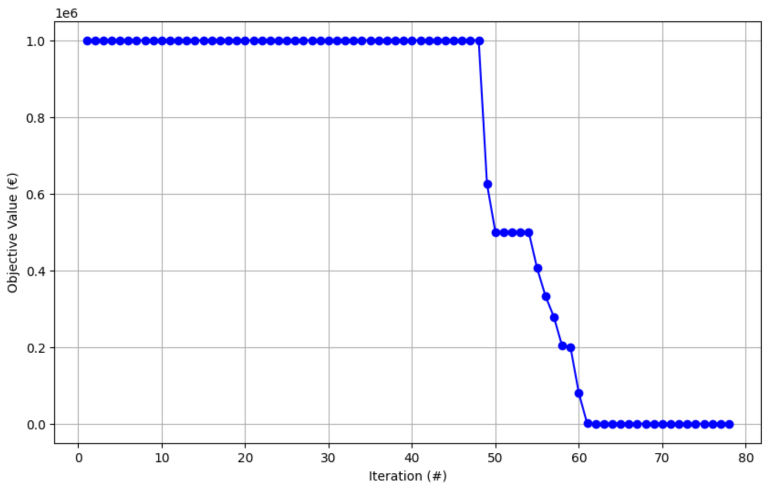

Figure 7.1: Column generation instance 2

Due to the large cost value of the infeasible route used for initialization, the last 18 iterations, in which the infeasible route is not part of an optimal solution, cannot be distinguished in Figure 7.1 and are therefore displayed in Figure 7.2. In the figure, the convergence to the optimum and the tail-off effect can be clearly seen.

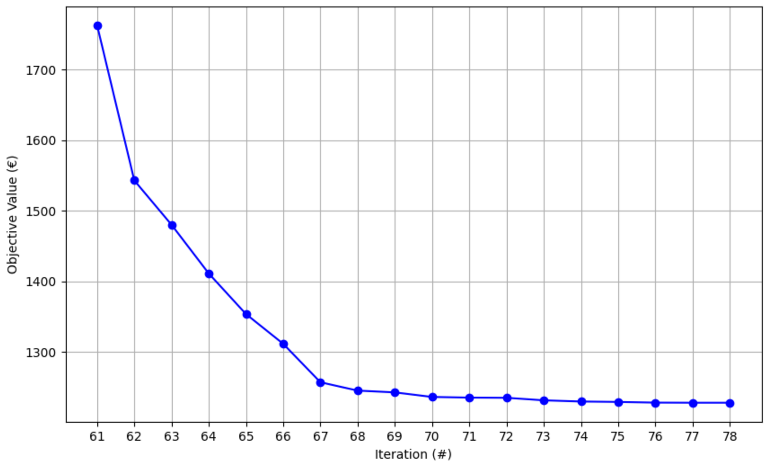

Figure 7.2: Column generation instance 2 for the last 18 iterations

7.4 Intialization

The initialization of the algorithm can be changed to speed up the column generation algorithm and solve the LP relaxation. Previously, the RMP was initialized with a single infeasible route that contained all the requests and did not consider charging. Now the RMP is initialized with a set of feasible routes and the column generation algorithm is run again. The feasible routes are generated by the construction heuristic described in Section 5.1.3 within a second for both Instance 2 and 3. The results are displayed in Table 7.4. For Instance 3, the computation time decreases by approximately 11.3%. The amount of iterations decreases by more than 50%. The optimality gap for the construction heuristic is 61.1%, indicating that this is not a good solution technique.

For Instance 2, the computation time decreases by approximately 8.6% and the amount of iterations by more than 50%. The construction heuristic (CH) finds a feasible solution with an optimality gap of 50.4%, indicating that the heuristic solution is far from optimal.

