Erasmus University Rotterdam
Erasmus School of Economics
Master Thesis Econometrics & Management Science

A column generation-based heuristic to check the feasibility of using electric buses instead of petrol buses

Name Student: Wessel Verhagen
Student ID number: 450866

Supervisor: Twan Dollevoet
Second assessor: Wilco van den Heuvel

February 27, 2023

Abstract

The goal of this paper is to evaluate the feasibility of using electric buses instead of petrol buses. To do this, a column generation-based heuristic is developed to schedule electric and petrol buses for a given bus timetable. This algorithm can be used to determine how many extra buses and costs are required if petrol buses were replaced with electric buses, as well as the number of recharge stations and their capacities needed to efficiently execute the bus timetable. Buses can start from multiple depots and must complete all trips in the bus timetable. No range constraint is imposed for petrol buses, and the costs consist of a fixed cost and a variable cost based on the distance traveled.

For electric buses, a range constraint is incorporated and electric buses can recharge their batteries at charging stations with limited capacity. As few assumptions as necessary are made to reflect real-world conditions as closely as possible. Therefore, non-linear recharging and partial recharging of electric buses are allowed. The cost of electric buses also includes a fixed cost and a variable cost based on distance traveled, as well as a cost for plugging the bus into a charger.

The algorithms were implemented in Python and tested on the data of the Vechtstreken region in the Netherlands for the bus company Transdev. The results indicate that from a logistical point of view, it is possible to schedule electric buses instead of petrol buses but it will require a few additional buses.

Contents

1 Introduction

Over the last few decades, the transportation sector shifted significantly towards sustainability. Supported by government regulations, citizens are encouraged to change their conventional petrol-driven vehicles to electric vehicles. On top of that, in some European countries, public transport became cheaper to encourage citizens to move towards public transport. However, simply shifting citizens towards using public transport is not sufficient to address the issue of emissions. In addition, public transport itself must innovate more sustainable solutions in order to reduce the emission of greenhouse gasses. One way to do so would be to change conventional petrol buses to electric buses.

Although the shift to electrifying buses has its upside of being more environmentally friendly, logistically it imposes challenges. One of these challenges is that the bus schedule of electric buses will be different and more complicated. It will be impossible to execute the bus schedule of petrol buses with electric buses as their range is limited by their battery. Until now, the range of electric buses is limited to 190-210 kilometers ("Electric bus range, focus on electricity consumption. A sum-up" (2022)).

Meeting this additional range requirement that is incorporated by using electric buses gives rise to the following research questions: How many electric buses are needed to execute the bus timetable? Where should recharge stations be located and how many? Which bus recharges its battery at what time, and at which recharge station?
These questions will be investigated to answer the central research question of this paper:

Given a set of bus trips, what are the 'extra' costs and corresponding bus schedules when using electric buses instead of petrol buses?

To answer the research question, first, an optimization algorithm for allocating petrol buses to trips is created. The algorithm will ensure that each bus trip in the bus timetable is executed. On top of that, the algorithm takes into account that each bus depot has a limited capacity. In literature, this problem is called the Multiple-Depot Vehicle Scheduling Problem (MDVSP). After developing the algorithm for petrol buses, the algorithm will be extended and the same set of trips will be allocated to electric buses. With electric buses, a limited range constraint is incorporated into the algorithm. The bus will start the day at a depot with a fully charged battery. During the day the bus will need to recharge its battery to keep enough charge. In between trips, a bus could fully or partly recharge its battery at recharge stations that have a limited capacity. Important to mention is that the recharging does not have to be linear. The more power a battery has, the slower it recharges. The problem in which electric buses are allocated to trips is called the Multiple-Depot Electric Vehicle Scheduling Problem (MDEVSP). The two algorithms can be used to calculate the cost of using electric buses as well as petrol buses to check the feasibility of using electric buses. But besides, the algorithm for electric buses can also be used to determine the number of recharge stations with their corresponding capacity to execute a bus timetable efficiently. In both algorithms, the least amount of assumptions is made to keep the problems as close to reality as possible. Both of the resulting problems are *NP-Hard* which is why this paper will elaborate on a column generation-based heuristic to find a lower bound as well as an integer heuristic solution.

The remainder of this paper is organized as follows: in section 2 a detailed description of the problem will be given. Section 3 will elaborate on the relevant literature which is available on both the MDVSP as well as the MDEVSP. Section 4 will elaborate on how the MDVSP will be solved.

A mathematical formulation will be given and a column generation-based heuristic will be used to get a feasible solution. In section 5 the same will be done for the MDEVSP. In section 6 the results will be given. Finally, in section 7 and section 8 the conclusion and the discussion will be presented.

2 Problem description

In this chapter, a detailed description of both the MDVSP and the MDEVSP is given. The constraints as well as the objective will be clarified in detail and all assumptions will be explained.

The main objective of this paper is to check the feasibility of using electric buses instead of conventional petrol buses. For this research, an open-source OV data set containing a timetable of bus trips, a set of bus stations, and a set of bus depots is used. Data about the bus schedule is not available because it is not yet determined which bus is executing which bus trips. To check the feasibility of using electric buses an algorithm will be developed that can generate bus routes that execute all trips in the bus timetable for both petrol and electric buses. A route is a sequence of bus trips that are executed sequentially by the same bus.

In this paper, the bus timetable of *Transdev* is obtained from the open-source OV data set. The bus schedules of these bus timetables will be generated and optimized for both petrol and electric buses on different days. There is a difference in the timetable between weekdays and weekend days. On weekdays 744 trips are scheduled, whilst on Saturday and Sunday, 533 and 382 are scheduled respectively. These trips will follow different bus lines. The bus lines of Transdev are visualized in Figure 1.

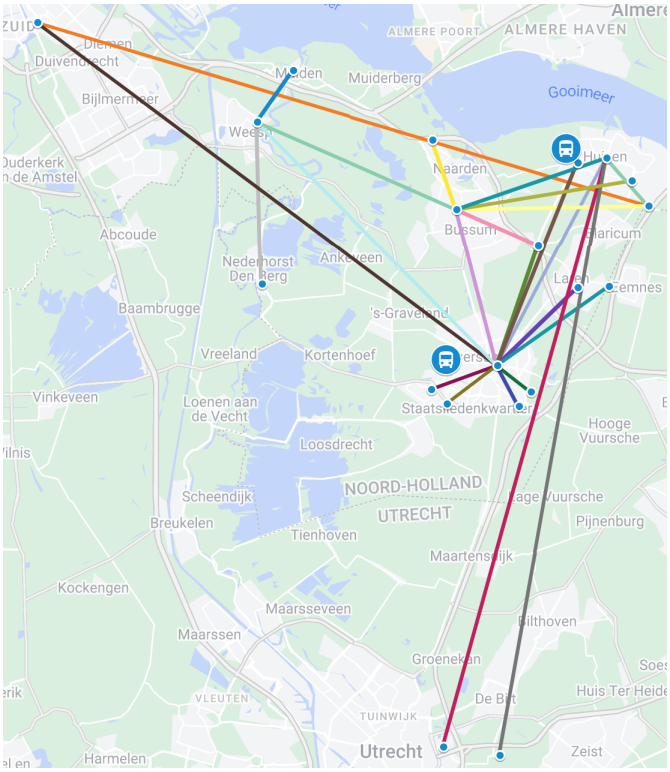


Figure 1: Bus lines and bus depots of Transdev.

Each bus line has a starting station l^s and an ending station l^e, these stations are depicted by a blue dot in Figure 1. The bus lines are depicted by a straight line between l^s and l^e. In reality, the bus will follow another route to get from l^s to l^e but for optimizing the bus schedule it does not matter how the bus drives in between these stations. Only the real travel distance and the time of the trips are essential. In Figure 1 the bus depots d are visualized by bus signs. These bus depots have been placed in logical locations, as the information about the location of these depots was missing in the data. At the start of the day, buses will leave these depots, and at the end of the day, the buses will return to the same depots as where they started. Each depot will have its corresponding capacity. Trip A is defined with a starting station l^s_A and an ending station l^e_A. These stations are equal to the starting location and ending location of a bus line. In addition, a trip also has a starting time t^s_A, an ending time t^e_A, and a distance dis_A.

2.1 Multiple Depot Vehicle Scheduling Problem

The first goal of this paper, is to construct an optimization algorithm that allocates petrol buses to routes, such that the total costs are minimized and all trips in the timetable are executed. At the start of the day, buses are stationed at multiple bus depots and need to be allocated to trips. One bus can do multiple bus trips in a sequence if these trips are compatible. Define ϕ_{ij} as the deadhead

time which is the time of driving an empty bus from station i to station j. Then two bus trips A and B are compatible if $l_A^e = l_B^s$ and $t_A^e < t_B^s$ or $l_A^e \neq l_B^s$ and $t_A^e + \phi_{ij} < t_B^s$. A sequence of compatible bus trips conducted by the same bus, starting and ending at the same depot, is called a bus route. The cost of a bus route will consist of a fixed part and a variable part which depends on the distance of the route. The algorithm will search for the best routes taking the following constraints into account. First, all bus trips need to be covered. The bus timetable is leading and each trip needs to be executed. Secondly, there are multiple bus depots with a limited bus capacity, meaning that a limited number of bus routes can start from the same depot. Lastly, it is assumed that buses are starting from the depot with a full tank which lasts for the rest of the day. The problem to allocate buses from multiple depots to trips is proven to be a *NP-hard* problem by Bertossi et al. (1987). Due to the problem's complexity, the problem can only be solved to optimality for small instances. This paper aims to solve the problem for a large instance. Therefore, a heuristic is constructed to solve this problem.

