Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis BSc^2 Econometrics and Economics

Electric Vehicle Routing Problem with Time Windows, Non-linear Partial Recharging & Multiple Chargers

Bhavika Adapa (559060)

Supervisor:	Bart van Rossum
Second assessor:	dr. Ece Karakoyun
Date final version:	1st July 2024

Abstract

The transition from the use of internal combustion engine vehicles to electric vehicles has prompted a change in infrastructure, giving rise to various recharging technologies, with typically non-linear recharging rates. To model this real-world scenario in the context of vehicle routing problems, we consider the electric vehicle routing problem with time windows and partial recharging, and adapt it to include multiple chargers and non-linear recharging rates (EVRPTW-NLPR-MC). We propose a matheuristic to solve this problem, which combines an adaptive large neighbourhood search with a set partitioning formulation. Furthermore, we test for a variant of this matheuristic by introducing a labeling algorithm with different recharge policies, to insert charging stations on the routes. A computational study is conducted using benchmark instances from literature. Firstly, we conclude that using the set partitioning formulation as an additional step can reduce the solution cost by up to 5.27% compared to the neighborhood search alone. Secondly, we find that the standard matheuristic outperforms the variant with the labeling algorithm, offering an average reduction of solution cost by 0.43% and significantly lower computational time - 32 times faster. Thirdly, we observe that solution costs for linear and non-linear recharging scenarios differ only by 0.12% on average. Lastly, including multiple technologies allows for cost savings up to 8.32% relative to the scenario with a single charger.

1 Introduction

Harbouring a 25% share of the world's CO2 emissions, the transport sector raises concerns over the global energy consumption and greenhouse gas (GHG) emissions (United Nations Environment Programme, 2024). Road transport accounts for the highest proportion of transport GHG emissions in the EU at 76% in 2021. Meanwhile, the demand for freight transport in the EU has increased by 22% between 2000 and 2019 (European Environment Agency, 2022). Therefore, the European Parliament (2022) necessitates the reduced use of internal combustion engine vehicles (ICEVs); 50% reduction in ICEVs in urban transport by 2030 and complete phase-out by 2050.

In this pursuit to achieve a carbon-neutral transport sector, the paradigm shift from ICEVs to electric vehicles (EVs) proposes a promising future, especially for freight transport. However, the mass implementation of EVs is still restricted given issues such as limited driving ranges due to battery capacity and degradation, long recharging times, restricted availability of infrastructure in terms of public charging stations and a high initial investment cost (Touati-Moungla & Jost, 2012). In EV management, a crucial aspect which justifies that investment cost is precise, efficient and real-time EV routing; it plays a significant role in promoting the use of EVs (see survey on VRP by Toth and Vigo (2002), Golden, Raghavan and Wasil (2008)). Therefore, this has given rise to the Electric Vehicle Routing Problem (EVRP), which aims to discover the optimal routing strategy with the primary objective of minimising the total distance or cost (including energy consumption costs) incurred by a fleet of EVs. Recent literature on EVRP and its variants attempt to make this combinatorial optimisation problem a tractable one.

Initial research dealing with routing problems in greener transportation mediums includes the Green Vehicle Routing Problem (GVRP) by Erdoğan and Miller-Hooks (2012). While the vehicles in GVRP benefit from rapid recharging times, the limited availability of biodiesel or CNG recharging stations poses a problem. Therefore, in contrast to GVRP, the Electric Vehicle

Routing Problem with Time Windows (EVRPTW) introduced by Schneider, Stenger and Goeke (2014), focuses on a fleet of EVs whose battery discharges proportional to the distance travelled. Thus, the EVs may require recharging stops to complete their tour. The authors assume a linear recharge rate, proportional to the recharge amount needed and also assume full recharging (FR) of the battery. Keskin and Çatay (2016) relax the FR assumption and cater to partial recharging (PR), to reduce recharging time based on the practical insights such as FR being unnecessary for a vehicle nearing the end of its tour to reach the depot. Hence, improved solutions may be found through permitting the EV to attend another unvisited customer in the freed up time window.

Further delving into managing the battery and recharging aspect of EVs, a battery recharge operation may take place via different technologies, suggesting different recharge times and corresponding costs. For instance, slow chargers are the cheapest charging option, necessitating a few hours of recharge time; suitable for overnight charging at the depot. Other types of charging technologies allow for faster yet more expensive recharges, such as the CHAdeMO protocol (e.g. Paschero, Anniballi, Del Vescovo, Fabbri and Mascioli (2013)), which permits full battery recharges within an hour. Therefore, the optimisation problem posits a tradeoff between the lower recharge times and the higher energy cost of using more efficient recharging technologies, as explored by Felipe, Ortuño, Righini and Tirado (2014). In addition to this, Montoya, Guéret, Mendoza and Villegas (2017) were the first to model and incorporate non-linear (NL) recharging functions for various charging technologies in their EVRP-NL model. This adds a realistic component to the EVRP problem, since battery level increases concavely over time (Pelletier, Jabali, Laporte & Veneroni, 2017), suggesting different charging rates despite using the same technology.

In this paper, we extend the EVRPTW-PR to a more practical setup with multiple chargers and non-linear recharging (EVRPTW-NLPR-MC), and present a mathematical formulation of the problem in terms of a mixed-integer linear program (MILP). Due to the complexity of the formulation, we present a two-stage matheuristic with Adaptive Large Neighbourhood Search (ALNS) and a set partitioning formulation (ALNS-SP) to optimise the combination of routes selected to minimise costs. Moreover, as an alternative method, we employ a *labeling algorithm* with the ALNS framework following Zhao and Lu (2019), to allow for optimal station insertion in the routes. This paper finds that ALNS-SP outperforms ALNS on average by 0.53%, with the largest improvement at 5.27%. We also discover that the standard ALNS-SP outperforms the ALNS-SP with *labeling algorithm* and FR policy by 0.43% for the objective value, and benefits from approximately 32 times faster average computational time. Moreover, the solution cost when adopting non-linear recharging is on average 0.12% higher than linear recharging; therefore, the linearity assumption for battery recharging is sufficient. Lastly, leveraging multiple charger technologies can reduce the average cost by up by 8.32% when using fast instead of slow chargers.

The remainder of the paper is organised as follows: Section 2 presents a literature review on the related studies in this field. Section 3 provides the problem description and the formulation of the mathematical model. Section 4 details the methodology including the ALNS-SP framework and the *labeling algorithm*. The computational study and discussion of results can be viewed in Section 5. Finally, concluding remarks and directions for future research are given in Section 6.

2 Literature review

Literature on EVRPs have been increasing in the recent years. The Recharging VRP (RVRP) was first presented by Conrad and Figliozzi (2011), where the EVs are able to recharge at specific customer nodes while servicing those customers. Routes in RVRP were iteratively constructed and improved, with the dual objective of minimising the size of the EV fleet used and also reducing total costs computed based on distance travelled, recharging and service times. By assumption, recharging time is constant and upon departure from a node, the battery level state of charge (SoC) could be fully or partially recharged; capped at 80% of battery capacity.

A key extension to VRP models was developing the EVRP with various recharging policies as well as time windows for customer deliveries. Primary, an EVRP was studied by Wang and Cheu (2013) through modelling an electric taxi fleet, wherein they aimed to minimise total distance travelled given an upper bound on the route time and recharging constraints. They assume constant battery discharge and recharge rates, as well as full recharging at charging stations (CSs). They also provided three different recharging plans offering different driving ranges and compared it with the FR policy. Their initial solution is constructed using a nearest-neighbour, sweep and earliest time window insertion heuristic and improved via Tabu Search (TS). Secondly, Schneider et al. (2014) introduced EVRP with time windows (EVRPTW) and tested the performance of their proposed hybrid Variable Neighborhood Search (VNS) and TS method on GVRP and Multi-Depot VRP with Inter-Depot Routes instances. Also, they modified and extended data simulated by Solomon (1987) for EVRPTW instances and reported their results. Thirdly, as an extension to EVRPTW, Keskin and Çatay (2016) proposed a PR model (EVRPTW-PR) with ALNS and simulated annealing (SA) as the acceptance criterion.

For EVRP, authors also placed importance on extending and adapting the methodological framework of ALNS to yield better quality solutions. For instance, Keskin and Çatay (2016) developed several custom destroy-repair operators for customers and stations, and showed that their PR scheme tested on EVRPTW instances from Schneider et al. (2014) improved the solution quality. In contrast, Zhao and Lu (2019) replace the destroy-repair operators for stations in the ALNS framework with a labeling algorithm, for optimal station insertion. Additionally, Zhao and Lu (2019) and Gunawan, Widjaja, Vansteenwegen and Yu (2020) among others performed the ALNS-SP for other variants of VRPs with promising results. This hybrid approach leverages the potential of ALNS to generate good quality solutions for both small and large instances, and the capacity of SP to improve the feasible solutions to nearly optimal solutions.

Concerning the study of using different options to charge EVs, Felipe et al. (2014) were the first to model partial recharging with multiple technologies (slow, medium, fast) in EVRP. They provided different linear recharging rates and associated costs for various technologies, in contrast to the constant recharging rates assumed by prior models. They adopted a constraint on the duration of the total route but not time windows, and proposed a metaheuristic approach involving local search algorithms and SA. Keskin and Çatay (2018) built on this study and extended their EVRPTW-PR presented in 2016 with multiple chargers. They developed a matheuristic approach to solve ALNS with an exact method in CPLEX, and demonstrated its effectiveness and the advantages of employing fast chargers on fleet size and total energy costs.

Since all previous studies considered EVRP with PR and linear recharging functions, the following studies explore more realistic modelling of battery recharging through non-linear charging functions. Montoya et al. (2017) pioneered the extension of EVRP-PR with non-linear charging (EVRP-NL), based on the study by Pelletier et al. (2017) showcasing that battery level increases concavely over time. Montoya et al. (2017) presented a metaheuristic to optimise charging decisions along fixed routes. Froger, Mendoza, Jabali and Laporte (2019) proposed new formulations of EVRP-NL in terms of replication-based models for creating copies of CSs and also a path-based model without CS copies. Similar to Montoya et al. (2017), they also used heuristics and a labeling algorithm to determine optimal charging decisions. Other works on EVRP-NL involve Lee (2021) and Karakatič (2021). The global optimal algorithm for EVRP with an exact non-linear charging function was designed by Lee (2021), where they employed the branch-and-price method to solve the problem. Karakatič (2021) extended EVRP-NL to multi-depot EVRPTW-NL and solved it using a Two-Layer Genetic Algorithm.