Table 7.4: Results of the instances with a different initialization

Instance	Iterations (#)	Comp. time LP (s)	Comp. time TPB (s)	LP (€)	CH (€)
2	28	965	5	1228	1847
3	32	711	2	1220.67	1969

The iterations for the column generation algorithm for Instance 2 with a feasible initialization are given in Figure 7.3. Compared to the original initialization, it can be seen that the amount of iterations drastically decreases and that the initial solution is deleted earlier from the optimal solution, meaning the heading-in effect is decreased. In the last iterations, the relative difference in change of objective function is small due to the infamous tail-off effect of column generation.

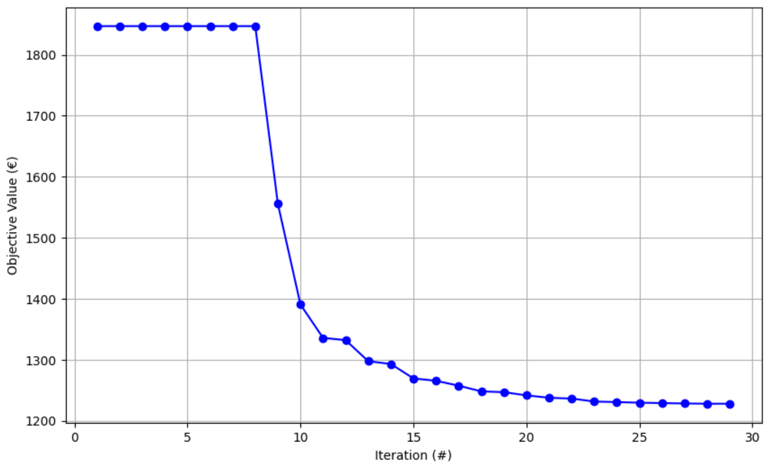

Figure 7.3: Column generation Instance 2

7.5 Obtaining integer solutions

After the LP relaxation has been solved to optimality, truncated column generation (TCG) and truncated price-and-branch (TPB) are applied as described in Section 5. The computation time of the truncated column generation algorithm is equal to the sum of the computation time for column generation on the LP relaxation and the computation time for applying truncated column generation, the same reasoning applies to the truncated price-and-branch algorithm. In the truncated column generation algorithm, the column generation phase is cut off if the objective value has not changed enough after a certain amount of iterations. This amount was not reached in solving the LP relaxation to optimality and thus the computation time for solving the LP relaxation to optimality is used.

From the results in Table 7.5 and 7.6, it can be seen that the truncated price-and-branch algorithm and truncated column generation algorithm perform both quite well in terms of optimality gap. For the truncated column generation algorithm, the highest found optimality gap was 4.91%. The computation time for the truncated column generation is much higher compared to TPB since new routes are added to the formulation in every iteration. For these results, the RMP was initialized with the previously mentioned infeasible solution.

Table 7.5: Results of the truncated column generation algorithm for the instances

Instance	Iterations (#)	Comp. time TCG (s)	TCG solution (€)	Opt. gap (%)
A	0	0	132	0.00
B	1	0	181	0.84
C	0	0	132	0.00
1	10	71	299	4.91
2	11	2198	1230	0.16
3	7	1132	1223	0.11
4	14	2535	1858	0.00
5	13	3827	2215	0.61

The truncated price-and-branch algorithm (TPB) was for every instance solved within 15 seconds. For all instances, the algorithm returns a solution with a relatively low optimality gap, indicating good algorithm performance. The highest optimality gap was found for Instance B: 0.84%. The results are displayed in Table 7.6.

Table 7.6: Results of the TPB algorithm for the instances

Instance	Comp. time TPB (s)	TPB solution (€)	Opt. gap (%)
A	0	132	0.00
B	3	181	0.84
C	0	132	0.00
1	1	287	0.70
2	5	1229	0.08
3	7	1222	0.11
4	14	1861	0.16
5	13	2216	0.66

The results for Instances 2 and 3 without speeding up the pricing problem by removing redundant arcs and nodes in the truncated column generation algorithm are displayed in Table 7.7. It can be seen that the computation time increases by more than 122% for Instance 2 and 133% for Instance 3.