2.2 Multiple Depot Electric Vehicle Scheduling Problem

The MDVSP will be extended so that electric buses are allocated to the same trips. The cost of electric bus routes consists of a fixed cost of taking an electric bus out of the depot, a variable cost depending on the distance traveled plus a fixed cost of plugging the bus into a charger. The main difference in allocating electric buses to trips is that it has to be taken into account that electric buses are limited in their range by their battery capacity. A bus will be fully charged at the start of the day and will be able to recharge at recharge stations. These recharge stations can be located at the same place as where bus lines are starting, but can also be located at any other location. These recharge stations have a limited number of chargers. A bus can recharge fully or partly at these recharge stations. The recharging places, bus depots, and bus lines are visualized in Figure 2. In this figure, the bus depots and recharge stations have been placed at logical locations, as the data about their locations were missing. In this research, it will be assumed that electric buses have a range of 210 kilometers. At least 10 kilometers of the battery range needs to remain at any point in time. This is to prevent the bus from being out of charge at any given moment as the range of a bus is stochastic in reality. Therefore, the algorithm considers a range of 200 kilometers after subtracting 10 kilometers from the range. In this research, the recharging of the battery does not have to be linear and the recharging process of this paper is visualized in Figure 3. The recharging of the bus is determined using the following formula: $range(m) = \sqrt{time(s)/total\ recharge\ time(s)} * total\ range(m)$. In this paper, the *total range* is set to 200000 meters and the *total recharge time* is set to 2100 seconds, which equals 35 minutes. Using this formula, the range of an electric bus can be determined after recharging for a specific time. First, the range at the start of recharging is used to determine the starting time of recharging. Next, the recharge time is added to the starting time of recharging to determine the range after recharging. This recharge process is fictional but for the algorithm, any recharging scheme can be implemented.

Concluding, the aim of this paper is to allocate both petrol and electric buses to bus trips taking as few assumptions as possible to bridge the gap between theory and practice. The model will strive to stay as close to reality as possible in order to make it applicable for use by various bus companies in allocating buses to trips and to evaluate the feasibility of replacing petrol buses with electric buses from a logistical perspective.

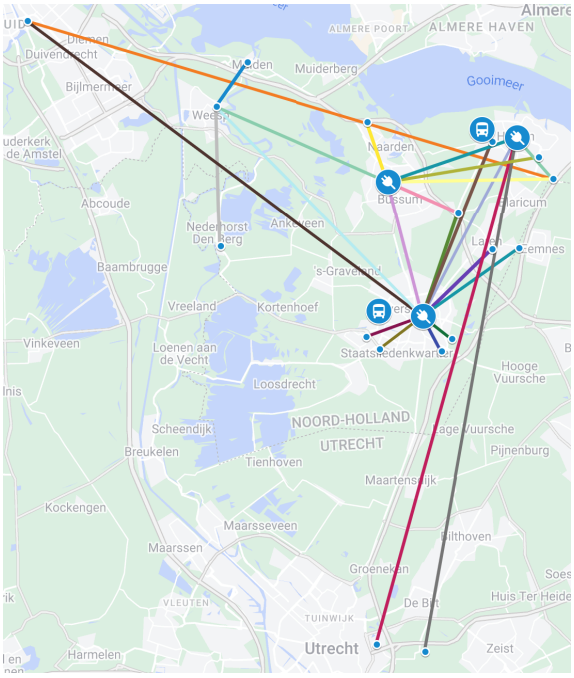


Figure 2: Bus lines, bus depots and recharging stations of Transdev.

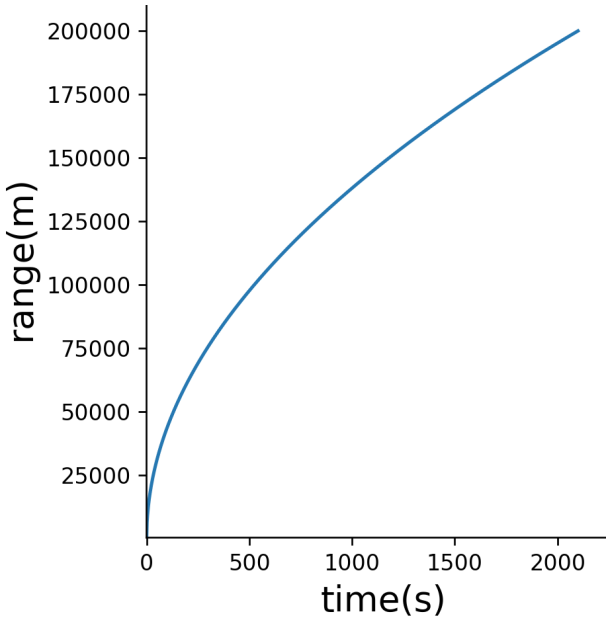


Figure 3: Recharging process.

3 Literature review

In this chapter, the relevant literature on the MDVSP and MDEVSP is reviewed. First, the conventional problem of allocating petrol buses to bus trips is discussed. To continue, the related literature of the MDEVSP will be discussed, incorporating a range constraint for electric buses as well as a capacity constraint at recharge stations.

3.1 Multiple Depot Vehicle Scheduling Problem

There are different ways of formulating the MDVSP. One way of formulating the problem is using a multi-commodity network flow formulation which is used in Forbes et al. (1994). The authors formulate the problem in a graph in which trips and depots are represented by vertices and connections of trips are represented by arcs. This formulation of the problem is straightforward but since each arc is a trip-to-trip connection the problem requires a large number of variables.

Another way of formulating the problem is formulating the problem as a set covering problem described in Bianco et al. (1994). The algorithm will pick the cheapest set of routes that will cover all trips in the bus timetable. The set covering formulation requires the set of all possible routes which is impractical to enumerate. Therefore, column generation heuristics are often used to solve this problem because they do not consider the entire set of routes, but rather generate routes during the optimization process.

The performance of five different algorithms to solve large instances of the MDVSP is researched in Pepin et al. (2009). This paper compares the following 5 techniques: a branch-and-cut method, a Large neighborhood search, a Tabu search, a Lagrangian heuristic, and a Truncated column generation. From this paper, for a large instance, the truncated column generation algorithm has the best performance if enough computational time is available. The large neighborhood search is the best alternative if less computational time is available.

A truncated column generation is a heuristic approach based on the branch and price algorithm. A branch and price algorithm is a branch and bound algorithm in which the LP relaxation is solved by column generation. The MDVSP is exactly solved using the branch and price algorithm by Ribeiro & Soumis (1994). The truncated column generation algorithm is similar to the proposed technique of Desaulniers et al. (1998) in which a variable is set to 1 if it has a high value in the LP relaxation.

In the literature, there are many more papers that are using a column generation-based heuristic to solve the MDVSP and related problems. The column generation heuristics seems to be a promising way of solving both vehicle/crew scheduling problems for large instances. For example, a truncated column generation is used for solving the MDVSP where heterogeneous buses are available by Guedes & Borenstein (2015). Heterogeneous buses do not include electric buses in this paper but buses of different sizes.

3.2 Multipe Depot Electric Vehicle Scheduling Problem

The MDEVSP is an extension of the MDVSP and is thereby proven to be a *NP-Hard* problem. With the rising interest in electric buses around the world, the MDEVSP has attracted increasing attention from researchers. Many papers are investing the problem by taking different assumptions. In this paper the following characteristics are added to the MDVSP:

- Buses are fully charged at a depot but regularly the bus will need to recharge at recharge stations as their range is limited by their battery capacity.
- Recharging stations with limited capacity are located nearby or at bus stations.
- Charging is not linear and any given recharging function can be used in the algorithm.

The goal of this paper is to solve large instances of the MDEVSP. Therefore, the problem will be solved heuristically. The most common ways to generate a feasible heuristic solution in literature are metaheuristics or column generation-based heuristics. In the column generation-based heuristic, the range constraint and the non-linearity of recharging are added to the pricing problem while the limited capacity constraint of recharge stations is added to the restricted master problem. Houwelingen (2018) solves the problem using a column generation heuristic without the limited charging capacity and uses an approximation algorithm to solve the pricing problem. Wang et al. (2021) is optimizing the same problem as Houwelingen (2018) using column generation in combination with a genetic algorithm. The column generation is used to generate a set of routes while the genetic algorithm selects the routes from the generated set to construct a feasible integer solution. Wu et al. (2022) introduces a column generation heuristic with recharging time windows to incorporate the limited capacity constraint of the charging stations. Partly charging is not incorporated in the paper and if a bus recharges, it can only fully recharge itself. A Multi Label Setting Algorithm is used to solve the pricing problem which seems to be promising. Lastly, the

paper of de Vos et al. (2022) solves the full problem using a column generation-based heuristic. This paper allows non-linear recharging, partly charging, and a limited capacity at recharge stations.

4 Methodology Multiple Depot Vehicle Scheduling Problem

In this chapter, the method for heuristically solving the problem described in subsection 2.1 is explained. The MDVSP is known to be a NP-hard problem (Bertossi et al. (1987)) and can only be solved to optimality, in a reasonable amount of time, for small instances. This paper will use a column generation-based heuristic to solve the problem so that it can find solutions for both small and large instances. First, a mathematical formulation is given, and thereafter the column generation-based heuristic is explained.

4.1 Mathematical formulation

There are several ways of mathematically formulating the MDVSP. In this paper, two main formulations of the MDVSP will be given. First, the multi-commodity network flow formulation will be explained. This is a formulation that is straightforward and is used to solve the problem to optimality. Thereafter, the set covering formulation is given. This formulation is used in the column generation algorithm.

4.1.1 Multi-commodity network flow formulation

The multi-commodity network flow formulation is a straightforward formulation that is easy to understand and implement. However, the drawback of the formulation is that it is difficult to incorporate additional constraints. For example, it is not possible to incorporate the range constraint considered later in this paper. Additionally, the algorithm uses a lot of variables, which makes it impossible to solve large instances in a reasonable amount of time.

For each depot d a graph is created $G^d = (V^d, A^d)$. The set V^d corresponds with vertices which consist of the *source*, and the *sink*, which are both depot d, and all the trips twice. The set of trips is duplicated; the first set is called T_{start} and the other set of trips is called T_{end}. The set A^d corresponds with the set of arcs and consists of the arcs from the source to each compatible trip in T_{start}, all the arcs between compatible trips from T_{end} to T_{start}, all arcs between the same trip from T_{start} to T_{end}, define this set as AT, all compatible arcs between T_{end} and the *sink* and an arc from the *sink* to the *source* define this as set AS.