Shifting the focus from solely EVs to mixed fleet models, authors have proposed different variants of these models and methods to solve them. Goeke and Schneider (2015) extended EVRPTW to incorporate a mixed fleet of ICEVs and EVs, with the objective to minimise energy consumption and solved it using an ALNS approach. Hiermann, Puchinger, Ropke and Hartl (2016) also used ALNS but with both local search and labeling mechanisms to solve the Fleet Size and Mix Vehicle Routing Problem with Time Windows, with only EVs. In more recent research, Dönmez, Koç and Altıparmak (2022) extended the above fixed fleet problem by including PR with multiple chargers. It aims to minimise both emissions and energy consumption and does so with an ALNS based algorithm, wherein they introduced new mechanisms to handle the complexity of the new constraints.

Overall, this paper fills the gap in the literature firstly from a modelling perspective by catering to time windows within an EVRP with various PR policies, multiple recharging technologies and non-linear recharging functions. Secondly, this paper builds upon the works of Zhao and Lu (2019) and offers a methodological extension by performing ALNS with a *labeling algorithm* instead of destroy-repair operators for stations. Specifically, we introduce various recharging policies for the *labeling algorithm*, to evaluate its performance. Lastly, we apply the set-partitioning formulation to potentially improve the solution quality.

3 Problem description and model formulation

In this paper we explore the electric vehicle routing problem with time windows, non-linear partial recharging and multiple chargers (EVRPTW-NLPR-MC). This problems entails a set of customers with predefined demand levels, time windows for delivery and service durations. The deliveries are serviced by a homogeneous fleet of EVs with fixed cargo load capacities and battery capacities. Hence, while the EVs are travelling, they may require a battery top-up to complete their route, since their battery level degrades proportional to the distance they travel. Under the assumption of PR, the battery level may be recharged to any quantity (only bounded by the battery capacity). Note that the EV departs from the depot fully charged, and it may arrive at / depart from a station with any battery level. However, it must return to the depot with an empty battery level if it has been recharged at least once during its route.

Additionally, we consider two extensions: multiple charging technologies and non-linear recharging. For multiple chargers, all three types (slow, medium, fast) are available at each CS, with different recharging costs (€/kwh). Hence, an EV may stop at a CS and utilise any of the three technologies to recharge; the choice of technology depends on the cost and rate of recharging, such that the cost is kept at a minimum and the time spent recharging does not violate time windows of customers on the route. The overall objective is to find a set of feasible routes covering the demand of all customers with minimum total distance travelled for the EVRPTW-PR, or minimum total cost for the EVRPTW-NLPR-MC. Note that the total cost includes recharging costs and unit costs for each unit of distance travelled.

We present the following model in line with the description provided by Keskin and Çatay (2016). Let $V = 1, \ldots, N$ denote the set of customers and F denote the set of CSs. We assume that a CS may be visited multiple times by the same or different EVs depending on the route requirements, and also may not necessarily be visited. Thus, we define the set F' which contains β dummy variables of each CS in F such that $|F'| = |F| \cdot (1 + \beta)$, with β illustrating the number of visits per CS permitted. Let vertices 0 and $N + 1$ denote two artificial nodes representing the depot, for departure and arrival respectively, such that every route must start at node 0 and terminate at node $N + 1$. Let V' be the set of vertices assigned by the union of all customer and (dummy) CS nodes: $V' = V \cup F'$. We define further sets associated with either the departure or arrival depot nodes (subscripted); namely, $F'_0 = F' \cup 0, V'_0 = V' \cup 0, V'_{N+1} = V' \cup N + 1$. Then, the problem can be represented by a complete directed graph $G = (V'_{0,N+1}, A)$, with the set of arcs $A = (i, j)|i, j \in V'_{0,N+1}, i \neq j$.

Every arc has three attributes: a distance d_{ij} (km), a travel time t_{ij} (hours) and we introduce a cost c_{ij}, which is unitary in this case (€1/km). All EVs have a predefined load capacity C (kg) and battery capacity Q (kW). Moreover, the battery discharges at a constant rate of h (kWh/km) proportional to the distance d_{ij} travelled by the EV ($h \cdot d_{ij}$). Every customer $i \in V'$ has a positive demand q_i (kg) with service duration s_i and time window $[e_i, l_i]$.

We introduce the decision variables τ_i, u_i, y_i which track the starting service time (minutes), the remaining cargo level (kg) and the battery SoC measured in kWh, when arriving at node $i \in V'_{0,N+1}$ respectively. Moreover, the decision variable Y_i (kWh) permits partial recharges, denoting the battery SoC upon departing from node i, such that the recharge quantity results in $Y_i - y_i$, where Y_i is not necessarily equal to Q. Lastly, the binary decision variable x_{ij} is equal to 1 if the arc (i,j) is traversed, else it is equal to 0. The MILP is written as follows:

$$\text{Minimise:} \quad \sum_{i \in V'_0} \sum_{j \in V'_{n+1}, i \neq j} d_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_{j \in V'_{n+1}, i \neq j} x_{ij} = 1 \quad \forall i \in V \tag{2}$$

$$\sum_{j \in V'_{n+1}, i \neq j} x_{ij} \leq 1 \quad \forall i \in F' \tag{3}$$

$$\sum_{j \in V'_0, i \neq j} x_{ij} - \sum_{j \in V'_{n+1}, i \neq j} x_{ij} = 0 \quad \forall j \in V' \tag{4}$$

$$\tau_i + (t_{ij} + s_i) x_{ij} - l_0 (1 - x_{ij}) \leq \tau_j \quad \forall i \in V_0, \forall j \in V'_{n+1}, i \neq j \tag{5}$$

$$\tau_i + t_{ij} x_{ij} + g(Y_i - y_i) - (l_0 + gQ)(1 - x_{ij}) \leq \tau_j \quad \forall i \in F', \forall j \in V'_{n+1}, i \neq j \tag{6}$$

$$e_j \leq \tau_j \leq l_j \quad \forall j \in V'_{0,n+1} \tag{7}$$

$$0 \leq u_j \leq u_i - q_i x_{ij} + C(1 - x_{ij}) \quad \forall i \in V'_0, \forall j \in V'_{n+1}, i \neq j \tag{8}$$

$$0 \leq u_0 \leq C \tag{9}$$

$$0 \leq y_j \leq y_i - (h \cdot d_{ij}) x_{ij} + Q(1 - x_{ij}) \quad \forall i \in V, \forall j \in V'_{n+1}, i \neq j \tag{10}$$

$$0 \leq y_j \leq Y_i - (h \cdot d_{ij}) x_{ij} + Q(1 - x_{ij}) \quad \forall i \in F'_0, \forall j \in V'_{n+1}, i \neq j \tag{11}$$

$$0 \leq y_i \leq Y_i \leq Q \quad \forall i \in F'_0 \tag{12}$$

$$x_{ij} \in \{0, 1\} \quad \forall i \in V'_0, \forall j \in V'_{n+1}, i \neq j \tag{13}$$

The objective function (1) minimises the total distance travelled by the EV. Connectivity constraints (2) and (3) ensure that customers are visited exactly once and CSs are visited at most once, respectively. Constraint (4) maintains flow conservation for the incoming and outgoing arcs at each vertex. Constraints (5) - (7) ensure the time feasibility of service times when travelling from customers (and depot) or CSs to another vertex. Constraints (8) and (9) guarantee that customer demands are satisfied by ensuring sufficient cargo levels are present. The battery SoC is updated and kept non-negative through constraints (10) - (12), when travelling from customers or CSs (and depot) to another node. Constraints (13) define the binary decision variable x_{ij} to determine which arcs are traversed by the EV.

Keskin and Çatay (2018) suggested a formulation to incorporate multiple chargers $m \in M$, where $M = \{1, 2, 3\}$ for slow, medium and fast technologies respectively. We extend their notation to include non-linear recharging, as specified by Montoya et al. (2017). We introduce the charging function $g_i^m(y_i, \Delta_i)$ for every CS $i \in F'$ and technology m, where Δ_i denotes the time spent charging at CS i and the output is the battery SoC upon leaving the CS. Details on the specification can be seen in Section 3.1. We compute the recharge rate $\mu_i^m(y_i, \Delta_i)$ as $\frac{\partial g_i(q_i, \Delta_i)}{\partial \Delta_i}$. Additionally, let θ_i^m be the amount recharged at CS i using technology m, and let c^m be the cost of recharging with technology m. Note that we assume that every CS is equipped with all three technologies. Hence, the binary variables a_i and b_i are used to determine which equipment is being used to charge the EV at CS $i \in F'$: $a_i = 1$ for slow charger, $b_i = 1$ for medium charger and $a_i = b_i = 0$ for fast charger usage. The adjusted MILP is written as follows:

$$\text{Minimise:} \quad \sum_{i \in F'} \sum_{m \in M} c^m \theta_i^m + \sum_{i \in V_0'} \sum_{j \in V_{n+1}', i \neq j} c_{ij} d_{ij} x_{ij} \tag{14}$$

$$\text{s.t.} \quad (2) - (5) \text{ and } (7) - (13)$$

$$\tau_i + t_{ij} x_{ij} + \sum_{m \in M} \mu_i^m (y_i, \Delta_i) \theta_i^m - (l_0 + gQ)(1 - x_{ij}) \leq \tau_j,$$

$$\forall i \in F', \forall j \in V_{n+1}', i \neq j \tag{15}$$

$$Y_i - y_i = \sum_{m \in M} \theta_i^m \quad \forall i \in F' \tag{16}$$

$$0 \leq \theta_i^1 \leq Q a_i \quad \forall i \in F' \tag{17}$$

$$0 \leq \theta_i^2 \leq Q b_i \quad \forall i \in F' \tag{18}$$

$$0 \leq \theta_i^3 \leq Q(1 - a_i - b_i) \quad \forall i \in F' \tag{19}$$

$$a_i, b_i \quad \forall i \in F' \tag{20}$$

The objective function (14) minimises the total energy cost and comprises of two terms. The first term indicates the total recharging cost proportional to the total recharge amounts. The second term incorporates the unit costs associated with the total distance travelled. Constraint (6) is replaced by constraint (15) to account for the different recharging technologies and ensure no violation of time windows when travelling from CSs to another node. Constraint (16) establishes the amount of energy recharge required by the EV at a CS. Constraints (17) - (19) construct bounds for the energy recharge amount and control the recharging technology used. Note that $a_i = b_i = 1$ is not possible due to the non-negativity constraint on θ_i^3. Constraint (20) defines the binary decision variable.