Table 7.7: Results for Instances 2 and 3 without speeding up the algorithm for the pricing problem

Instance	Iterations (#)	Comp. time TCG (s)	TCG solution (€)	Opt. gap (%)
2	16	4900	1230	0.16
3	9	2648	1222	0.11

7.6 Occupation charging stations

This section aims to visualize the occupation of the charging station during the scheduling horizon of 72 time blocks.

The utilized capacity at the charging station at the depot for Instance 4 is depicted in Figure 7.4. This solution is based on the outcome of the truncated column generation algorithm. It can be seen that the full capacity of the charging station is fully utilized in 5 time blocks. After time block 41, no charging actions are performed.

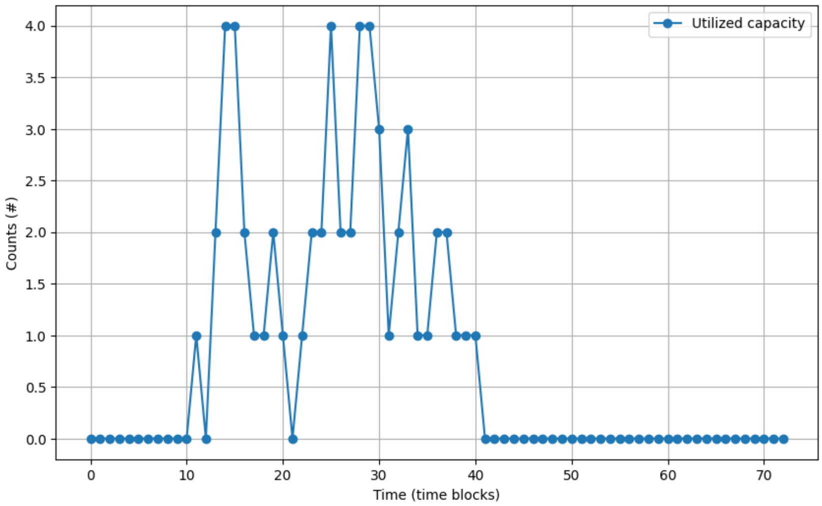

Figure 7.4: The utilized capacity at the charging station.

In Figure 7.5, the utilization of the charging station at the depot is depicted for Instance 5. It can be seen that the occupancy of the charging station increases while the number of trucks decreases compared to Instance 4. Even though there are fewer trucks, the full capacity of the charging station is reached once more compared to the case with 24 trucks. This is caused by the fact that the same amount of kilometres has to be driven with a smaller fleet of trucks and thus more charging actions are needed.

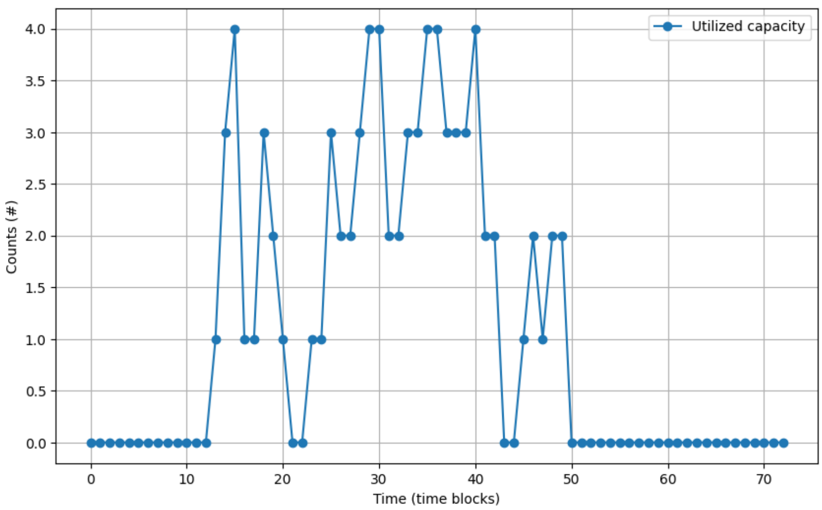

Figure 7.5: The utilized capacity at the charging station for 19 trucks

7.7 Change in battery capacity

This section aims to present the effect of battery capacity on the costs.