Then define T as the set of all trips, cap^d as the capacity of depot d, $c^d_{i,j}$ as the cost of arc A^d_{ij} which is 0 if $A_{i,j} \in \{AT, AS\}$ and otherwise it is the cost of performing trip j after trip i and x^d_{ij} as the flow over arc A^d_{ij}. Then the problem can be written in the following integer linear program:

$$\min \sum_{d \in D} \sum_{(i,j) \in A^d} c^d_{ij} x^d_{ij} \tag{1}$$

$$\sum_{j \in V^d:(i,j) \in A^d} x^d_{ij} = \sum_{j \in V^d:(j,i) \in A^d} x^d_{ji} \qquad \forall d \in D, \forall i \in V^d \tag{2}$$

$$\sum_{d \in D} \sum_{(i,j) \in AT^t} x^d_{ij} = 1 \qquad \forall t \in T \tag{3}$$

$$\sum_{(i,j)\in AS^d} x_{ij}^d \leq cap^d \qquad \forall d \in D \tag{4}$$

$$x_{i,j} \in \{0,1\} \qquad \forall (i,j) \in A^d \setminus AS^d, \forall d \in D \tag{5}$$

$$x_{i,j} \in \mathbb{N}^+ \qquad \forall (i,j) \in AS^d, \forall d \in D \tag{6}$$

The objective (1) ensures that the total cost is minimized. Constraint (2) ensures that, if a bus is driving a trip, it will continue its route from the end station of the previous trip, ensuring the route is feasible. Constraint (3) ensures that all bus trips are executed by one bus. Constraint (4) ensures that the number of buses leaving a depot does not exceed its capacity. The last constraints (5,6) ensure that the solution is an integer.

4.1.2 Set covering formulation

Another mathematical way of formulating the MDVSP with petrol buses is formulating the problem as a set covering problem. Define T as the set of all possible trips, and R as a set of all possible routes. Remind that a route is a sequence of compatible trips that can be assigned to one bus. Define c_v as the variable cost of driving a kilometer, dis_r as the distance of route r, and c_f as the fixed cost of taking a bus out of the depot. Then c_r, the cost of route r is defined as $c_r = dis_r * c_v + c_f$. Define D as a set of all depots, cap_d as the capacity of depot d, R_d as the set of all feasible routes starting and ending at depot d, x_r as a binary variable which is 1 if route r is chosen and 0 otherwise. Finally, define a_{tr} as an indicator parameter that indicates whether trip t is included in route r. Then the problem can be written as the following integer linear program (ILP):

$$\min \sum_{d\in D} \sum_{r\in R_d} c_r x_r \tag{7}$$

$$\sum_{d\in D} \sum_{r\in R_d} a_{tr} x_r = 1 \qquad \forall t \in T \tag{8}$$

$$\sum_{r\in R_d} x_r \leq cap_d \qquad \forall d \in D \tag{9}$$

$$x_r \in \{0,1\} \qquad \forall d \in D, \forall r \in R_d \tag{10}$$

The objective (7) ensures the total cost of all routes is minimized. Constraint (8) ensures that all bus trips are executed by one bus. Constraint (9) ensures that no more buses are leaving a depot than its capacity. The last constraint (10) ensures the solution is binary.

4.2 Column generation heuristic

A common heuristic for solving an MDVSP is using a column generation-based heuristic. The column generation algorithm will solve the LP relaxation of the ILP. The column generation algorithm consists of a restricted master problem and a pricing problem. The restricted master problem solves the LP relaxation with a selected set of routes while the pricing problem updates this set.

At the start, the algorithm solves the restricted master problem with an initial set of routes. This set of routes can be any set of routes, and it does not matter if the initial set of routes is inefficient. The only requirement is that a feasible solution can be obtained from the initial set of routes. Thereafter, the pricing problem is solved. The pricing problem searches for routes with negative reduced costs. Routes with negative reduced costs are added to the set of routes in the restricted master problem. The restricted master problem will be resolved, including the extra routes obtained by the pricing problem. This process will continue until the pricing problem is unable to generate routes with negative reduced costs. If there are no routes with negative reduced costs, the solution of the restricted master problem is the optimal solution of the LP relaxation. The solution found for the LP relaxation will be used to produce the final heuristic integer solution of the ILP.

4.2.1 Restricted master problem

For the restricted master problem, the binary constraint (10) is relaxed and the equal sign of constraint (8) is changed to a greater or equal sign. This LP relaxation will be solved to optimality using Gurobi. Important to notice is that the restricted master problem does not include all possible routes but a limited set of routes. It starts with an initial solution and the pricing problem will search for routes to improve the solution. The restricted master problem of the column generation algorithm is as follows:

$$\min \sum_{d \in D} \sum_{r \in R_d} c_r x_r \tag{11}$$

$$\sum_{d \in D} \sum_{r \in R_d} a_{tr} x_r \geq 1 \qquad \forall t \in T \tag{12}$$

$$\sum_{r \in R_d} x_r \leq cap_d \qquad \forall d \in D \tag{13}$$

$$x_r \geq 0 \qquad \forall d \in D, \forall r \in R_d \tag{14}$$

To start the heuristic an initial set of routes is added to the restricted master problem which can generate a feasible solution. With the MDVSP a feasible starting solution is executing each bus trip by a bus from a virtual bus depot. The cost of using this virtual bus depot will be high enough to ensure that it will not be in the optimal solution if enough buses are available at the real depots. If there are not enough buses and the virtual bus depot is used in the optimal solution, then the problem is infeasible, and more buses at the depots are needed to execute the bus timetable. The next step is to solve the pricing problem to find new routes which are then added to the set R of the restricted master problem to improve the solution.

4.2.2 Pricing problem

The aim of the pricing problem is to find routes with negative reduced costs. These routes will be added to the restricted master problem and will improve the solution. For a detailed derivation of the pricing problem we refer to Desrosiers & Lübbecke (2005). The pricing problem for the MDVSP is equivalent to the shortest path problem in a directed graph. For each depot, a new directed graph is created. The graph G^d=(V^d,A^d) defines the graph of depot d. V^d corresponds with the vertices in the graph which correspond to all trips which can be executed from depot d, a source so, and a sink si which are the same depot d. A^d corresponds with the arcs which are weighted with the reduced costs. Each directed graph consists of all trips placed in chronological order and arcs connect two trips if the trips are compatible. From each trip vertex in the directed graph, there is also an arc from the source vertex and to the sink vertex. The shortest path problem could be solved using a dynamic programming algorithm. Define c_{ij} as the cost of $arc(i,j)$ which equals the cost of performing trip j after trip i and is equal to:

$$c_{ij} = (\rho_{ij} + dis_j) * c_v$$

Here ρ_{ij} is the deadhead distance of trip i to trip j, c_v is the variable cost of the bus, and dis_j is the distance of trip j. Besides these parameters, define λ_j as the dual of constraint (12) for each trip, and μ_d as the dual of constraint (13) for each depot. Lastly, define x^d_{ij} as the decision variable of $arc(i,j)$ for the graph of depot d which is 1 if $arc(i,j)$ is selected and 0 otherwise. Then the pricing problem for depot d is:

$$\min \mu_d + \sum_{(i,j)\in A^d} (c_{ij} - \lambda_j) x^d_{ij} \tag{15}$$

$$\sum_{i\in V^d} x^d_{ij} = \sum_{i\in V^d} x^d_{ji} \qquad \forall j \in T \tag{16}$$

$$\sum_{i\in T} x^d_{so,i} = 1 \tag{17}$$

$$\sum_{i\in T} x^d_{i,si} = 1 \tag{18}$$

$$x^d_{ij} \in \{0,1\} \qquad \forall (i,j) \in A^d \tag{19}$$

The objective (15) searches for the shortest path in G^d. Constraint (16) ensures that if a bus is driving a trip, it will proceed its route from the end station of the trip. Constraint (17,18) ensures that each route starts and ends from the same depot. Lastly, constraint (19) ensures that the solution is binary.

To clarify, the pricing problem of a graph G^d is visualized in Figure 4 and is equivalent to the shortest path problem in this graph. The graph G^d consists of 4 trips. There is an arc from the source vertex to each trip vertex, an arc between two compatible trips, and an arc from each trip

vertex back to the sink vertex. The corresponding reduced cost of a route is equal to the distance of the path in the graph if each arc(i,j) of graph G^d is assigned the following weight w_{ij}:

$$w_{ij} = \begin{cases} c_f + (\rho_{ij} + dis_j) * c_v + \mu_d - \lambda_j, & \text{if } i = so \\ \rho_{ij} * c_v, & \text{else if } j = si \\ (\rho_{ij} + dis_j) * c_v - \lambda_j, & \text{else if } i, j \in T \end{cases}$$

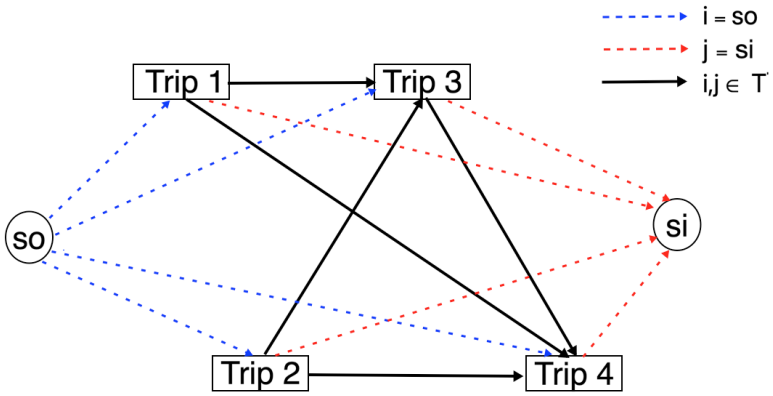


Figure 4: Pricing problem visualization.

4.2.3 Integer solution

The column generation algorithm described above will solve the LP-relaxation of the ILP. The solution of the column generation will be a lower bound of the ILP and can be used to compare the solution of the ILP. To construct an integer solution, the solution of the LP relaxation is used. From the solution, a chosen number of *good routes* will be set to 1. Thereafter, the LP relaxation is solved again without all trips which are included in a route that is selected and fixed to 1. This procedure will continue until no more trips are left and an integer solution is found. Different methods to select *good routes* will be tested but the selection criteria will be mainly based on the solution value of the LP relaxation.

5 Methodology Multiple Depot Electric Vehicle Scheduling Problem

This chapter explains the method used to solve the MDEVSP. The goal is to schedule electric buses in real-time schedules, primarily for large instances. Until now it is impossible to solve large instances of *NP-hard* problems to optimality in a reasonable amount of time. To find a solution to this problem, a column generation-based heuristic is constructed.