3.1 Modelling the charging functions

Following the estimation method used by Montoya et al. (2017), we define a recharging function $g_i^m(y_i, \Delta_i)$ for every CS $i \in F'$, where $m \in M$ indicates the function specific to the charger in use (slow, medium, fast). This function intakes two inputs; the battery SoC when arriving at i (y_i) and the time spent recharging at i (Δ_i). Montoya et al. (2017) adopt the transformation proposed by Zündorf (2014) as follows to estimate a one-dimensional function. We assume that given $y_i = 0$ and the battery is charged for l time units ($\Delta_i = l$), then the charging function can be estimated as $\hat{g_i^m}(l)$. Furthermore, note that $\Delta_i = \hat{g_i^m}^{-1}(Y_i) - \hat{g_i^m}^{-1}(y_i)$ by using the inverse functions to retrieve the corresponding time units. Therefore, $g_i^m(y_i, \Delta_i)$ can be estimated by $\hat{g_i^m}(\Delta_i + \hat{g_i^m}^{-1}(y_i))$.

Bruglieri, Colorni and Lue (2014) and Hõimoja, Rufer, Dziechciaruk and Vezzini (2012), among others, claim that the function $\hat{g_i^m}(l)$ is known as a concave function with a horizontal asymptote at Q. Moreover, Zündorf (2014) claim that $\hat{g_i^m}(l)$ can be approximated rather accurately through piecewise-linear functions, and Montoya et al. (2017) demonstrate this empirically by fitting such functions to the data provided by Uhrig, Weiß, Suriyah and Leibfried (2015). They do so with nominal average absolute error rates of 0.90%, 1.24%, and 1.90% for CSs of 11, 22, and 44 kW, respectively. Hereforth, we utilise the piecewise-linear function estimated by Montoya et al. (2017) and present a visual representation of this below in Figure 1.

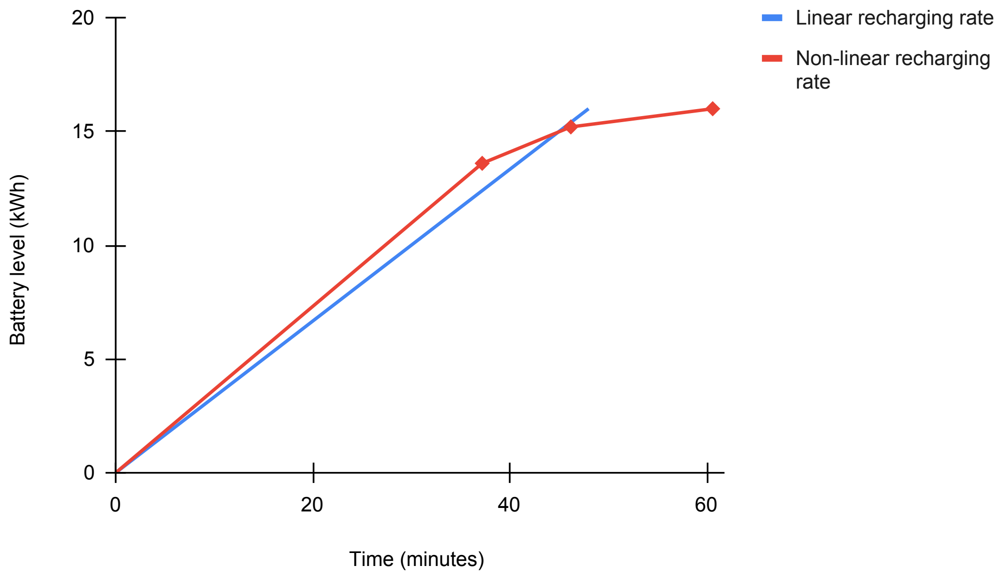


Figure 1: The linear and non-linear recharging functions

Figure 1 shows the linear and non-linear recharging functions with the assumption that the linear recharging rate is 3 min/kwH, and the battery capacity of the EV is 16 kWh (Montoya et al., 2017). The following breakpoints for time in minutes and battery level in kWh are employed: (0, 0), (37.2, 13.6), (46.2, 15.2) and (60.6, 16); at 0%, 85%, 95% and 100% of the battery capacity. To illustrate how the recharging time is estimated in the non-linear case, assume that the arrival battery of an EV is 10 kWh and it needs to recharge until 14.5 kWh. We first calculate the corresponding time in minutes, 27.4 and 81.56 minutes respectively. Therefore, the recharging time required is $\Delta = 81.56 - 27.4 = 54.16$ minutes.

4 Methodology

As adopted by Keskin and Çatay (2016), we introduce ALNS to solve the EVRPTW-PR. Section 4.1 describes the standard ALNS procedure and Section 4.2 explains the construction of the initial solution. Sections 4.3.1, 4.3.2, 4.4.1 and 4.4.2 detail the Customer Removal (CR), Customer Insertion (CI), Station Removal (SR) and Station Insertion (SI) heuristics employed within ALNS. Extending the works of Keskin and Çatay (2016), we employ a matheuristic; namely, ALNS-SP. Section 4.5 presents the SP formulation and elaborates how it is employed to build the ALNS-SP procedure. Furthermore, we introduce the *labeling algorithm* in Section 4.6 and provide an alternative ALNS framework with the *labeling algorithm* in Section 4.6.1.

Our aim is to investigate the following: the effectiveness of ALNS-SP against ALNS, and the performance of the standard ALNS with customer and station destroy-repair operators against ALNS with only customer destroy-repair operators and the *labeling algorithm*. We then apply the best method to the linear and non-linear recharging cases, to compare the effects of the recharging functions. Finally, we will proceed with one type of recharging functions and apply it to the case of multiple chargers.

4.1 Standard Local Search

The ALNS heuristic is performed with CR (Section 4.3.1), CI (Section 4.3.2), SR (Section 4.4.1) and SI (Section 4.4.2) operators. The initial solution is constructed as described in Section 4.2, and ALNS attempts to improve it iteratively for a finite number of iterations. In each iteration, a feasible solution (routes) is destroyed using either SR, CR or *Random Route Removal* (RRR) / *Greedy Route Removal* (GRR); as detailed in Algorithm 1. SR removes stations from the route and requires the potentially infeasible solution to then be repaired with SI. CR or RRR/GRR are conducted to omit visited customers or entire routes from the solution, followed by CI to re-insert those customers into existing or new routes. In case CI creates a battery-infeasible route, *Greedy Station Insertion* (Section 4.4.2) is utilised to make the route battery-feasible. The selected operators iteratively explore the search space until they cannot further minimise the cost function. This local search continues until a local minimum is found in all operators. Note that in each iteration, the operators are chosen randomly based on an adaptive weighting scheme (Section 4.1.1). Moreover, for each iteration, a solution is either accepted or rejected using the SA acceptance criterion (Section 4.1.2).

4.1.1 Choice of destroy-repair operators

The choice of destroy-repair operators depends on a dynamic weighting and scoring procedure, which are adapted for each heuristic based on its past performance, to prepare a set of selection probabilities for all heuristics. Each heuristic i has a weight w_i and a score π_i which are initialised to zero at the start of ALNS. The score may be updated in three ways. If a pair of insertion and removal heuristics discover a new best solution, their scores improve by σ_1; if it yields a solution that is not yet accepted but is better than the previous solution, their scores increase by σ_2; if the solution is worse but accepted in the current iteration due to the SA acceptance criteria (Section 4.1.2), their scores increase by σ_3. This iterative ALNS process is split into several segments, with N_C and N_S iterations for customer-related and station-related operators respectively, as seen in Algorithm 1. When a segment s ends, the weight of heuristic i for the next segment $s+1$ is updated as $w_i^{s+1} = w_i^s(1-\rho) + \rho\pi_i/\theta_i$, where θ_i is the number of times it was used in segment s and ρ is a roulette wheel parameter. Finally, the selection probability of heuristics $i \in 1, \ldots, k$ for the next segment is $P_i^{s+1} = w_i^s / \sum_{l=1}^{m} w_l^s$.

4.1.2 Acceptance Criteria

Similar to Keskin and Çatay (2016), we use the SA acceptance criteria to avoid getting trapped in a local minimum, as seen in Algorithm 1. A new solution X_{new} is always accepted if its cost function is smaller or equal to the current best solution $X_{current}$ ($f(X_{new}) <= f(X_{current})$). Otherwise, X_{new} is accepted with probability $e^{-(f(X_{new})-f(X_{current}))/T}$, where T denotes the current temperature. Based on Ropke and Pisinger (2006), T is initialised as T_0 and is reduced in every iteration proportional to a cooling parameter $\epsilon \in (0,1)$, such that $T = T \cdot \epsilon$. T_0 then equals $-\omega \cdot f(X_0)/ln(0.5)$, where ω is the initial temperature control parameter and a solution $\omega\%$ worse than the initial solution X_0 is accepted with probability 0.5.

Algorithm 1 ALNS algorithm with CR, CI, SR, SI

Generate an initial solution
$j \leftarrow 1$
repeat
    if $j \equiv 0 \pmod{N_{SR}}$ **then**
        Select SR algorithm and remove stations
        Select SI algorithm and repair solution
    else if $j \equiv 0 \pmod{N_{RR}}$ **then**
        for n_{RR} iterations **do**
            Select RRR or GRR algorithm and destroy ω routes from the initial solution
            Select CI algorithm and repair partial solution
            Perform Greedy Station Insertion to insert CS if required
        end for
    else
        Select CR algorithm and remove customers
        if destroyed solution infeasible **then**
            Select CI algorithm and repair solution
            Perform Greedy Station Insertion to insert CS
        end if
    end if
    Utilise SA criterion to accept/reject solution
    $j \leftarrow j + 1$
    if $j \equiv 0 \pmod{N_C}$ **then**
        Update adaptive weights w_i of CR and CI algorithms
    else if $j \equiv 0 \pmod{N_S}$ **then**
        Update adaptive weights w_i of SR and SI algorithms
    end if
until stopping criterion met

4.2 Construction of initial solution

Following from Keskin and Çatay (2016), the initial solution for the ALNS is constructed by iteratively generating feasible routes. Firstly, an empty route is initiated and customer $k \in V$ closest to the depot is added to the route. The set $\hat{V}$ keeps a track of all unvisited customers, and thus is updated by removing the initial customer visited from this set. The insertion cost $c^i_{jk} = d_{ji} + d_{ik} - d_{jk}$ is then computed for all $i \in \hat{V}$ and the customer with the minimum cost is added to the route and removed from $\hat{V}$, if their insertion is feasible. This implies that the time windows of the customers must be respected. If no customer can be added due to insufficient battery SoC, then a CS may be inserted.