For Instance 5, the capacity of the BET is set to $125kW$ instead of 95 kW and the results are displayed in Table 7.8 and 7.9. It can be seen that a larger battery capacity of the BET leads to lower costs. The results in Table 7.8 correspond to the LP relaxation and truncated price-and-branch algorithm. The results in Table 7.9 correspond to the truncated column generation algorithm.

Table 7.8: Results for instance 5 when increasing the battery capacity

Instance	Iterations (#)	Comp. time LP (s)	Comp. time TPB (s)	LP solution (€)	TPB solution (€)
5	100	2506	9	2045	2046

Table 7.9: Results for instance 5 when increasing the battery capacity

Instance	Iterations (#)	Comp. time TCG (s)	TCG solution (€)	Opt. gap (%)
5	18	4417	2056	0.54

8 Conclusion & Discussion

In this study, the single depot joint routing and charging scheduling problem for battery-electric trucks (BETs) was investigated, accounting for range constraints, capacitated charging stations, partial charging, and varying customer demand. The proposed solution method involved the development of two column generation heuristics: a truncated price-and-branch algorithm and a truncated column generation algorithm. Both methodologies relied on a column generation algorithm integrated with a Multi-Label Setting algorithm to solve the pricing problem. The lower bounds on the solution values were determined by solving the LP relaxation to optimality.

Eight instances, varying in requests, amount of chargers and amount of available trucks were analysed. The three smallest instances were implemented in CPLEX to obtain an exact solution, but CPLEX failed to find an optimal solution within one hour of computation time. For all three instances, the optimality gap was larger than 27%. These results highlight the need for new and better solution methods. Furthermore, a simple construction heuristic was developed to quickly find feasible solutions. However, the obtained solution values resulted in an optimality gap of more than 50%, highlighting the need for better solution methods even more.

The largest data instance, involving 72 request locations, 24 trucks, and 4 chargers, was successfully solved in 2213 seconds using the truncated price-and-branch algorithm, achieving an optimality gap of 0.66%. The truncated column generation algorithm solved the same instance in 3827 seconds, with an optimality gap of 0.61%. These outcomes demonstrate good performance of both algorithms in terms of optimality gap and computation time. Moreover, both column generation approaches provided for all instances an optimality gap that is smaller than 5%.

Several actions were implemented to improve computation times, including removing redundant arcs and nodes in the pricing problem of the truncated column generation algorithm. The impact of these actions was evident for Instances 2 and 3, where the computation time increased by more than 120% when redundant arcs and nodes were not removed from the graph. These results prove the significant benefits of removing redundant arcs and nodes from the formulation to reduce computation time.

Furthermore, initializing the column generation algorithm with a feasible solution rather than an artificial or infeasible solution led to significantly fewer iterations and reduced computation time to reach the optimum. A sensitivity analysis revealed that utilizing trucks with lower battery capacity resulted in higher costs. Similarly, fewer available charging stations produced similar results, indicating the influence of battery capacitiy and charging stations on the objective value. This sensitivity analysis answered the subquestions of this research.

From the results, it can also be concluded that most of the computation time goes into the pricing problem indicating potential room for improvement. To further enhance the computational efficiency, more efficient algorithms for the pricing problem or alterations to the currently used algorithm can be made, including different dominance checks for the labels that further reduce the search space. Now, a conservative dominance check is used.

In the current research, a constant charge amount is used, which prohibits trucks from visiting a charging station if the current capacity added with the constant charge amount exceeds the maximum capacity of a truck. Further research could allow for a variable amount, which includes adding more big-M constraints to the exact formulation.

Overall, both the truncated column generation algorithm and the truncated price-and-branch algorithm proved effective in solving the problem, yielding nearly equivalent solution qualities. The truncated price-and-branch algorithm demonstrated a lower computation time. Both algorithms can be used to solve the problem within reasonable computation time and with a reasonable optimality gap.

In conclusion, this research successfully developed and validated column generation algorithms for the problem of joint routing and charging scheduling of BETs. Future research should focus on enhancing the computational efficiency and the practical implementation of the algorithms.