5.1 Mathematical Formulation

To solve and formulate the MDEVSP, charging windows are introduced which represent recharging activities. These recharging windows split the planning horizon into recharge time windows with the same lengths in which a bus can recharge. These recharge time windows have a starting time t^s, ending time t^e, starting recharge station l^s, and ending recharge station l^e. This is similar to a trip with three differences: for a recharge window, the starting recharge station is equal to the ending recharge station $l^s = l^e$. Secondly, a trip will have a negative impact on the battery level while a charging window has a positive impact on the battery level. Lastly, $t^e - t^s$ will for each recharge time window be equal while for trips this is not equal. For each time interval and for each recharge station, a recharge time window is created. Define R again as the set with all possible routes. Notice that this set of routes is different from the set of routes at the MDVSP as the range constraint needs to be incorporated. Define S as the set of recharging stations, W_s as the set of time windows for station s, C_s as the recharge capacity of recharge station s, and d_{swr} as an indicator parameter which indicates whether time window w of recharge station s is included in route r. The parameters and variables D, x_r, and a_{tr} are the same as in the MDVSP. The cost of a route c_r will change from the MDVSP and will depend on the distance of the route, the number of times it recharges, and a fixed cost. Then the integer linear program for electric buses (ILPE) of the MDEVSP is:

$$\min \sum_{d \in D} \sum_{r \in R_d} c_r x_r \tag{20}$$

$$\sum_{d \in D} \sum_{r \in R_d} a_{tr} x_r = 1 \qquad \forall t \in T \tag{21}$$

$$\sum_{r \in R_d} x_r \leq cap_d \qquad \forall d \in D \tag{22}$$

$$\sum_{d \in D} \sum_{r \in R_d} d_{swr} x_r \leq C_s \qquad \forall s \in S, \forall w \in W_s \tag{23}$$

$$x_r \in \{0, 1\} \qquad \forall d \in D, \forall r \in R_d \tag{24}$$

The objective (20) ensures the total cost of all routes is minimized. Constraint (21) ensures that all bus lines are executed by one bus. Constraint (22) ensures that no more buses are leaving a depot than its capacity. Constraint (23) guarantees that the number of buses recharging at any given time does not exceed the capacity of the recharge stations. Lastly, constraint (24) ensures the solution is binary.

5.2 Column generation heuristic

The heuristic used to solve the MDEVSP is again based on the solution of the LP relaxation solved by a column generation algorithm. The column generation algorithm is the same as described in subsection 4.2 only the master problem and the pricing problem will differ due to the added constraint of the range of electric buses.

5.2.1 Restricted master problem

The restricted master problem of the column generation algorithm is the same as the ILPE despite the equal sign of constraint (21) being changed to a greater or equal sign and the integrality constraint (24) being relaxed. The restricted master problem of the MDEVSP is:

$$\min \sum_{d \in D} \sum_{r \in R_d} c_r x_r \tag{25}$$

$$\sum_{d \in D} \sum_{r \in R_d} a_{tr} x_r \geq 1 \qquad \forall t \in T \tag{26}$$

$$\sum_{r \in R_d} x_r \leq cap_d \qquad \forall d \in D \tag{27}$$

$$\sum_{d \in D} \sum_{r \in R_d} d_{swr} x_r \leq C_s \qquad \forall s \in S, \forall w \in W_s \tag{28}$$

$$x_r \geq 0 \qquad \forall d \in D, \forall r \in R_d \tag{29}$$

For the restricted master problem in the column generation algorithm, not all the routes are added to the set R. This prevents us from calculating all possible routes which require heavy calculations. At the start of the algorithm, the restricted master problem starts with an initial solution. An initial solution for the MDEVSP is to place virtual bus depots with unlimited capacity at each starting station of a trip. Perform each trip with a new bus from the depot at the start of the trip and drive directly back to the depot. The cost of using a virtual depot is sufficiently high that it should never be part of the optimal solution. If a bus from the virtual depot is used in the optimal solution the problem is infeasible and more buses are required at the real depots to execute all bus trips in the bus schedule. After solving the restricted master problem the pricing problem is solved to update the set of routes.

5.2.2 Pricing problem

The pricing problem of the MDEVSP is more complicated than the pricing problem of the MDVSP because it has to incorporate the range constraint while searching for new routes with negative reduced costs. The reduced cost of a route can be calculated using equation (30). Define λ_t as the dual of constraint (26) for each trip t, and μ_r as the dual of constraint (27) for the depot d used in route r, and σ_{sw} as the dual of constraint (28) for each recharge time window w for each recharge station s.

$$rc_r = c_r + \sum_{s \in S} \sum_{w \in W_s} d_{swr} * \sigma_{sw} + \mu_r - \sum_{t \in T} a_{tr} * \lambda_t \tag{30}$$

The pricing problem of the MDEVSP is equivalent to an Elementary Shortest Path Problem with Resource Constraints (ESPPRC). The graph of the MDEVSP $G^d = (V^d, A^d)$ is a directed graph with all vertices V^d placed in chronological order and arcs A^d double weighted. The set V^d consists of all trips which are possible to execute from depot d, all recharge time windows for

each recharge station, and a source so and a sink si which are the same depot d. The set A^d consists of all arcs between compatible vertices. This includes arcs from each trip to any other compatible trip, from each trip to any other compatible recharge time window, from each recharge time window to any other compatible recharge time window, from each recharge time window to any other compatible trip, from the source to any trip which is accessible from depot d, and each trip back to the sink. In order to decrease computation time, the set does not include arcs from a recharge window to the sink. This is because it is assumed that a bus will not recharge before it returns to the depot as it can recharge at the depot. The arcs A^d in the graph are double weighted. One weight will correspond with the reduced cost rc. The other weight pow will indicate how much power an arc reduces from the battery. Define c_{fe} as the fixed cost of using an electric bus, c_{ve} as the variable cost per kilometer of using an electric bus, c_{fp} as the fixed cost of connecting a bus to a charger, ρ_{ij} as the deadhead distance from vertex i to vertex j, dis_j as the distance of vertex j, μ_d as the dual of constraint(27) for each depot d, A as the set of all recharge time windows w together, σ_a as the dual of constraint(28) for each w in the set A. Then the rc of $arc(i,j)$ on graph G^d is:

$$rc_{ij} = \begin{cases} c_{fe} + (\rho_{ij} + dis_j) * c_{ve} + \mu_d - \lambda_j & \text{if } i = so \\ \rho_{ij} * c_{ve} & \text{else if } j = si \\ (\rho_{ij} + dis_j) * c_{ve} - \lambda_j & \text{else if } j \in T \\ c_{fp} + \rho_{ij} * c_{ve} + \sigma_j & \text{else if } i \in T \text{ and } j \in A \\ \rho_{ij} * c_{ve} + \sigma_j & \text{else if } i, j \in A \end{cases}$$

For $arc(i,j)$ the second weight pow_{ij} will indicate how much power is used if j is in T or D. The power gained can not be indicated if j is a recharging time window as the recharging is non-linear and the recharging will depend on the state of charge SoC of a bus at a specific point. Define p_u as the power used per kilometer. Then the pow_{ij} on G^d is:

$$pow_{ij} = \begin{cases} \rho_{ij} * p_u & \text{if } j = si \\ (dis_j + \rho_{ij}) * p_u & \text{else if } j \in T \end{cases}$$

An example of the pricing problem for graph G^d is visualized in Figure 5. G_d consists of 4 trips and 1 recharge station in which two recharge time windows are available. There is an arc from the depot to each trip, between two compatible trips, from each trip to each compatible recharge time window, from each recharge time window to each compatible trip, between two compatible recharge time windows, and from each trip back to the depot. If arcs are double-weighted as described above then the ESPPRC of graph G^d is equal to the pricing problem.

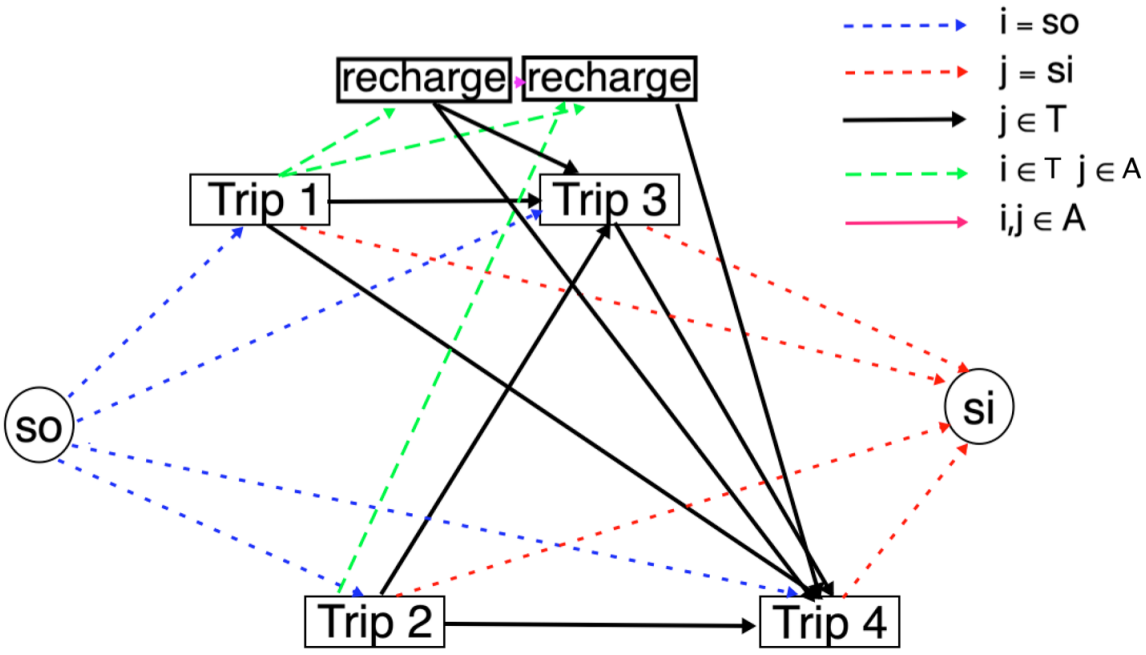


Figure 5: Pricing problem visualization.