For the standard ALNS detailed by Keskin and Çatay (2016), we perform *Greedy Station Insertion* (Section 4.4.2) to determine the cheapest feasible insertion of the station to the current route. For the alternative case, different to Keskin and Çatay (2016), we instead adopt a *labeling algorithm* (Section 4.6) to determine the optimal positioning of stations for the given route. If no customer can be inserted due to time window and battery SoC violations, then a new route is created and this process is repeated until all customers have been visited i.e. $\hat{V} = \emptyset$. The initial solution construction is detailed below in Algorithm 2.

Algorithm 2 Initial solution construction

Initiate a new route and add the customer $k \in V$ closest to the depot
Let set of unvisited customers be $\hat{V} = V \setminus \{k\}$
repeat
    Insertion cost $c^i_{jk} = d_{ji} + d_{ik} - d_{jk}$
    Compute $c^i_{jk} \forall i \in \hat{V}$ on the current route
    if no feasible insertion of customer possible **then**
        Initiate a new route and add the unvisited customer $l \in \hat{V}$ closest to the depot
        Update $\hat{V} = \hat{V} \setminus \{l\}$
    else
        Select customer l s.t. $c^l_{jk} = \min_{\forall i \in \hat{V}} c^i_{jk}$ and make the insertion
        Update $\hat{V} = \hat{V} \setminus \{l\}$
    end if
    if a recharging station is required **then**
        Perform Greedy Station Insertion / Labeling Algorithm
    end if
until all customers are visited i.e. $\hat{V} = \emptyset$

4.3 Destroy-Repair operators for customers

4.3.1 Customer removal

Previous literature by Ropke and Pisinger (2006) and Demir, Bektaş and Laporte (2012) have stated well-known customer removal heuristics. These include *Random, Worst-Distance, Worst-Time, Shaw, Proximity-based, Time-based, Demand-based, Zone, RRR*, and *GRR*. In each iteration of ALNS, one of these CR operators is chosen in an adaptive manner (see Section 4.1.1) to remove γ customers from the route and store them in a removal list $\mathcal{L}$. The number of customers n_c determine the value of $\gamma \in (\underline{n_c}, \overline{n_c})$ randomly via the uniform distribution.

Random Removal selects γ customers at random and removes them from their respective routes. *Worst-Distance Removal* calculates the removal costs of customers as the sum of the distances between the customer and its preceding and succeeding nodes, while *Worst-Time Removal* calculates the cost as $|\tau_i - e_i|$. After collating a sequence of non-increasing costs, these operators repeatedly remove the customer at index $\lfloor |\gamma| \lambda^\kappa \rfloor$, until $\gamma = 0$, where $\lambda \in [0, 1]$ and $\kappa \geq 0$ to introduce randomness in the selection of customers. *Shaw Removal* applies a relatedness measure to remove customers that have similar attributes, such as close proximity, time windows and demand levels: $R_{ij} = \phi d_{ij} + \phi_2 |e_i - e_j| + \phi_3 l_{ij} + \phi_4 |D_i - D_j|$. The higher R_{ij}, the greater the similarity between customers i and j, where $\phi_1 - \phi_4$ are the Shaw parameters and $l_{ij} = -1$ if i and j are on the same route. After arranging relatedness measures in a non-decreasing order, the customer at index $\lfloor |\gamma| \lambda^\eta \rfloor$ is removed, until $\gamma = 0$, where $\eta \geq 0$. *Proximity-based, Time-based* and *Demand-based* removals are special cases of *Shaw Removal*, where ϕ_1, ϕ_2, ϕ_4 are 1 respectively, while other Shaw parameters are 0. *Zone Removal* requires the customers to be assigned to one of the n_Z zones in the Cartesian coordinate system, and all the customers from a randomly chosen zone are removed from the solution.

RRR and *GRR* remove ω routes randomly or greedily (removing route with highest number of customers first) from a feasible solution to try and service all customers over a smaller set of routes, thereby minimising the number of vehicles used. Shorter routes may prompt lower objective values by removing unnecessary trips to and from the depot. Keskin and Çatay (2016) introduced two CR operators which we will also employ in our research; namely, *Remove Customer with Preceding Station (RCwPS)* and *Remove Customer with Succeeding Station (RCwSS)* operators. If a customer is removed from the route, then the battery SoC may be sufficient to visit the following customer without recharging, eliminating the need to visit a station and reducing battery recharging costs.

4.3.2 Customer insertion

Following Keskin and Çatay (2016), we employ the following customer insertion operators iteratively to insert all the customers in $\mathcal{L}$ back into the solution: *Greedy, Regret-k, Time-based* and *Zone insertions*. *Greedy Insertion* inserts the lowest-cost customer i between two feasible nodes j and k. *Regret-k* insertion calculates the difference between a customer's first best insertion and k^{th} best insertion. The customer with the highest difference is then inserted at its best position, to prevent inserting customers earlier on which may yield higher costs in subsequent iterations. We employ *Regret-2* and *Regret-3* methods in our research. *Time-based Insertion* calculates the difference between the total route times before and after adding a customer to the route at a given position. The customer with the lowest insertion cost is inserted first at its best position. *Zone insertion* only considers a subset of routes in contrast to all the routes in the solution. If a randomly selected zone contains customers already visited on certain routes ($V \setminus \mathcal{L}$), it adds those routes to the subset. It then calculates the same cost measure as *Time-based Insertion*, and attempts to insert customers in $\mathcal{L}$ solely in the subset routes.

4.4 Destroy-Repair operators for stations

4.4.1 Station removal

The station removal operators used in this research include *Random Station, Worst Distance Station, Worst-Charge Usage Station* and *Full Charge Station Removal*. The number of stations σ to be removed are determined similarly as γ, depending on the total number of stations (and their copies) available for recharging. *Random Station* and *Worst Distance Station Removal* work similarly to *Random Customer* and *Worst Distance Customer Removal*. *Worst-Charge Usage Station Removal* sorts stations in non-increasing order of the battery SoC with which the EV visits the station, and removes the first σ stations. The idea is to deplete the battery as much as possible prior to visiting a station, thereby improving the efficiency in station usage. Lastly, for the *Full Charge Station Removal* operator, Keskin and Çatay (2016) suggested removing σ stations randomly where EVs are fully recharged. However, we modify this operator and instead remove the first σ stations which have the highest battery recharge amounts at a station. This modification is better suited for the partial recharging situation, since experimentation with various datasets showed that battery levels rarely reach the full recharge amount.

4.4.2 Station insertion

After removing stations from the solution, some routes may become battery-infeasible. Therefore, with every iteration of station removal, one of these station insertion operators is employed: *Greedy Station Insertion, Greedy Station Insertion with Comparison* or *Best Station Insertion. Greedy Station Insertion* identifies a customer/depot on the route with negative battery SoC, and attempts to insert the minimum-cost (least increase in distance) station on the arc between that customer/depot and the previous node. If this insertion is infeasible, then the operator checks to insert a station in the previous arcs. *Greedy Station Insertion with Comparison* compares the cost of inserting the best station on the arc before the negative battery customer as well as the previous arc. The position which yields a lower cost is chosen for station insertion. If neither position offers feasible insertion, then *Greedy Station Insertion* is performed on the remaining previous arcs. Finally, *Best Station Insertion* considers the cost of inserting a station on all previous arcs of the negative battery customer until we hit a station or a depot. The combination of station and position which yields the lowest increase in distance is selected. These station insertion operators are iterated until no negative customer/depot exists on the route.

Note that when SI is required to make a feasible trip to an unvisited customer, Keskin and Çatay (2018) use these insertion operators only in conjunction with the fastest recharging option station insertions. In this paper we consider all three recharging options: slow, medium and fast, to evaluate the trade-off between the number of EV routes required to service all customers and the total routing cost for all EVs. For instance, one may argue that a medium recharging option might be the best to balance number of vehicles used and total energy costs. However, another argument could be that by only allowing the fast charging option during station insertion, we reduce charge durations (Δ_i) which could help one EV serve more customers. Moreover, the recharging cost related decisions are optimised in the second stage of the matheuristic; the SP formulation as described in Section 4.5.

4.5 Using Set Partitioning

We consider a set partitioning (SP) formulation as the second stage of the matheuristic approach. Let Ψ be the set of feasible routes for EVs, which are local optimum solutions populated from various ALNS iterations. We declare a route $r \in \Psi$ feasible if the time windows of the customers are respected, the total cargo load does not exceed the EV's capacity, the battery levels are non-negative and sufficient to visit the next node on the route, and the route starts and ends at depot nodes 0 and $N + 1$ respectively. The rationale for using SP as a post-processing stage of ALNS results is as follows. Keeping in mind our cost minimisation objective (1) or (14), it may be possible that a combination of routes across two local optimum solutions belonging to different ALNS iterations yields a lower total cost of the route.

We introduce a binary decision variable x_r and two parameters α_{ir} and c_r. Note that x_r takes the value 1 if route $r \in \Psi$ is selected in the optimal solution, else it is 0. $\alpha_i r$ is equal to 1 if the customer i is in the selected route $r \in \Psi$, else it is 0. Finally, c_r is the total cost associated with the route. The formulation can then be written as follows:

$$\text{Minimise:} \quad \sum_{r \in \Psi} c_r x_r \tag{21}$$

$$\text{s.t.} \quad \sum_{r \in \Psi} \alpha_{ir} x_r = 1 \quad \forall i \in V \tag{22}$$

$$x_r \in \{0, 1\} \quad \forall r \in \Psi \tag{23}$$

This SP outputs an optimal solution consisting of a subset of the routes $\{r_1, r_2, \ldots, r_k\} \subseteq \Psi$. The objective function (21) minimises the total cost of the optimal solution, aggregated over the costs of individual routes. We only impose one constraint (22), which necessitates that every customer node should be covered exactly once by one of the routes in the optimal solution. Constraint (23) states the binary nature of x_r.

The SP model (21) - (23) is solved using a mixed-integer programming solver, Gurobi. The MILP can be provided a warm start with the best solution of ALNS to reduce the runtime (Zhao & Lu, 2019). The optimisation terminates either upon finding an optimal solution or exceeding the predefined time limit. We will compare the objective values and runtimes of ALNS and ALNS-SP, to evaluate whether ALNS-SP outperforms ALNS. Our hypothesis is that better solutions overlooked by ALNS can be found during this second stage of the matheuristic, and thus ALNS-SP will outdo ALNS. The ALNS-SP procedure is detailed below.