As a feasible initialization has proven to be computionally beneficial, further research should focus on improving the initial solution to the column generation algorithm to speed up the algorithms further. Since charging stations for BETs are not widely available, a promising direction for further research could be to add a component of optimizing the place of public charging stations. One could also look at the joint optimization of charging actions and mandatory breaks for drivers.

Moreover, certain assumptions have been made in this research which can limit the current practical implementation of the algorithms. Currently, only trips between a depot and customers are considered in the graph formulation. Within container transport, it is also possible to, for example, deliver a container to a customer and subsequently pick up an empty container and transport it back to the depot. To allow for such trips and thus enhance the algorithm's practical implementation, the graph formulation needs to be extended. Furthermore, realistic cost values should be used in order to effectively schedule BETs. Different cost values could also lead to a change in algorithm performance.

To further enhance the practical implementation of the algorithms, stochastic variables such as traffic congestion, waiting times at charging stations and stochastic unloading times can be added to the problem formulation.

Further research in the previously mentioned aspects can help improve the sustainability of the transport sector.

References

Ali, Y., Socci, C., Pretaroli, R., & Severini, F. (2018). Economic and environmental impact of transport sector on europe economy. *Asia-Pacific Journal of Regional Science*, *2*, 361–397.

Amor, H. M. B., Desrosiers, J., & Frangioni, A. (2009). On the choice of explicit stabilizing terms in column generation. *Discrete Applied Mathematics*, *157*(6), 1167–1184.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W., & Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations research*, *46*(3), 316–329.

Bluekens. (2024). *Oplaadpunten voor elektrische vrachtwagens.* `https://bluekenstruckenbus.nl/elektrisch/truck-oplaadpunten-in-nederland`. (Accessed: 2024-16-04)

Bragin, M. A., & Tucker, E. L. (2022). Surrogate "level-based" lagrangian relaxation for mixed-integer linear programming. *Scientific Reports*, *12*(1), 22417.

Bragin, M. A., Ye, Z., & Yu, N. (2024). Toward efficient transportation electrification of heavy-duty trucks: Joint scheduling of truck routing and charging. *Transportation Research Part C: Emerging Technologies*, *160*, 104494.

Center, A. F. D. (2020). *Maps and data–average annual vehicle miles traveled by major vehicle category.*

Chabrier, A. (2006). Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, *33*(10), 2972–2990.

Cui, S., Gao, K., Yu, B., Ma, Z., & Najafi, A. (2023). Joint optimal vehicle and recharging scheduling for mixed bus fleets under limited chargers. *Transportation Research Part E: Logistics and Transportation Review*, *180*, 103335.

Desaulniers, G., Desrosiers, J., & Solomon, M. M. (2006). *Column generation* (Vol. 5). Springer Science & Business Media.

de Vos, M., van Lieshout, R. N., & Dollevoet, T. (2022). Electric vehicle scheduling with capacitated charging stations and partial charging. *arXiv preprint arXiv:2207.13734*.

El Hachemi, N., Gendreau, M., & Rousseau, L.-M. (2011). A hybrid constraint programming approach to the log-truck scheduling problem. *Annals of Operations Research*, *184*(1), 163–178.

Elshaer, R., & Awad, H. (2020). A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants. *Computers & Industrial Engineering*, *140*, 106242.

Froger, A., Jabali, O., Mendoza, J. E., & Laporte, G. (2022). The electric vehicle routing problem with capacitated charging stations. *Transportation Science*, *56*(2), 460–482.

Gaudioso, M. (2020). A view of lagrangian relaxation and its applications. *Numerical Nonsmooth Optimization: State of the Art Algorithms*, 579–617.

Gunawan, S., Susyanto, N., & Bahar, S. (2019). Vehicle routing problem with pick-up and deliveries using genetic algorithm in express delivery services. In *Aip conference proceedings* (Vol. 2192).