5.2.3 Multi Label Setting Algorithm

To solve the ESPPRC a Multi-Label Setting Algorithm (MLSA) is used similarly as described in Wu et al. (2022). MLSA searches for the route with the lowest reduced cost while the state of charge of a bus will stay positive. The algorithm will work in a label-correcting way in which it places multiple labels on vertices. Using backtracking it will be possible to reconstruct the route of the label. The final label with the optimal solution will be the label at the sink with the lowest reduced cost.

In contrast with the dynamic programming algorithm of subsubsection 4.2.2 which only uses a single label for each vertex, this algorithm uses multiple labels. The label l_i^m is the m^{th} label of vertex i and will have the parameters (rc_i^m, SoC_i^m) the rc_i^m is the reduced cost of label m to get to vertex i and SoC_i^m will be the corresponding state of charge at the end of vertex i of label m. The labels are maintained by using forward dynamic programming. The label l_i^m of the vertex i will be extended to label l_j^m for vertex j over $arc(i,j)$ in the following way:

$$l_j^m = \begin{cases} None & \text{if } SoC_i^m - pow_{ij} \leq 0 \text{ and } j \in (T \cup D) \\ None & \text{else if } SoC_i^m - \rho_{ij} * p_u \leq 0 \text{ and } j \in A \\ (rc_i^m + rc_{ij}, SoC_i^m - pow_{ij}) & \text{else if } j \in (T \cup D) \\ (rc_i^m + rc_{ij}, Soc_i^m + f(SoC_i^m, \rho_{ij})) & \text{else if } j \in A \end{cases}$$

The function f is a function that calculates the SoC_j of a bus given the SoC_i and the deadhead distance of $arc(i,j)$ if a bus is recharging. This enables the algorithm to handle non-linear charging

schemes.

The efficiency of the MLSA heavily depends on the number of labels connected to each vertex. It is possible to eliminate a label of a vertex if it is dominated by another label. A label l_i^m is dominated by label l_i^n if:

$$SoC_i^m \geq SoC_i^n \text{ and } rc_i^m \leq rc_i^n$$

Using the dominance rule above the final solution will still be optimal and there is no loss of generality. The label l_i^m has more power left in the battery and on top of that a lower reduced cost than the label l_i^n. In both ways, this is better and the label l_i^n can be eliminated.

5.2.4 Integer solution

To get a feasible solution first the LP relaxation will be solved using column generation. The pricing problem and restricted master problem described above will be used in the same way as outlined in subsection 4.2. The solution of the LP relaxation will be a lower bound of the MDEVSP. The solution of the LP relaxation is most likely not an integer solution. However, in order to obtain an integer solution, variables with high values in the LP relaxation will be fixed to 1. The problem is then resolved using column generation while ignoring the fixed trips and decreasing the recharge capacity for the recharge windows used in those fixed routes. This process is repeated until the algorithm converges to an integer feasible solution.

5.3 Reducing computational time

The column generation algorithm described above is a common way to solve the MDEVSP but for some instances, it might require too much running time. In many ways, it is possible to reduce the computational time of a column generation algorithm and we will elaborate on some of them. The goal is to find a way to reduce the computational time while the effect on the solution quality is as small as possible. The different methods to reduce the computation time can be used separately or in combination but there will always be a trade-off between the solution quality and the computational time.

5.3.1 Stopping the column generation

In the LP-relaxation solved by a column generation algorithm, the pricing problem adds in each iteration routes with negative reduced costs until no more routes are found with negative reduced costs. In the first few iterations, the solution will improve significantly. At the start of the column generation algorithm, the solution is far from optimal and a lot of routes can be added to improve the solution. In contrast, iterations at the end of the algorithm hardly improve the solution as the solution is close to optimal. In the end, it takes a lot of iterations to find the optimal solution. Therefore, one way to speed up the algorithm is to terminate the algorithm before the solution of the LP relaxation is optimal. If the solution does not improve more than x percent over the last y iterations the algorithm will be terminated and the current solution will be considered optimal. The tuning parameter x is mostly in the order of 0.1 percent. This tuning parameter together with y is problem specific. Another way of terminating the column generation algorithm is to set a running time limit. The algorithm of this paper will terminate the LP-relaxation for petrol buses after one hour and for electric buses after six hours. For the integer solution, in which in each

iteration one bus is selected and fixed, the time limit is set for petrol buses to 30 minutes in the first iteration. After the first iteration, the algorithm will terminate after 15 minutes if there are more than 400 trips left to schedule and after five minutes if there are less than 400 trips left to schedule. For electric buses, the time limit is set to 3 hours and 20 minutes in the first iteration. After the first iteration, the algorithm will terminate after 30 minutes if there are more than 400 trips left to schedule and after ten minutes if there are less than 400 trips left to schedule.

5.3.2 Deleting routes

In the LP-relaxation solved by the column generation algorithm, routes are only added to the restricted master problem. After each iteration, the size of the restricted master problem increases. The main idea of deleting routes in the restricted master problem is to keep the restricted master problem manageable. After x iterations the algorithm will check whether the routes currently in the restricted master problem will still have negative reduced costs. If the route has positive reduced costs the route can be deleted from the restricted master problem without consequences. the parameter x is a tuning parameter and will be problem-specific. The algorithm in this paper will delete routes with positive reduced costs after 200 iterations if petrol buses are scheduled, and for electric buses, it will delete routes with positive reduced costs after 30 iterations.

5.3.3 Initial solution

One way to reduce the computation time which does not have an impact on the solution quality is changing the initial solution for different fixing iterations in the restricted master problem. After fixing routes in the column generation-based heuristic the algorithm starts the column generation over again with a new set of trips that needs to be scheduled. Instead of starting all over again, the routes of the last iteration can be used to speed up the algorithm. The initial solution for iteration n described in subsubsection 5.2.1 will be added to the routes of the solution in iteration $n-1$. From these routes, routes are deleted which contain trips that are fixed in iteration $n-1$. The column generation algorithm will speed up because the better the initial solution is, the faster the column generation converges to optimality.

5.3.4 Heuristic pricing problem

In the column generation algorithm, the pricing problem is always solved to optimality. Only the routes with the lowest reduced costs of each depot are added to the restricted master problem. This is not necessarily needed and adding the routes with the lowest reduced cost is not guaranteed to be the best strategy. The only requirement is that the routes which are added to the restricted master problem are having negative reduced costs. Besides, adding multiple routes in one iteration can speed up the algorithm because less iterations will be required to find the optimal solution. Theoretically, only in the last iteration, the pricing problem needs to be solved to optimality to prove that there is no more route with negative reduced costs.

A well-defined heuristic to solve the ESPPRC will find routes much faster than the MLSA. Especially, at the start of the algorithm, the chance of succeeding in finding routes with negative reduced costs will be high. An example of a heuristic to solve the ESPPRC before solving the problem using the MLSA is the following. From each vertex iterate over the outgoing arcs and if the battery is above x percent choose randomly one of the y outgoing arcs which have the lowest reduced costs. If the battery is below x percent the bus will consider the first z recharge time

windows of the q closest recharge stations and chooses the one with the lowest reduced costs. It will recharge until the battery is above sixty percent or the reduced cost of a recharge time window is p times higher than the last recharge window. For each graph, this heuristic is performed t times, and if more than u routes are found the MLSA will not be executed. Otherwise, the MLSA is executed after the heuristic. All parameters, x, y, z, p, t, and u are tuning parameters and are problem specific. In the algorithm used for this paper, the heuristic example was included, but after tuning the tuning parameters, no substantial improvement was observed and the heuristic was excluded from the optimization.

5.3.5 Deleting arcs in pricing problem

For the pricing problem described in subsubsection 5.2.2 each graph consists of all possible arcs that connect two compatible vertices. Reducing the number of arcs will speed up the computational time in finding the shortest path in a graph. Since, for each depot and in each iteration, the column generation algorithm needs to find the shortest path in a graph, reducing the number of arcs in each graph will reduce the computational time. Different restrictions can be introduced but the effect on the solution quality must be as small as possible. One way to reduce the number of arcs is only adding arcs to compatible vertices if the starting time of a compatible vertex is not more than x time units later than the ending time of the previous vertex. This prevents considering routes that are waiting for a long time between compatible vertices. By not considering routes where buses have to wait for a significant amount of time, the impact on the solution quality is expected to be minimal.

Another smart way of reducing the number of arcs in a graph of the pricing problem is deleting all arcs which include a deadhead distance of more than y kilometers. Taking these arcs out of the graph prevents the algorithm from considering routes that include long-deadhead distances. Driving long deadhead distances is likely, not optimal, therefore by removing these arcs the impact on the solution quality is expected to be minimal. The parameters x and y are tuning parameters and will be problem specific. When the pricing problem is solved heuristically in this paper, the x parameter is set to one hour. As a result, the algorithm will only consider trips and recharge windows if the start time is within one hour of the previous ending time. The main objective of the optimization of this paper is to reduce the number of buses required and not the length of the routes. Therefore, the arcs in the pricing problem are not reduced by the length of the deadhead distances and the y parameter is set to infinity.

6 Results

In this chapter, the results of the methods described in previous chapters are presented. The multi-commodity flow formulation given in subsubsection 4.1.1 is used to solve the MDVSP to an integer optimal solution. This solution is then used to evaluate the performance of the column generation algorithm. To evaluate its performance, the column generation algorithm will be used to solve the LP relaxation for petrol buses, as described in subsection 4.2. The column generation algorithm will then be used to generate an integer solution for the problem, which will be the bus schedule for petrol buses and is described in subsubsection 4.2.3.

Next, the same set of trips will be executed by electric buses. In this optimization, recharge stations with a limited capacity will be added at some of the bus stations. The LP relaxation will be solved using the column generation algorithm described in subsection 5.2. This will result in

a lower bound for the MDEVSP. The integer solution for the electric bus schedule will be based on the column generation algorithm, as described in subsubsection 5.2.4, resulting in the final bus schedule for electric buses.

Last, the algorithm will be used to determine the impact of using different types of electric buses. By varying their range and recharging speed, and calculating the effects of adding or removing recharge stations on the schedule.

6.1 Parameters

The aim of this paper is to research the consequences of incorporating the range constraint when using electric buses. To isolate the effect of the range constraint the fixed cost and the variable cost per kilometer will be equal for electric buses and petrol buses. The fixed cost of a bus will be €1000 and the variable cost will be €1 per kilometer. The fixed cost is significantly higher because the main goal is to minimize the number of buses rather than the total distance traveled by all buses.