Algorithm 3 ALNS-SP procedure

```
z ← Initial solution
z* ← ALNS(z)
Add local optimum z* to Ψ
repeat
    Update local optimum z* ← ALNS(z*)
    Add local optimum z* to Ψ
until stopping criterion met
z* ← Solve SP with input Ψ
return z*
```

4.6 Labeling algorithm

Taking a different approach than Keskin and Çatay (2016), instead of defining CS removal and insertion operators, we adopt a *labeling algorithm* following Zhao and Lu (2019) to optimise the placement of CSs in our routes. As seen in Figure 2 below, the *labeling algorithm* takes as input the route of artificial depot nodes and customers $r = 0, v_1, v_2, \ldots, v_n, N+1$, where $|V| = n$ as well as the set of CSs F (and their respective copies), where $|F| = m$. For the scenario with multiple chargers, the set F will contain all stations with three charger types. For example, for station S13 the set F contains S13-A, S13-B, S13-C for three types of recharging options and their associated copies (see Section 3). Note that the route r satisfies constraints such as time windows and load capacities, but battery SoC constraints may be violated. Hence, the *labeling algorithm* attempts to satisfy the energy constraints by finding the cheapest feasible insertions for CSs and thus the cheapest overall route from 0 to $N+1$.

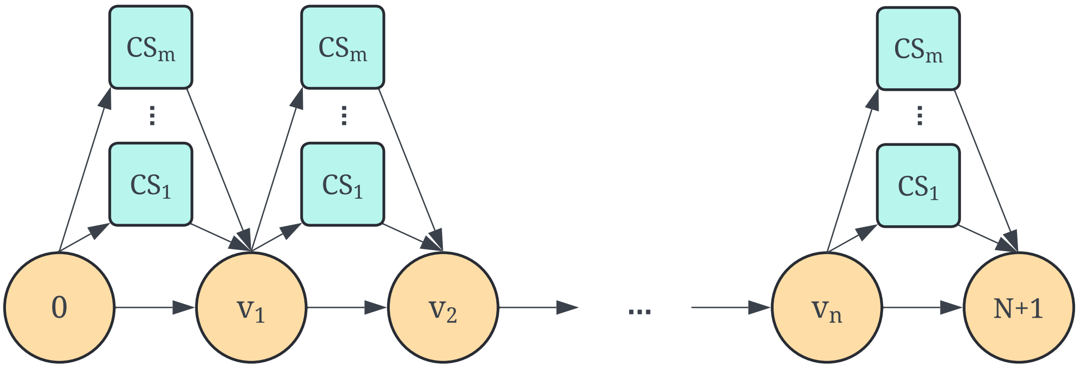


Figure 2: Auxiliary graph for demonstrating the labeling algorithm

We simplify the notation from Zhao and Lu (2019) for our case and denote the label of customer $i \in r$ as $l_i = (\tau_i, y_i, f_i, i, l)$. As stated previously, τ_i and y_i denote the start of service time (i.e. arrival time) at customer i and the battery SoC when arriving at customer i respectively. f_i is the current accumulated value of the total energy costs (see Equation (1)) and l precedes the label l_i. We then define L_i as the set of labels for customer i, and $l_i \in L_i$ is one such label. When travelling from customer i to j, two situations may arise. Firstly, the EV may travel directly from customer i to j, yielding one label. Secondly, the EV may travel from customer i to j via a CS $\in F$, generating at most m labels. Note that in this case it is important to determine the amount of energy that should be recharged at the CS as the recharging time will affect visits to other customers in the route. Since it is difficult to determine an optimal recharging policy, we explore few suggestions from literature and devise the following 5 strategies: Full Recharge (Schneider et al., 2014), Fixed Partial Recharge at 80%, 60% and 40% of battery capacity (Keskin & Çatay, 2016), and Free Partial Recharge with 2 customers (Zhao & Lu, 2019). The last policy suggests that the EV should be charged such that it is able to reach the next customer $j+1$; if that violates the time window of j, then the EV should just be charged sufficiently to travel from the CS to customer j.

We consider a label to be valid if it does not violate time window and battery SoC constraints. Also, we state that a label $l_i = (\tau_i, y_i, f_i, i, l)$ dominates a label $l_i^{'} = (\tau_i^{'}, y_i^{'}, f_i^{'}, i, l^{'})$ for customer i if: $\tau_i \leq \tau_i^{'}, y_i \geq y_i^{'}, f_i \leq f_i^{'}$ and at least one of the inequalities is strict. We prefer a label which has an earlier arrival time at a customer, because each route is time-limited and it allows the EV to meet the time windows of other potential customers yet to be added to the route. Similarly, we prefer a label which has a higher battery SoC upon arrival at a customer, since the EV has more energy to continue its route and can reduce unnecessary visits to CSs; important for lowering both route distance and energy costs. These dominance rules allow for pruning of feasible solutions and focusing on more promising routes by removing the dominated labels; reducing the number of labels to be considered and thus improving the efficiency of the algorithm.

Let $v_0 = 0$ and $v_{n+1} = N+1$. Now, for each vertex $v_{i-1}, i \in 1, \dots, n+1$ we create a set of labels $L_1, \dots, L_{N+1}$ and only include valid labels for each vertex, which are not dominated by other labels of the same vertex. Then, we find the cheapest label in L_{N+1} (end of route) and trace back to v_0 following the cheapest predecessor labels. Finally, the labeling algorithm outputs the cheapest path from v_0 to v_{n+1}. Else, if the labeling algorithm cannot find the cheapest v_0, v_{n+1}-path we consider the inputted sequence of customers to be infeasible.

4.6.1 Alternative Local Search: Labeling Algorithm

As an alternative to the ALNS procedure described in Algorithm 3, we propose a modified ALNS framework wherein CR and CI operators are present; however, SR and SI operators are replaced with the *labeling algorithm* (Section 4.6) for the optimal positioning of CSs on a given route. Therefore, in each ALNS iteration, either CR or RRR/GRR is performed, followed by CI. During CI, the *labeling algorithm* is employed to find the cheapest feasible insertion of CSs. The weights are only updated for customer-related operators per segment of size N_C, as shown in Algorithm 4. The motivation behind applying the *labeling algorithm* instead of the SI and SR operators is that it systematically tracks all feasible, non-dominated labels of an EV. Hence, this results in a more extensive search for optimal station positioning in a route, which may be overlooked by the heuristic-based station operators. Note that it cannot be guaranteed that the *labeling algorithm* will always find a solution that is at least as good at the SI operations, because the quality of solutions depend upon the dominance rules that are applied. Dominance rules can help reduce the computational time, but may lead to premature pruning of potentially optimum solutions in some cases, depending on the nature of the rules.

Algorithm 4 ALNS algorithm with CR, CI and Labeling Algorithm

Generate an initial solution
$j \leftarrow 1$
repeat
    if $j \equiv 0 \pmod{N_{RR}}$ **then**
        for n_{RR} iterations **do**
            Select RRR or GRR algorithm and destroy ω routes from the initial solution
            Select CI algorithm and repair partial solution
            Perform labeling algorithm to insert CS if required
        end for
    else
        Select CR algorithm and remove customers
        if destroyed solution infeasible **then**
            Select CI algorithm and repair solution
            Perform labeling algorithm to insert CS
        end if
    end if
    Utilise SA criterion to accept/reject solution
    $j \leftarrow j + 1$
    if $j \equiv 0 \pmod{N_c}$ **then**
        Update adaptive weights w_i of CR and CI algorithms
    end if
until stopping criterion met

5 Computational study

Within the computational study, we first discuss the performance of ALNS versus ALNS-SP by reporting the results for the small instances. We then evaluate the results for large instances with the better model, for various PR policies (free and fixed) and compare them against the benchmark of the FR policy. Secondly, we compare the results of the standard ALNS model against the alternative ALNS with *labeling algorithm* for various recharging policies and evaluate the models based on computational efficiency and solution quality. Thirdly, we proceed with the best model from the previous step to run analysis for the cases of linear and non-linear recharging functions. Finally, we select either the linear or non-linear functions depending on how different the solution cost is under those two cases, and showcase the effect of multiple chargers on the solution cost through sensitivity analysis.

5.1 Set-Up

The proposed ALNS procedure was validated through running computational experiments on EVRPTW test instances generated by Schneider et al. (2014), which were originally introduced by Solomon (1987). The instances differ based on the type of customer geographical distribution: clustered (C), randomly distributed (R), and both clustered and randomly distributed (RC). The 56 large instances include these problems with 100 customers and 21 recharging stations, wherein each problem contains two subsets, which have different time window lengths, battery and vehicle load capacities. The 36 small instances consist of 3 subsets for 5, 10 and 15 customers respectively, drawn randomly from the 100-customer instances. For hyperparameter tuning, we refer to Keskin and Çatay (2016) and utilise the following parameters present in Table 1.

Table 1: Results from hyperparameter tuning

Customer Removal	Station Removal	Shaw parameters	ALNS iterations	ALNS solution
$\kappa = 4$	$m_r = 0.3$	$\phi_1 = 0.5$	$N_{SR} = 60$	$\sigma_1 = 25$
$\eta = 12$	$\underline{n_s} = \min(0.1\lvert F_N\rvert, 30)$	$\phi_2 = 13$	$N_{RR} = 2000$	$\sigma_2 = 20$
$n_z = 25$	$\overline{n_s} = \min(0.4\lvert F_N\rvert, 60)$	$\phi_3 = 0.15$	$n_{RR} = 1250$	$\sigma_3 = 21$
$\underline{n_c} = \min(0.1\lvert V_N\rvert, 30)$		$\phi_4 = 0.25$	$N_S = 5500$	$\rho = 0.25$
$\overline{n_c} = \min(0.4\lvert V_N\rvert, 60)$			$N_C = 200$	$\epsilon = 0.9994$
			$\mu = 0.4$	

All the results were run on a MacBook Air M2 (2022) with 8-core GPU. The MILP model was coded in Java and solved with a mixed-integer programming solver, Gurobi (11.0), and the time limit was set to 7200 seconds per instance. The ALNS algorithm was also coded in Java, where small instances were run for 25,000 iterations, because it was observed that additional runtime improved the solution quality; similar observations were made by Ropke and Pisinger (2006) and Keskin and Çatay (2016). Since the large instances were more computationally expensive, we performed them for 3,000 iterations and our reported results show relatively similar values to those found by Keskin and Çatay (2016).