Jia, Y.-H., Mei, Y., & Zhang, M. (2021). A bilevel ant colony optimization algorithm for capacitated electric vehicle routing problem. *IEEE Transactions on Cybernetics*, *52*(10), 10855–10868.

Jungwirth, A., Desaulniers, G., Frey, M., & Kolisch, R. (2022). Exact branch-price-and-cut for a hospital therapist scheduling problem with flexible service locations and time-dependent location capacity. *INFORMS Journal on Computing*, *34*(2), 1157–1175.

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.

Kluschke, P., Gnann, T., Plötz, P., & Wietschel, M. (2019). Market diffusion of alternative fuels and powertrains in heavy-duty vehicles: A literature review. *Energy Reports*, *5*, 1010–1024.

Kucukoglu, I., Dewil, R., & Cattrysse, D. (2021). The electric vehicle routing problem and its variations: A literature review. *Computers & Industrial Engineering*, *161*, 107650.

Kumar, S. N., & Panneerselvam, R. (2012). A survey on the vehicle routing problem and its variants.

Labadie, N., Prins, C., & Yang, Y. (2014). Iterated local search for a vehicle routing problem with synchronization constraints. In *Icores* (pp. 257–263).

Lam, E., Desaulniers, G., & Stuckey, P. J. (2022). Branch-and-cut-and-price for the electric vehicle routing problem with time windows, piecewise-linear recharging and capacitated recharging stations. *Computers & Operations Research*, *145*, 105870.

Lam, E., & Hentenryck, P. V. (2016). A branch-and-price-and-check model for the vehicle routing problem with location congestion. *Constraints*, *21*, 394–412.

Lübbecke, M. E. (2010). Column generation. *Wiley encyclopedia of operations research and management science*, *17*, 18–19.

Ozturk, O. (2020). A truncated column generation algorithm for the parallel batch scheduling problem to minimize total flow time. *European Journal of Operational Research*, *286*(2), 432–443.

Parliament, E. (2024). *Meps adopt stricter co2 emissions targets for trucks and buses.* `https://www.europarl.europa.eu/news/en/press-room/20240408IPR20305/meps-adopt-stricter-co2-emissions-targets-for-trucks-and-buses?xtor=AD-78-[Social_share_buttons]-[linkedin]-[en]-[news]-[pressroom]-[strengthening-the-co2-emission-performance-targets]&`. (Accessed: 2024-19-04)

Pingen, G. L., van Ommeren, C. R., van Leeuwen, C. J., Fransen, R. W., Elfrink, T., de Vries, Y. C., ... Yorke-Smith, N. (2022). Talking trucks: Decentralized collaborative multi-agent order scheduling for self-organizing logistics. In *Proceedings of the international conference on automated planning and scheduling* (Vol. 32, pp. 480–489).

Raeesi, R., & Zografos, K. G. (2020). The electric vehicle routing problem with time windows and synchronised mobile battery swapping. *Transportation Research Part B: Methodological*, *140*, 101–129.

Weiss, M., Cloos, K. C., & Helmers, E. (2020). Energy efficiency trade-offs in small to large electric vehicles. *Environmental Sciences Europe*, *32*, 1–17.

Wen, M., Linde, E., Ropke, S., Mirchandani, P., & Larsen, A. (2016). An adaptive large neighborhood search heuristic for the electric vehicle scheduling problem. *Computers & Operations Research*, *76*, 73–83.

Wu, W., Lin, Y., Liu, R., & Jin, W. (2022). The multi-depot electric vehicle scheduling problem with power grid characteristics. *Transportation Research Part B: Methodological*, *155*, 322–347.

Xia, J., He, Z., Wang, S., Liu, S., & Zhang, S. (2024). Mean–standard-deviation-based electric vehicle routing problem with time windows using lagrangian relaxation and extended alternating direction method of multipliers-based decomposition algorithm. *Soft Computing*, 1–22.

Zsófia, P. (2021). *Electric trucks in europe to keep an eye on in 2021.* `https://trans.info/en/electric-trucks-in-europe-to-keep-an-eye-on-in-2021-244840`. (Accessed: 2024-16-04)

A Important variables in the data

In this subsection, the important variables from the dataset are explained. The variables and their description can be found in Table A.1. These variables are used in filling in the missing values.