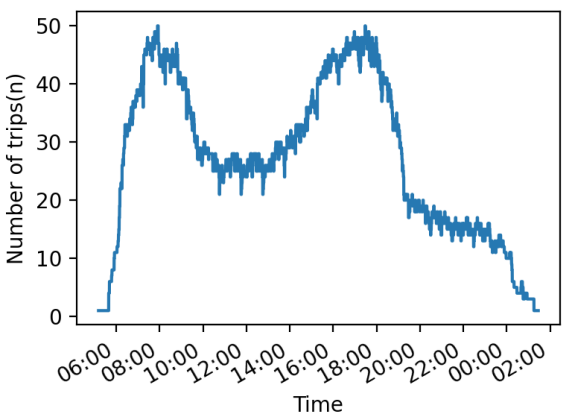
In the algorithm for the MDEVSP, there is an option to include a connection cost for plugging a bus into a charger. This cost can be used to prevent the algorithm from constantly recharging the bus and instead charge it for longer periods. To obtain a more realistic solution, the connection cost is set to €50 during the optimization. However, to fairly compare the solution, the connection cost is subtracted after the optimization. There is also an option to include a depot capacity constraint, but this constraint is disregarded as this data is not available in the open-source data set. To solve the MDVSP and the MDEVSP, information about the deadhead distances between stations is necessary. In this study, the deadhead distance between two stations is defined as the Euclidean distance multiplied by 1.3. To calculate the deadhead time between two stations, the distance is used in combination with an average empty driving speed of 14 m/s.

When optimizing electric buses, we need to specify additional parameters. First, the range of an electric bus is set to 200 kilometers. Secondly, the recharging speed of an electric bus is non-linear and is shown in Figure 3. Finally, the length of each recharge window is set to 300 seconds. This means that a bus will recharge for at least 300 seconds and then, after 300 seconds, it will decide whether to start driving again or to continue recharging for another 300 seconds.

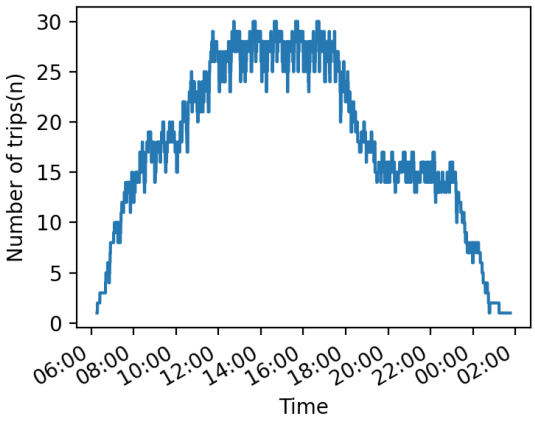
6.2 Data

The optimization algorithms are implemented in Python and tested on the data of *Transdev*. *Transdev* is a bus company in the Vechstreken of the Netherlands. Their data is retrieved from an open-source OV data set which contains the data of the trips driven in 2021. The data set consists of bus lines between 33 different bus stations. On weekdays 744 trips are scheduled while on Saturday and Sunday, 533 and 382 trips are scheduled respectively. The distribution of these trips is shown in Figure 6. On weekdays, the busiest hours are between 7:00 and 9:00 and 16:00 and 19:00, while on Saturday the busiest hours are in the middle of the day. On Sunday the trips are evenly distributed throughout the day. The locations of two fictional bus depots and three recharge stations are shown in Figure 2. These locations are not included in the open-source OV dataset and are placed at logical locations.

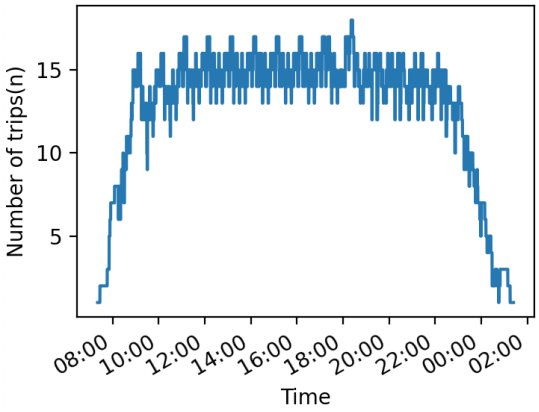
Figure 6: Trip distributions.



(a) Trip distribution weekdays.



(b) Trip distribution Saturday.



(c) Trip distribution Sunday.

In Table 1, the total distance of all trips and the maximum number of trips that need to be executed at the same time are presented. Using this data, a lower bound can be calculated by adding the product of the total distance of all trips with the variable cost and the product of the maximum number of trips with the fixed cost.

	Weekdays	Saturday	Sunday
Number of trips(n)	744	533	382
Total distance(km)	17697	11315	7590
Max number of trips(n)	50	30	18
Data lower bound(€)	67697	41315	25590

Table 1: Specifics of the data set.

6.3 Petrol buses

First, the MDVSP is solved using the multi-commodity flow formulation to get an optimal integer solution. Then, the MDVSP is solved using the column generation algorithm. The column generation algorithm will be used to solve the LP-relaxation of the problem and thereafter it will be used to get an integer solution.

6.3.1 Multi-commodity flow formulation

The multi-commodity flow formulation as described in subsubsection 4.1.1 is implemented in *Gurobi Optimizer* and for the instances of *Transdev*, it is possible to generate the optimal solution because the instances are relatively small. The values of the solution can be found in Table 2. These solutions are used to test the performance of the column generation algorithm used for the heuristic.

	Weekdays	Saturday	Sunday
Number of buses(n)	50	30	18
Solution value(€)	70185	42544	25714
Running time(s)	386	207	99
Relative gap with the data lower bound(%)	3.7	2.9	0.5

Table 2: Optimal solutions found using the multi-commodity formulation.

The results show that the gap between the data lower bound found in Table 1 is not more than 3.7 percent. The relative gap is decreasing by the number of trips performed per day for the data of *Transdev*. With a gap that is at most 3.7 percent the data lower bound is close to the optimal value.

6.3.2 Column generation approach

First, the LP relaxation of the MDVSP is solved using the column generation algorithm. In this algorithm, the shortest path is calculated for each depot in the pricing problem, and if the route found has negative reduced cost, it is added to the restricted master problem. In each iteration, the number of routes added to the restricted master problem is smaller or equal to the number of depots. To keep the restricted master problem manageable, routes with positive reduced costs are deleted from the restricted master problem after every 200 iterations, as described in subsubsection 5.3.2. If the number of trips exceeds 400 trips, the pricing problem is solved heuristically. The technique described in subsubsection 5.3.5 is used. When the pricing problem is solved heuristically, compatible trips are only considered if the starting time is within an hour of the previous trips ending time. The results for the LP relaxation when the column generation algorithm runs for one hour are outlined in Table 3.

	Weekdays	Saturday	Sunday
Solution value(€)	71632	44307	25714
Number of iterations(n)	2000	2700	1950
Gap with the optimal value (%)	2.0	4.1	0.0
Time used to solve the pricing problem(s)	450	350	140
Time used to solve the master problem(s)	3150	3250	1200

Table 3: Solution of the LP relaxation solved by column generation for one hour.

As shown in Table 3, the algorithm is able to find the optimal value within one hour for Sundays. On weekdays and Saturdays, the algorithm is unable to find the optimal value within one hour. But the gap between the optimal value and the solution is not more than 4.1 percent. More time is used in the master problem compared with the time used in the pricing problem.

Second, the column generation algorithm is used to generate an integer solution, as described in subsubsection 4.2.3. In the first iteration, when no buses are scheduled, the algorithm will search for 30 minutes before selecting a route to fix. The technique of adding routes found in the last iteration to the initial solution of the current iteration, as described in subsubsection 5.3.3, is used to speed up the algorithm. For iterations other than the first one, in which more than 400 trips need to be scheduled, the algorithm searches for 15 minutes before selecting a route to fix, and for fewer than 400 trips, it searches for 5 minutes before selecting a route to fix. The pricing problem is solved heuristically if more than 250 trips are not yet fixed, and it is solved completely if fewer than 250 trips are not yet fixed. Similar to the LP relaxation, only trips starting within an hour of the ending time of the last trip are considered if the pricing problem is solved heuristically. If the optimal solution is found in an iteration, the algorithm will immediately select a route to fix. When selecting a route to fix, the algorithm will prioritize routes with an x value higher than 0.8. If there are multiple routes with x values higher than 0.8, the route with the most trips will be chosen. If none of the routes have an x value higher or equal to 0.8, the route with the highest x value is selected. The algorithm stops after all trips are scheduled. The results can be found in Table 4.

	Weekdays	Saturday	Sunday
Number of buses(n)	52	31	18
Solution value(€)	72186	43789	26160
Gap with the optimal value(%)	2.9	2.9	1.7
Running time(s)	27900	14600	8400

Table 4: Solution for petrol buses.

The results of Table 4 show that the column generation algorithm is working well. The gap between the column generation algorithm and the optimal value is not more than 2.9%. The number of buses does not significantly increase, with only one additional bus on Saturdays and two additional buses on weekdays. On Sundays, the number of buses is equal to the optimal value.

To see the performance of the column generation algorithm, there is a visualization in Figure 7 of the column generation algorithm for the weekdays that shows the objective value over time for

the LP relaxation. As depicted in this figure, the objective value decreases significantly at the start of the algorithm's execution. Eventually, it reaches an asymptote which is the optimal value. This is because it becomes increasingly difficult for the algorithm to improve the solution when it gets closer to the optimal value. The algorithm will stop when the optimal value is found, but a time limit of one hour has been set for the algorithm to run. This visualization helps to understand the convergence of the column generation algorithm.