5.2 Numerical results for standard ALNS

In order to report the numerical results for ALNS, we consider the performance of ALNS verus ALNS-SP. Note that for implementing the ALNS-SP, we established a criteria to select the subset of routes to be considered for post-processing with the SP formulation. The criteria is as follows: in a given ALNS iteration, if a solution is a local optimum such that it is better than the previous solution found by ALNS, or if it is the current global optimum such that it is better than the previous global optimum, then we input the routes from these solutions into the subset of routes for SP. We applied both ALNS and ALNS-SP on the small instances, to compare objective values and runtimes, and we present the results in Table 8 (Appendix A).

Table 8 (Appendix A) showcases that on average, ALNS-SP results in 0.53% lower objective value than ALNS, with the largest improvement at 5.27%, for the instance R102-15. This can be justified due to the fact that Gurobi solved this instance until the runtime limit, thereby providing an upper bound on the objective value, but not necessarily the optimal objective value. Hence, ALNS-SP was able to find a lower but not necessarily optimal upper bound for this instance. In most cases, ALNS-SP provides better or the same objective values as ALNS, as hypothesised in Section 4.5. Moreover, the additional runtime required for ALNS-SP relative to ALNS is 0.05 to 0.75 seconds depending on the size of the instance. Hence, at the cost of nominal runtime increase, ALNS-SP proves to be an efficient and effective matheuristic for minimising the distance travelled by EVs. Hereforth, we only present results gathered under the context of ALNS-SP.

5.2.1 Numerical results for small-size instances

In Table 2 below, we present results for the FR case solved by Gurobi, and the PR case solved by both Gurobi and ALNS-SP, for small-size instances.

Table 2: Comparison of results obtained with Gurobi and ALNS-SP on the small-size instances

Instance	FR optimal		PR Gurobi			PR ALNS-SP				
	#Veh	TD	#Veh	TD	Time (sec)	#Veh	TD	Δ^{Gurobi} %	Δ^{FR} %	Time (sec)
C101-5	3	247.15	3	247.15	0.23	3	247.15	0.00	0.00	4.35
C103-5	3	165.67	3	165.67	0.05	3	165.67	0.00	0.00	5.30
C206-5	2	236.58	2	236.58	0.05	3	236.58	0.00	0.00	7.47
C208-5	3	158.48	3	158.48	0.06	1	158.48	0.00	0.00	9.48
R104-5	2	136.69	2	136.69	0.01	2	136.45	-0.18	-0.18	3.59
R105-5	2	156.08	2	156.08	0.02	2	156.08	0.00	0.00	4.12
R202-5	1	128.78	1	128.78	0.02	1	128.78	0.00	0.00	7.02
R203-5	1	179.06	1	179.06	0.03	1	179.06	0.00	0.00	10.31
RC105-5	3	238.05	3	233.77	0.06	3	238.05	1.80	0.00	4.54
RC108-5	2	253.93	2	253.93	0.06	2	253.93	0.00	0.00	3.80
RC204-5	1	176.39	1	176.39	0.07	1	176.39	0.00	0.00	9.40
RC208-5	1	167.98	1	167.98	0.04	1	167.98	0.00	0.00	5.07
C101-10	4	393.76	4	388.25	575.43	4	383.56	-1.22	-2.66	17.26
C104-10	2	273.93	2	273.93	2.50	2	273.93	0.00	0.00	19.38
C202-10	2	243.20	2	243.20	7.85	2	243.20	0.00	0.00	34.25
C205-10	2	228.28	2	228.28	0.02	2	228.28	0.00	0.00	16.28
R102-10	3	249.19	3	249.19	1.40	4	258.03	3.43	3.43	9.66
R103-10	3	202.85	3	202.85	13.41	3	202.85	0.00	0.00	10.69
R201-10	3	217.68	3	217.68	0.08	3	217.68	0.00	0.00	42.76
R203-10	1	218.21	1	218.21	21.21	1	218.21	0.00	0.00	30.33
RC102-10	4	423.51	4	423.51	0.07	4	423.51	0.00	0.00	10.64
RC108-10	3	345.93	3	345.93	2.90	3	345.11	-0.24	-0.24	13.12
RC201-10	3	310.06	3	310.06	0.04	3	310.06	0.00	0.00	37.85
RC205-10	2	325.98	2	325.98	0.09	2	325.98	0.00	0.00	33.03
C103-15	3	371.70	3	371.70	7200.00	4	369.32	-0.64	-0.64	49.48
C106-15	3	275.13	3	275.13	0.50	3	275.13	0.00	0.00	30.06
C202-15	3	376.79	3	376.79	2300.00	3	369.56	-1.96	-1.96	75.66
C208-15	2	300.55	2	300.55	4.15	2	300.55	0.00	0.00	73.49
R102-15	5	419.64	5	419.64	7200.00	5	413.46	-1.50	-1.50	30.54
R105-15	4	336.15	4	336.15	7200.00	4	339.88	1.10	1.10	27.98
R202-15	2	358.00	2	358.00	7200.00	2	358.22	0.06	0.06	210.09
R209-15	2	293.20	2	293.20	185.33	2	293.20	0.00	0.00	126.17
RC103-15	3	397.67	3	397.67	7200.00	4	394.65	-0.77	-0.77	24.43
RC108-15	2	370.25	2	370.25	7200.00	3	375.88	1.50	1.50	31.88
RC202-15	2	394.39	2	394.39	1126.00	2	394.39	0.00	0.00	141.13
RC204-15	2	310.58	2	310.58	7200.00	2	310.58	0.00	0.00	275.32
Average					1517.82			0.04	-0.05	40.16

Considering the FR and PR schemes, Gurobi found optimal solutions for all small instances containing 5 or 10 customers, since they finished within the prescribed runtime limit of 7200 seconds. However, as expected, the runtime increases as the number of customers increase; while the average runtime for instances with 5 and 10 customers is 0.06 seconds and 52.1 seconds, the average runtime already increases significantly for the case of 15 customers; 4501.33 seconds.

For the latter instances, 7 out of 12 solutions ran up until the time limit, suggesting that we found upper bounds and not necessarily optimal solutions in the give time limit.

The results for ALNS-SP show that our procedure is effective for solving small instances with a relatively shorter average runtime - only 40.16 seconds relative to the 1517.82 seconds for Gurobi. Note that Δ^{Gurobi} % and Δ^{FR} % indicate the percentage deviation of the total distance found by the ALNS solution from the PR Gurobi and FR optimal solution.

For most of the cases, ALNS-SP was able to identify the same objective values as the FR optimal and PR Gurobi solutions. Given certain 15-customer instances for which Gurobi ran until the time limit (C103-15, R102-15, RC103-15), ALNS-SP provided a lower but not necessarily optimal upper bound on the objective values. However, for four instances such as RC105-5, R102-10, R102-15 and RC108-15, ALNS-SP provided slightly worse solutions, indicating that ALNS was perhaps trapped in a local optimum. Although the SA criterion was implemented to reduce the risk of this occurrence, it is possible that it fails to find the best solutions for certain instances. For the remaining 88.89% instances, the criterion succeeded in finding the global optimum solutions or lower upper bounds, depending on the instances under consideration.

On average, ALNS-SP provides a 0.05% improvement on FR optimal, indicating the benefit of adopting the PR scheme, which yields lower objective values on average than the FR scheme. For the case of PR Gurobi, ALNS-SP performs 0.04% worse on average. However, the trade-off with a 97.37% reduction in average computational time highlights the advantage of employing ALNS-SP. Lastly, note that FR optimal and PR Gurobi have the same number of vehicles; ALNS-SP manages to reduce the number of vehicles used for one instance (C208-5) and also increase the number of vehicles used in two cases (C103-15, RC103-15) at the benefit of lowering the objective value.

5.2.2 Numerical results for large-size instances

When studying large instances, in addition to free PR, we implement ALNS-SP for different recharging strategies, such as FR and PR where the recharge amount is fixed at certain proportions of the battery capacity; tested for the purpose of comparison. Table 9 (Appendix B) shows the results for the 56 instances with 100 customers for the different strategies, where the FR results are treated as the benchmark which the four PR policies are compared to, through a percentage deviation in total distance travelled by the EVs. Note that 'q free' permits PR at any battery level, while 'q = 0.3', 'q = 0.4', and 'q = 0.5' imply that PR must be performed at 30%, 40% and 50% of the battery capacity, respectively.

As hypothesised by Keskin and Çatay (2016), the PR strategy has benefits over the FR scheme; Table 9 (Appendix B) denotes that on average, adopting PR (q free) results in 1.69% lower distance travelled. For the fixed PR cases of q = 0.3, q = 0.4, and q = 0.5, the improvements were poorer at 0.64%, 0.79% and 0.62%, respectively. These results can be visualised by the column chart in Figure 3, which demonstrates the average percentage improvement in objective for each set of 100 customer instances (Section 5.1) contributed by various recharging policies relative to the FR policy. The performance of each policy in the improvement of the objective value can be seen below 0%, since they reduce the objective value. Policies with bars above 0% depict an increase in the objective value.