Table A.1: Summary of variables

Variable	Description
`deadlineDateTime`	Variable representing the deadline of a request
`Pickup date`	The earliest date and time at which the request container can be picked up from the depot.
`Inland time`	The date and time at which the container is processed at the inland terminal/depot
`Goods Issued date time`	The date and time at which the container left the depot
`Arrival time`	The date and time at which the container arrived at the depot
`Import goods issued`	The date and time at which the container was picked up to be transported to the depot

B Missing values

In Figure B.1, the procedure to handle missing values is given. The variable descriptions can be found in Section A.

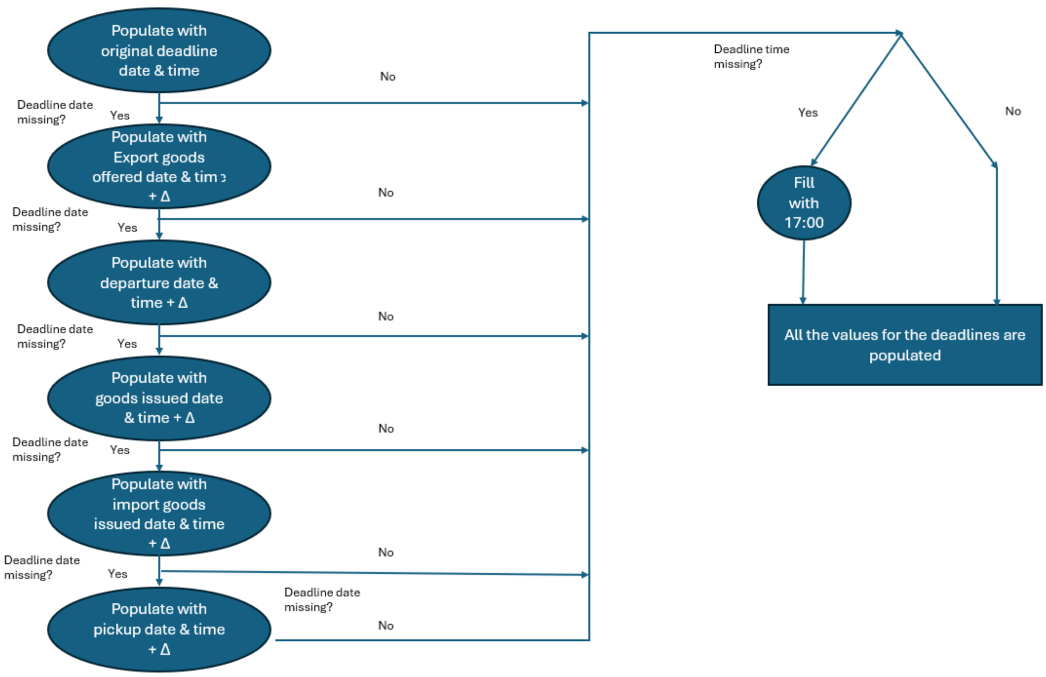

Figure B.1: Procedure to fill in missing values for the deadlines

C Code description

In the ZIP file 'MathematicalFormulationSmallInstanceCorrect.ZIP' several Java files are stored.

- The Main file contains the initialization of the RMP, the column generation algorithm, the truncated price-and-branch algorithm and the truncated column generation algorithm
- The MIPArcs file is used to solve the exact formulation
- The dataReaderArcs file is used to read the data
- The RMP file is used to solve the RMP
- The RMPMip file is used for the truncated price-and-branch algorithm
- The PricingProblem file is used to solve the Multi Label Shortest Path algorithm
- The GreedyPricingProblem file is used to generate a feasible initial solution
- The RouteInfo file is used to retrieve information on a route

If the Main file is now run, Instance 3 will be solved for the LP relaxation, the truncated price-and-branch algorithm and the truncated column generation (in that order). The RMP is initialized with a feasible set of routes.