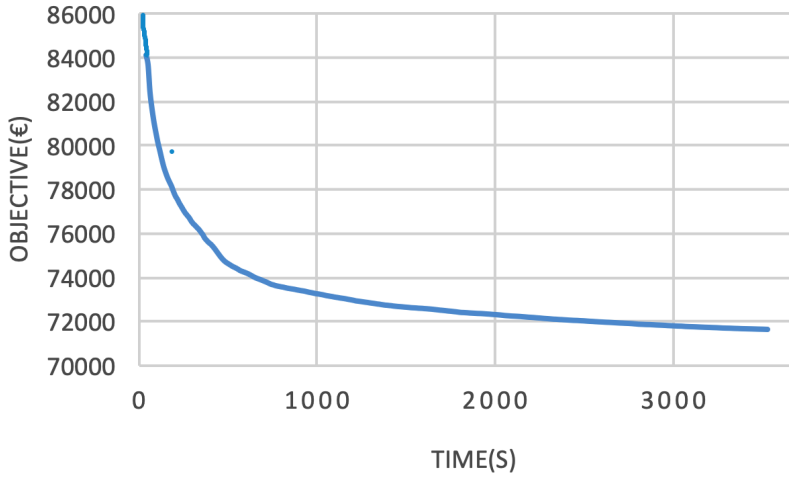


Figure 7: Objective of the LP relaxation over time for the column generation algorithm for a weekday.

6.4 Electric buses

For electric buses, there is no formulation implemented that can solve the problem to optimality. Therefore, the LP relaxation is solved using the column generation algorithm described in subsection 5.2 to retrieve a lower bound to the problem. The stopping criteria are the optimal solution or a running time of 6 hours. For none of the test instances, the algorithm was able to find the optimal solution within six hours, so the solution can not be considered as a lower bound. Therefore, the optimal solution for petrol buses will also be the lower bound for electric buses.

Similarly as for petrol buses, the pricing problem in the column generation algorithm is solved heuristically if more than 400 trips need to be scheduled. When the pricing problem is solved heuristically, compatible trips are considered if their starting time is within one hour of the previous ending time.

In the column generation algorithm to solve the LP relaxation, the algorithm will add multiple routes to the restricted master problem after it solved the pricing problem. The MLSA used to solve the pricing problem for each depot often has multiple labels at the sink, so multiple routes can be added to the restricted master problem in each iteration the pricing problem is solved. In this algorithm, the 15 routes with the lowest negative reduced costs for each depot will be added to the restricted master problem. This means that the maximum number of routes added to the restricted master problem in one iteration is 15 times the number of depots. To keep the restricted master problem manageable routes are deleted after every 30 iterations if their reduced cost is positive as

described in subsubsection 5.3.2. In Table 5 the result of the column generation algorithm, used to solve the LP relaxation of the MDEVSP, after a running time of six hours is reported. In this optimization, there are six recharge points at each recharge station.

Bus range of 200 km	Weekdays	Saturday	Sunday
Costs of the bus schedule(€)	74925	47903	26890
Number of iterations(n)	480	476	555
Gap between the optimal value of petrol buses	6.7	12.6	4.6

Table 5: Solution of the LP relaxation solved by column generation for six hours.

The table above indicates that the price of the LP-relaxation of the bus schedule increases by a maximum of 12.6% when switching from petrol buses to electric buses.

Next, the column generation algorithm is used to solve the MDEVSP and generate an integer solution, as described in subsubsection 5.2.4. In the first iteration, when no buses have been selected, the algorithm will run for 3 hours and 20 minutes before selecting a route to fix. The technique of adding routes from the last iteration to the initial solution of the current iteration is used, as described in subsubsection 5.3.3. After the first iteration, if there are more than 400 trips to schedule, the column generation algorithm will run for 30 minutes before selecting a route to fix. Otherwise, it will run for ten minutes. If the optimal solution is found in the column generation algorithm at any iteration, it will select a route to fix immediately. The pricing problem in the column generation is solved heuristically if more than 150 trips are not yet fixed in an iteration, and it is solved completely if fewer than 150 trips are not yet fixed. Similar to the LP relaxation, only trips starting within an hour of the ending time of the last trip are considered if the pricing problem is solved heuristically. When selecting a route, the algorithm will prioritize routes with an x value higher than 0.8. If there are multiple routes with x values higher than 0.8, the route with the most trips will be chosen. If none of the routes have an x value higher or equal to 0.8, the route with the highest x value is selected. The algorithm stops after all trips are planned and the bus schedule is created. First, The results with different numbers of rechargers at each recharge station are shown in Table 6.

Bus range of 200 km	Weekdays		Saturday		Sunday	
	Cost (€)	Buses(n)	Cost (€)	Buses(n)	Cost (€)	Buses(n)
Two recharge points at each recharge station	77062	56	50086	37	34243	25
Four recharge points at each recharge station	75784	55	49943	37	34101	25
Six recharge points at each recharge station	75750	55	50186	37	34055	25
unlimited recharge points at each recharge station	75268	55	49890	37	33987	25

Table 6: Solutions for electric buses with different numbers of recharge points at each recharge station.

Thereafter, The algorithm will evaluate the impact of changing the range of an electric bus, by comparing the results if the range is set to 250 kilometers, 300 kilometers, and 350 kilometers instead of 200 kilometers. In the future, battery technology may improve, and therefore the range of a bus can be increased. It is also assumed that the time required for a full recharge, from 0 to 100%, will remain the same, at 35 minutes, and the number of recharges at the three recharge

stations is set to six. The results are outlined in table Table 7.

	Weekdays		Saturday		Sunday	
	Cost (€)	Buses(n)	Cost (€)	Buses(n)	Cost (€)	Buses(n)
Buses with a range of 200 kilometers	75750	55	50186	37	34055	25
Buses with a range of 250 kilometers	74975	55	49300	36	33174	24
Buses with a range of 300 kilometers	74146	54	48463	35	32365	23
Buses with a range of 350 kilometers	73676	53	47245	34	29987	21

Table 7: Solutions for electric buses with different ranges.

Last, the algorithm will assess the impact of changing the recharge time of electric buses. With the potential for improvements in recharge stations, the speed of recharging a bus may increase. The time to recharge a battery from 0% to 100% will be set to 20 minutes, 25 minutes, and 30 minutes instead of 35 minutes and the formula given in subsection 2.2 will be used. The range of electric buses is set at 200 kilometers and the number of recharges at all recharge stations is set to six. The results are outlined in Table 8.

Bus range of 200 km	Weekdays		Saturday		Sunday	
	Cost (€)	Buses(n)	Cost (€)	Buses(n)	Cost (€)	Buses(n)
Complete battery recharge time of 35 minutes	75750	55	50186	37	34055	25
Complete battery recharge time of 30 minutes	75450	55	49683	37	33604	25
Complete battery recharge time of 25 minutes	74970	54	49054	36	33126	24
Complete battery recharge time of 20 minutes	74546	54	48690	36	32569	23

Table 8: Solutions for electric buses with different range speeds.

In Table 6, Table 7, and Table 8 the same bus timetable is scheduled with different configurations. The consequences of changing the number of charging points, the range of a bus, and the recharge speed of a bus can be observed. The greatest impact is seen with changing the range of the bus, followed by the recharge speed, and finally, the number of charging points has the least impact. The biggest improvement in the bus schedule is observed on weekends when extending the range of an electric bus or decreasing the recharge time of an electric bus. The impact of changing the number of recharge points at each station is only noticeable on weekdays since there are enough chargers available for weekends. During weekdays, there is only a noticeable difference in costs when two chargers are available at each station. Otherwise, there are already sufficient chargers, and the schedule costs hardly differ.

To evaluate the performance of the column generation algorithm for electric buses, a visualization is created similar to that of petrol buses. The objective function of the LP relaxation is plotted against time, showing how it decreases over time. The visualization is for buses with a range of 200 kilometers and a recharge speed consistent with the one in Figure 3. The visualization is of a weekday, and six rechargers are placed at each recharge station.

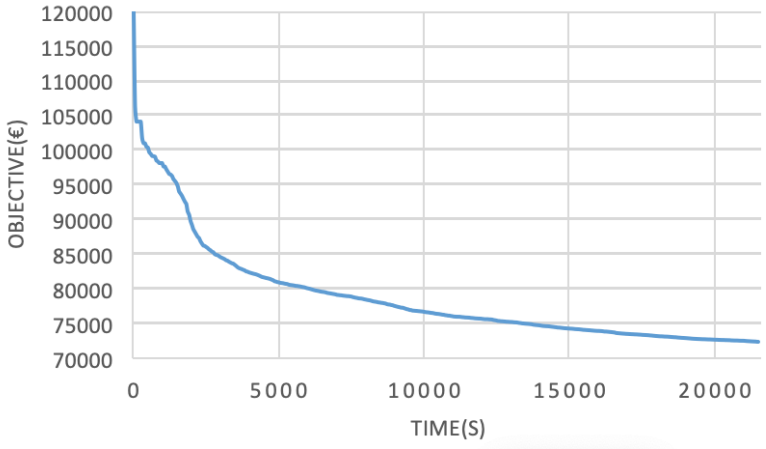


Figure 8: Objective of the LP relaxation of electric buses over time for the column generation algorithm for a weekday.

Once the optimization process is completed and the bus schedule is created, visualizations are created to help understand the bus schedule. These visualizations include a representation of all the recharge stations, showing how many buses are being charged at each station at any given point in time. Besides, there is a visualization of all the buses, showing their activities and state of charge at any point in time. These visualizations allow for a better understanding of the bus schedule and how to improve the configurations. The visualization of the recharge stations with six rechargers at all recharge stations for weekdays with electric buses with a range of 200 kilometers and 35 minutes of full recharge time is shown in Figure 9.

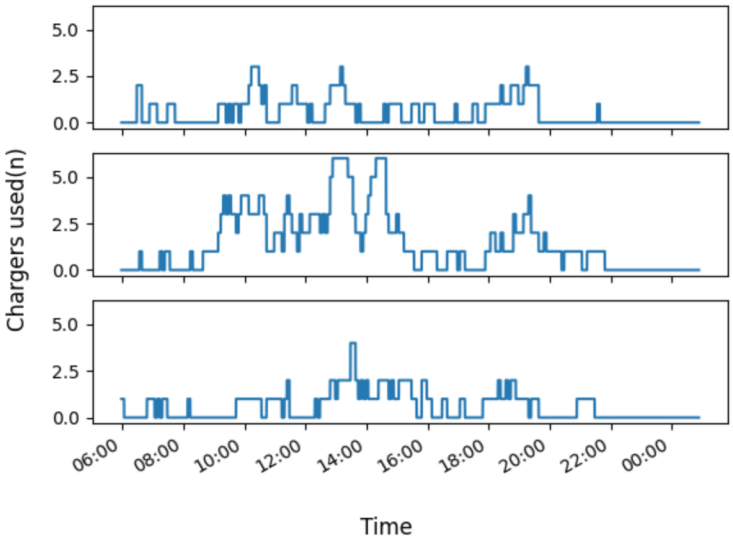


Figure 9: Visualization of the use of recharges at the three recharge stations.