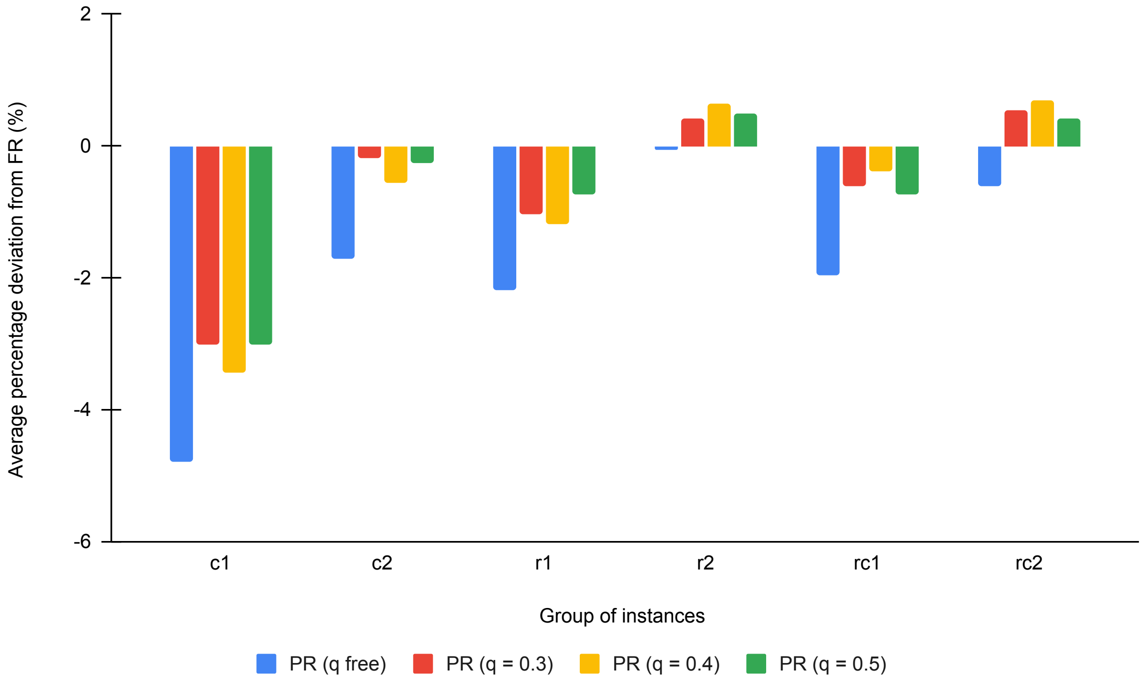


Figure 3: Column chart demonstrating the average percentage improvement in objective value per group of instances, provided by various recharging policies relative to the full recharge policy

Figure 3 showcases that the effect of the recharging policy under question varies across instances. Policies PR ($q = 0.3$), PR ($q = 0.4$) and PR ($q = 0.5$) have a similar performance trend across all groups. For groups c1, c2 and r1, PR ($q = 0.4$) outperforms the other two policies, but dominantly provides worse objective values relative to the other two policies for groups r2 and rc2. PR ($q = 0.3$) and PR ($q = 0.5$) outperform PR ($q = 0.4$) in the case of rc1 by further reducing the objective value, and also outperform PR ($q = 0.4$) for the r2 and rc2 instances by providing a lower increase in the objective value.

Conversely, the PR (q free) appears to outperform the other policies by demonstrating substantial negative contributions for each group of instances, indicating that allowing free PR can significantly reduce the objective value on average. There are two cases where this policy contributes positively to increasing the objective value, minimally in r201 by 0.07% and dominantly in rc206 by 1.33%, as shown in Table 9 (Appendix B). Nonetheless, overall, free PR can be concluded to be the best recharging policy when considering ALNS-SP.

5.3 Numerical results for ALNS with Labeling Algorithm

Concerning the *labeling algorithm* in the ALNS-SP framework, we consider the small-size instances to test for the most effective recharging policy under the *labeling algorithm*, and secondly for evaluating whether the ALNS-SP with or without labeling algorithm is better, both in terms of objective values and computation times. Firstly, we consider the objective values across all policies, as shown below in Table 3. FR, PR ($q = 0.4$), PR ($q = 0.6$), PR ($q = 0.8$) and PR (2 customers) all pertain to ALNS-SP with the *labeling algorithm*, while ALNS-SP PR refers to the standard ALNS-SP with free recharge quantity introduced in Section 5.2.2.

Table 3: Comparison of results for different recharge strategies under ALNS with Labeling Algorithm against the ALNS-SP free PR

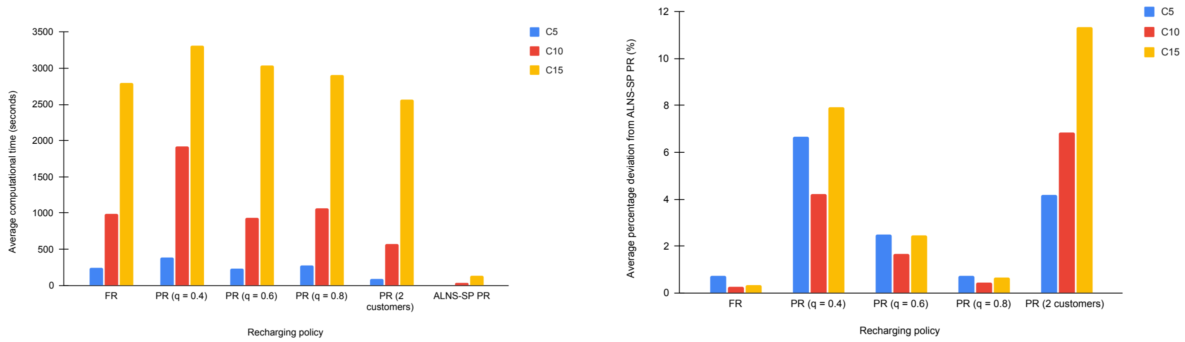
Instance	ALNS-SP PR		FR			PR (q = 0.4)			PR (q = 0.6)			PR (q = 0.8)			PR (2 customers)		
	#Veh	TD	#Veh	TD	Δ%	#Veh	TD	Δ%	#Veh	TD	Δ%	#Veh	TD	Δ%	#Veh	TD	Δ%
C101-5	3	247.15	3	247.15	0.00	3	247.15	0.00	3	247.15	0.00	3	247.15	0.00	4	250.04	1.16
C103-5	3	165.67	3	165.67	0.00	3	165.67	0.00	3	165.67	0.00	3	165.67	0.00	3	165.67	0.00
C206-5	3	236.58	3	236.58	0.00	3	244.30	3.16	3	236.58	0.00	3	236.58	0.00	3	241.49	2.04
C208-5	1	158.48	1	164.34	3.57	1	199.47	20.55	1	164.34	3.57	1	164.34	3.57	1	174.82	9.35
R104-5	2	136.45	2	136.69	0.18	2	136.69	0.18	2	136.69	0.18	2	136.69	0.18	2	136.69	0.18
R105-5	2	156.08	2	156.08	0.00	3	182.92	14.67	3	182.92	14.67	2	156.08	0.00	3	168.47	7.35
R202-5	1	128.78	1	128.88	0.08	2	143.39	10.19	1	128.88	0.08	1	128.88	0.08	1	146.77	12.26
R203-5	1	179.06	1	179.06	0.00	2	208.19	13.99	1	191.61	6.55	1	179.06	0.00	1	197.99	9.56
RC105-5	3	238.05	3	238.05	0.00	3	238.05	0.00	3	238.05	0.00	3	238.05	0.00	3	238.05	0.00
RC108-5	2	253.93	2	253.93	0.00	2	264.92	4.15	2	253.93	0.00	2	253.93	0.00	2	253.93	0.00
RC204-5	1	176.39	2	185.16	4.73	2	185.44	4.88	2	185.16	4.73	2	185.16	4.73	2	185.16	4.73
RC208-5	1	167.98	1	167.98	0.00	1	182.79	8.10	1	167.98	0.00	1	167.98	0.00	1	174.38	3.67
C101-10	4	383.56	4	399.31	3.95	4	409.65	6.37	4	399.31	3.95	4	399.31	3.95	3	401.97	4.58
C104-10	2	273.93	2	273.93	0.00	2	300.50	8.84	2	276.09	0.78	2	273.93	0.00	2	308.51	11.21
C202-10	2	243.20	2	243.20	0.00	2	250.88	3.06	2	243.21	0.00	2	243.21	0.00	3	270.05	9.94
C205-10	2	228.28	2	228.28	0.00	2	228.91	0.28	2	228.91	0.28	2	228.28	0.00	2	234.52	2.66
R102-10	4	258.03	3	249.19	-3.55	3	255.88	-0.84	3	255.88	-0.84	3	249.19	-3.55	4	262.92	1.86
R103-10	3	202.85	3	202.85	0.00	3	202.85	0.00	3	202.85	0.00	3	202.85	0.00	3	203.60	0.37
R201-10	3	217.68	3	217.68	0.00	3	227.41	4.28	3	224.54	3.05	3	217.68	0.00	4	239.99	9.30
R203-10	1	218.21	1	222.64	1.99	-	-	-	1	235.15	7.20	1	222.64	1.99	2	263.87	17.31
RC102-10	4	423.51	4	423.51	0.00	5	437.59	3.22	4	423.51	0.00	4	423.51	0.00	5	436.05	2.88
RC108-10	3	345.11	3	347.90	0.80	4	395.45	12.73	3	352.09	1.98	3	347.90	0.80	3	347.90	0.80
RC201-10	3	310.06	3	310.06	0.00	3	311.14	0.35	3	310.06	0.00	3	310.06	0.00	4	341.65	9.25
RC205-10	2	325.98	2	325.98	0.00	3	355.00	8.17	2	337.89	3.53	2	332.99	2.11	3	370.51	12.02
C103-15	4	369.32	4	371.70	0.64	4	371.70	0.64	4	371.70	0.64	4	371.70	0.64	4	371.70	0.64
C106-15	3	275.13	3	275.13	0.00	4	327.19	15.91	3	281.82	2.37	3	275.13	0.00	4	320.37	14.12
C202-15	3	369.56	3	369.56	0.00	4	398.22	7.20	3	376.79	1.92	3	369.56	0.00	4	428.36	13.73
C208-15	2	300.55	2	300.55	0.00	3	325.75	7.74	2	308.45	2.56	2	300.55	0.00	3	370.46	18.87
R102-15	5	413.46	5	413.93	0.11	6	438.88	5.79	5	420.10	1.58	6	419.64	1.47	6	432.96	4.50
R105-15	4	339.88	4	336.15	-1.11	4	340.17	0.09	4	336.15	-1.11	4	336.15	-1.11	4	352.39	3.55
R202-15	2	358.22	3	359.08	0.24	3	382.93	6.45	3	383.58	6.61	3	359.08	0.24	3	411.57	12.96
R209-15	2	293.20	2	307.68	4.71	4	390.22	24.86	2	332.79	11.90	3	316.75	7.43	3	380.75	22.99
RC103-15	4	394.65	4	394.65	0.00	4	400.10	1.36	4	394.65	0.00	4	394.65	0.00	5	401.56	1.72
RC108-15	3	375.88	3	370.25	-1.52	5	471.77	20.32	3	387.93	3.10	3	370.25	-1.52	5	458.15	17.96
RC202-15	2	394.39	3	397.20	0.71	3	401.79	1.84	2	394.39	0.00	3	397.20	0.71	4	436.47	9.64
RC204-15	2	310.58	2	310.58	0.00	2	319.41	2.77	2	310.58	0.00	2	310.58	0.00	2	367.60	15.51
Average					**0.43**			6.32			2.2			0.6			7.46

From Table 3 we observe that different recharging policies have significantly different implications for their effect on the total distance travelled by EVs. FR appears to provide relatively similar distances as the ALNS-SP, only increasing the objective values by 0.43% on average. The next best policy is that of fixed PR at q = 0.8, only worsening the objective value by 0.6% on average. PR (q = 0.6), PR (q = 0.4) and PR (2 customers) perform 2.2%, 6.32% and 7.46% worse on average than ALNS-SP PR. Moreover, instance R203-10 was infeasible with PR (q = 0.4), implying that this policy results in insufficient battery levels for EVs to successfully complete their routes.