7 Conclusion

A column generation-based heuristic was used to solve the MDVSP and MDEVSP in order to determine the additional cost of incorporating a range constraint when using electric buses. To isolate the effects of the range constraint, the fixed and variable costs of petrol and electric buses were set equal to each other. As few assumptions as possible were made in order to keep the problem as realistic as possible. This includes allowing for non-linear recharging and partial charging of electric buses.

All instances of the MDVSP were solved to optimality using the multi-commodity formulation and implemented in *Gurobi Optimizer*. These optimal values indicate that for the MDVSP the column generation algorithm is able to find solutions close to the optimal value. The LP relaxation was solved to optimality using the column generation algorithm within one hour for an instance of 382 trips, and for larger instances, the solution was approaching the optimal value. This can be seen in the asymptotic behavior of the objective in Figure 7 and the gap between the optimal value and the LP relaxation solution, found by the column generation algorithm, was at most 4.1%.

The column generation algorithm was then used to generate a heuristic integer solution for the MDVSP that was at most 2.9% higher than the optimal solution for the test data. This indicates that the algorithm performs well. The multi-commodity flow formulation was able to find the optimal solution for the MDVSP when scheduling up to 744 trips for the *Transdev* data set. However, for larger instances, the multi-commodity flow formulation would not be able to find the optimal solution within a reasonable amount of time. Therefore, the column generation algorithm was developed as a solution method for larger instances. In addition, the incorporation of additional constraints can be difficult or even impossible for the multi-commodity flow formulation but is easier for the column generation algorithm. Therefore, the column generation algorithm is used to incorporate the range constraint for electric buses.

After scheduling the bus timetable with petrol buses, the same timetable was scheduled using electric buses. Recharge windows were set to 300 seconds, after which the algorithm determined whether the bus continues recharging or resume driving based on the current state of the battery. The recharging process is not linear and can be modeled by any given function to stay as close as possible to reality. For the data set of *Transdev*, the column generation algorithm showed that at most seven additional buses were required for the bus schedule on Sunday, six extra buses on Saturday, and three extra buses for weekdays when compared to the column generation schedule of petrol buses. For these results, it is assumed that an electric bus has a range of 200 kilometers and a recharge scheme as visualized in Figure 3. The least amount of extra buses were needed on weekdays. This can be explained by the peak hours in the bus timetable on weekdays as shown in Figure 6. The peak hours on weekdays allow the buses to recharge during non-peak hours, and therefore less extra buses are required. On Saturday and Sunday, the distribution of trips is more equally spread out over the day. This is making it more difficult to find *good moments* for recharging and therefore more extra buses are required to schedule the same timetable.

Most trips are executed on weekdays, and most bus schedules have peak hours on weekdays. Therefore, from a logistical point of view, it is possible to switch from petrol to electric buses. By changing the waiting time that already exists in most bus routes to recharging moments, the algo-

rithm is able to efficiently plan the recharge schedule in a way that minimizes the additional costs associated with the use of electric buses. In addition, the buses will mostly be recharged during non-peak hours, times at which the electricity grid is less used. This is an advantage as it helps to prevent overloading the grid. By effectively planning the recharging of electric buses and making sure there are enough recharge points, the algorithm is able to make the transition to electric buses a feasible option for bus operators from a logistical point of view.

In addition to optimizing the schedule for electric buses, the algorithm can also be used to determine the optimal number and locations of recharge stations. By running the algorithm with different configurations, the effects of adding or removing a recharge point can be determined. The visualizations provided in the Figure 9 can assist in determining the necessary number of recharge stations and recharges to implement at a recharge station. For the data set of *Transdev*, the impact of extra buses by adding or subtracting rechargers at a recharge station was only observed for the data of weekdays. When two rechargers were placed at all three recharge stations, it resulted in an extra bus when compared to placing four or more rechargers at recharge stations. However, when analyzing the results more closely, it was discovered that most of the recharging happened at one recharge station. Therefore, placing only extra chargers at that station will most likely be sufficient.

After optimizing the timetable with electric buses with a range of 200 kilometers and a recharge time of 35 minutes, the configurations of buses were varied to see the consequences of changing the recharge time and range of an electric bus. The bus schedule is most affected on weekends when the range of the electric bus is extended. The number of buses required to perform the timetable on Sunday decreased from 25 buses to 21 buses when the range of an electric bus was extended to 350 kilometers. During weekdays, the impact is smaller when changing the configurations of electric buses as the schedule is already closer to the optimal value.

Concluding, the results indicate that it is feasible to calculate the additional cost of transitioning from petrol to electric buses. The column generation algorithm can be used to efficiently plan the charging of electric buses, minimizing the number of additional buses required to execute the same bus timetable. Therefore, the algorithm can be used to compare the bus schedules of electric and petrol buses using a given data set in order to assess the logistical impact of the transition. Besides, it can also be run with different configurations to determine the optimal number of recharge stations and rechargers per station and to assess the impact of improving the range and recharge speed of electric buses.

8 Discussion

This thesis analyzed the additional cost and corresponding bus schedules of using electric buses instead of petrol buses in public transportation schedules. A column generation heuristic was used to solve the MDVSP and the MDEVSP, and minimal assumptions were made about the characteristics of electric buses to keep them as close to reality as possible.

The results showed that it is possible to calculate the extra cost of replacing petrol buses with electric buses and check the feasibility of using electric buses from a logistical point of view. While the implementation of electric buses may require a few additional buses to execute the same bus timetable, the use of the column generation-based heuristic to efficiently plan the charging of the electric buses can help to optimize their use. This algorithm can be used to compare the bus sched-

ules of electric and petrol buses using a given bus timetable in order to assess the impact of the transition.

An advantage, when transitioning to electric buses, is that it has the least impact on the number of buses needed during weekdays when there are peak hours in the timetable and most buses are required. This means that the total number of buses required at a bus depot does not change significantly when electrifying the buses. Besides, extra buses required for the weekends are already available making the effect of the extra buses needed on Saturdays and Sundays less of a problem. Additionally, electric buses are mostly charged during off-peak hours on weekdays, which helps to avoid overloading the electric grid as in off-peak hours the grid is used the least.

However, the feasibility and impact of changing from petrol to electric buses require further investigation. Factors such as the cost of changing all petrol buses to electric buses, the difference in fixed and variable costs between petrol and electric buses, the power supply at charging stations, the availability of multiple types of chargers with different recharge speeds, and the existence of different types of electric buses with varying ranges and recharge speeds must be considered.

Overall, this thesis highlights the logistical impact of using electric buses instead of petrol buses as a more sustainable alternative in public transportation systems. By carefully planning and optimizing the use of electric buses, it is possible to minimize the number of extra buses associated with this transition. Further research in this area can help to accelerate the transition of electric buses and contribute to the overall goal of reducing emissions and improving the sustainability of transportation.

References

Bertossi, A. A., Carraresi, P., & Gallo, G. (1987). On some matching problems arising in vehicle scheduling models. *Networks*, *17*, 271-281. Retrieved from https://doi.org/10.1002/net.3230170303

Bianco, L., Mingozzi, A., & Ricciardelli, S. (1994, Jan). A set partitioning approach to the multiple depot vehicle scheduling problem. *Optimization methods*, *3*(1-3), 163-194. Retrieved from https://www.tandfonline.com/doi/abs/10.1080/10556789408805563

Desaulniers, G., Lavigne, J., & Soumis, F. (1998). Multi-depot vehicle scheduling problems with time windows and waiting costs. *European Journal of Operational Research*, *111*(3), 479-494. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221797003639

Desrosiers, J., & Lübbecke, M. E. (2005). A primer in column generation. In G. Desaulniers, J. Desrosiers, & M. M. Solomon (Eds.), *Column generation* (pp. 1–32). Boston, MA: Springer US.

de Vos, M., van Lieshout, R., & Dollevoet, T. (2022). *Electric vehicle scheduling with capacitated charging stations and partial charging.* Retrieved from https://doi.org/10.48550/arXiv.2207.13734

Electric bus range, focus on electricity consumption. a sum-up. (2022, Nov). *Sustainable bus*. Retrieved from https://www.sustainable-bus.com/news/electric-bus-range-electricity-consumption

Forbes, M. A., Holt, J. N., & Watts, A. M. (1994). An exact algorithm for multiple depot bus scheduling. *European Journal of Operational Research*, *72*(1), 115. Retrieved from https://doi.org/10.1016/0377-2217(94)90334-4

Guedes, P. C., & Borenstein, D. (2015, Dec). Column generation-based heuristic framework for the multiple-depot vehicle type scheduling problem. *Computers & Industrial Engineering*, *90*, 361-370. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835215003976

Houwelingen, J. V. (2018). A column generation heuristic for multi-depot vehicles scheduling with electric vehicles. *MSc thesis, Erasmus University Rotterdam*. Retrieved from http://hdl.handle.net/2105/44794

Pepin, A. S., Desaulniers, G., Hertz, A., & Huisman, D. (2009). A comparison of five heuristics for the multiple depot vehicle scheduling problem. *Journal of scheduling*, *12*(1), 17-30. Retrieved from https://doi.org/10.1007/s10951-008-0072-x

Ribeiro, C. C., & Soumis, F. (1994). A column generation approach to the multiple-depot vehicle scheduling problem. *Operations research*, *42*(1), 41-52. Retrieved from http://or.journal.informs.org/cgi/content/abstract/42/1/41

Wang, C., Guo, C., & Zuo, X. (2021, Nov). Solving multi-depot electric vehicle scheduling problem by column generation and genetic algorithm. *Applied soft computing*, *112*, 107774. Retrieved from https://dx.doi.org/10.1016/j.asoc.2021.107774

Wu, W., Lin, Y., Liu, R., & Jin, W. (2022, Jan). The multi-depot electric vehicle scheduling problem with power grid characteristics. *Transportation research. Part B: methodological*, *155*, 322-347. Retrieved from https://dx.doi.org/10.1016/j.trb.2021.11.007