The overall trend observed in the performance of fixed recharging policies is as follows: as the fixed recharge amount decreases, the average deviation of the objective values from the ALNS-SP increases. This indicates contrasting results to the standard ALNS with SI and SR operators, as seen in Section 5.2.2, where FR was the worst-performing policy. One possible explanation for this varying policy performance is the dominance rules in the *labeling algorithm*, which prioritise labels with higher battery SoC and earlier arrival times. Full recharge allows for higher SoC when leaving a CS, such that the vehicle can travel longer distances and cover more customers without requiring another recharge. Therefore, from the policies we tested for the *labeling algorithm*, implementing FR closely mimics the effect of free PR under the standard

ALNS-SP, through ensuring sufficient battery levels across routes and thereby achieving the most similar objective values.

We now study the average runtimes per group of instances for both types of algorithms under different recharging policies, provided in Table 10 (Appendix C) and visualised in Figure 4a. We also illustrate the average objective values per group of instances under different recharging policies for the *labeling algorithm* relative to the ALNS-SP PR, in Figure 4b, facilitating an evaluation of the models based on both runtime and objective values.



(a) Average computational time

(b) Average objective values

Figure 4: Comparison of average objective values and computational times for various recharging policies and instance sizes

Figure 4a showcases that the computational time increases with the number of customers. The C15 instances consistently exhibit the highest computational times across all recharging policies, followed by C10 and C5 instances. This trend demonstrates the increased complexity and computational effort required to handle larger customer groups, with PR ($q = 0.4$) showcasing the worst runtimes and ALNS-SP PR denoting the best runtimes across all instances, with a remarkably lower average runtime of 40.16 seconds (Table 2). This highlights the efficiency of the ALNS-SP approach in managing computational time. The second and third best runtimes, as seen in Table 10 (Appendix C), are attributed to PR (2 customers) with an average runtime of 1075.83 seconds, followed by FR at 1344.14 seconds.

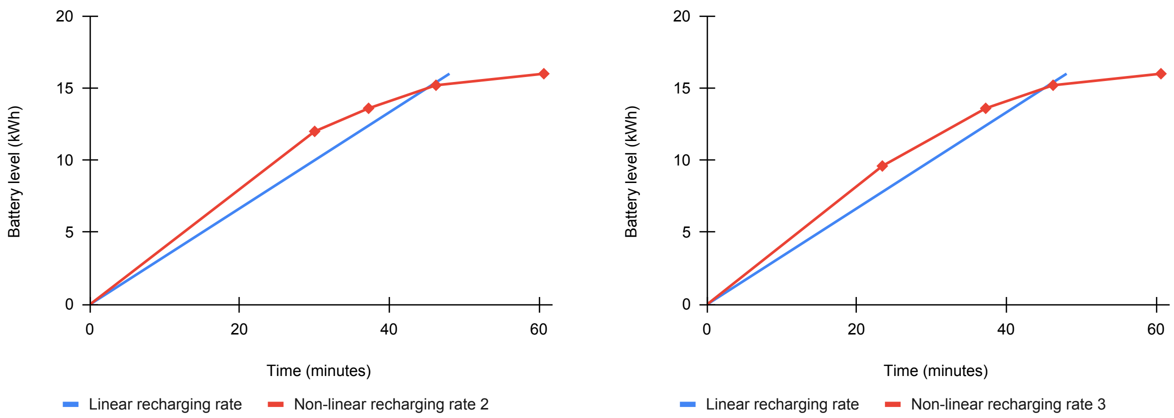
When considering objective values, PR (2 customers) with the second-best runtime performs the worst relative to the standard ALNS-SP across all instance groups, as seen in Figure 4b. On the other hand, the FR policy managed to have the third-best computational time and demonstrated distances comparable to those of the ALNS-SP PR. Therefore, for ALNS with the *labeling algorithm*, FR policy appears to provide the best trade-off between runtime and objective value.

Nonetheless, on average, the FR policy performs 0.43% worse than ALNS-SP in terms objective value, and is approximately 32 times slower than ALNS-SP's average runtime. Therefore, we will proceed with our standard setup of ALNS-SP with customer and station destroy-repair operators, for exploring the effects of linear versus non-linear recharging, as well as multiple recharging technologies.

5.4 Linear versus Non-Linear Recharging

5.4.1 Recharging Functions

In the non-linear scenario, we utilise the recharging function presented by Montoya et al. (2017), as mentioned in Section 3.1. For the case of an EV with a battery capacity of 16 kWh and constant recharge rate of 3 min/kWh, we considered three non-linear recharging functions. The first non-linear recharging function was shown in Figure 1, with breakpoints at 0%, 85%, 95% and 100% of the battery capacity, as suggested by Montoya et al. (2017). To examine whether non-linearity in recharging functions has any effect on objective values, we consider two additional recharging functions with additional breakpoints at 75% and 60% of the battery capacity respectively, as shown in Figure 5 below.



(a) Non-linear recharging function 2

(b) Non-linear recharging function 3

Figure 5: Other types of non-linear recharging functions considered for analysis

Figure 5a displays breakpoints at (0, 0), (30, 12), (37.2, 13.6), (46.2, 15.2) and (60.6, 16); at 0%, 75%, 85%, 95% and 100% of the battery capacity. Figure 5b illustrates breakpoints at (0, 0), (23.4, 9.6), (37.2, 13.6), (46.2, 15.2) and (60.6, 16); at 0%, 60%, 85%, 95% and 100% of the battery capacity. Given a constant recharge rate (3 min/kWh), we can calculate the slopes (recharging rates) of the different segments within the piecewise-linear function and scale them for different instances based on the varying battery capacities. Table 4 illustrates the scaling faction for non-linear recharging rates at various breakpoints, given a linear recharging rate r.

Table 4: Recharging rates at various segments of the piecewise-linear recharging functions

Non-linear functions	Non-linear recharging segments used						
	0%-60%	60%-85%	0%-75%	75%-85%	0%-85%	85%-95%	95%-100%
Function 1	-	-	-	-	0.91*r	1.875*r	6*r
Function 2	-	-	0.83*r	1.5*r	-	1.875*r	6*r
Function 3	0.81*r	1.15*r	-	-	-	1.875*r	6*r

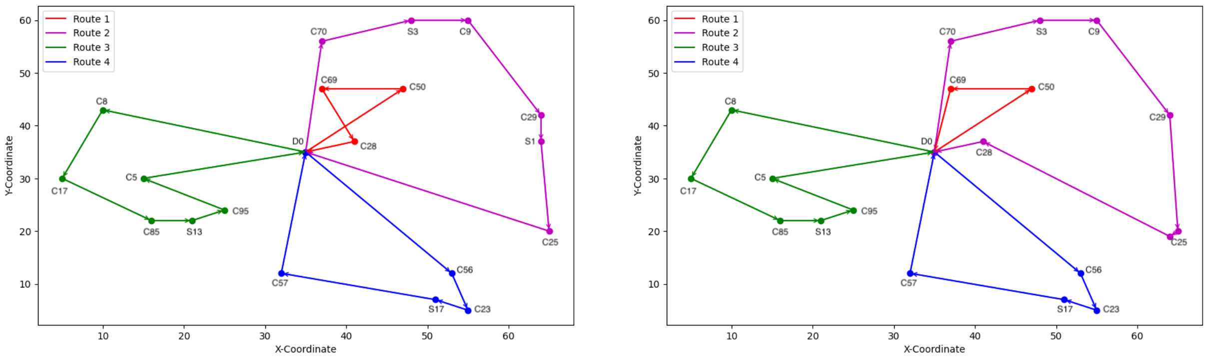
5.4.2 Results for linear versus non-linear recharging

We now evaluate the effect of linear against non-linear recharging on the objective values, as shown below in Table 5.

Table 5: Comparison of results under linear and non-linear recharging rates

Instance	Linear		Non-linear: 85%-100%			Non-linear: 75%-100%			Non-linear: 60%-100%		
	#Veh	TD	#Veh	TD	Δ%	#Veh	TD	Δ%	#Veh	TD	Δ%
C101-5	3	247.15	3	247.15	0.00	3	247.15	0.00	3	247.15	0.00
C103-5	3	165.67	3	165.67	0.00	3	165.67	0.00	3	165.67	0.00
C206-5	3	236.58	3	236.58	0.00	3	236.58	0.00	3	236.58	0.00
C208-5	1	158.48	1	158.48	0.00	1	158.48	0.00	1	158.48	0.00
R104-5	2	136.45	2	136.45	0.00	2	136.69	0.18	2	136.69	0.18
R105-5	2	156.08	2	156.08	0.00	2	156.08	0.00	2	156.08	0.00
R202-5	1	128.78	1	128.78	0.00	1	128.78	0.00	1	128.78	0.00
R203-5	1	179.06	1	179.06	0.00	1	179.06	0.00	1	179.06	0.00
RC105-5	3	238.05	3	238.05	0.00	3	238.05	0.00	3	238.05	0.00
RC108-5	2	253.93	2	253.93	0.00	2	253.93	0.00	2	253.93	0.00
RC204-5	1	176.39	1	176.39	0.00	1	176.39	0.00	1	176.39	0.00
RC208-5	1	167.98	1	167.98	0.00	1	167.98	0.00	1	167.98	0.00
C101-10	4	383.56	3	393.56	2.54	3	393.56	2.54	3	393.56	2.54
C104-10	2	273.93	2	273.93	0.00	2	273.93	0.00	2	273.93	0.00
C202-10	2	243.20	2	243.20	0.00	2	243.20	0.00	2	243.20	0.00
C205-10	2	228.28	2	228.28	0.00	2	228.28	0.00	2	228.28	0.00
R102-10	4	258.03	4	262.93	1.86	4	262.93	1.86	4	262.93	1.86
R103-10	3	202.85	3	202.85	0.00	3	202.85	0.00	3	202.85	0.00
R201-10	3	217.68	3	217.68	0.00	3	216.60	-0.50	3	217.68	0.00
R203-10	1	218.21	1	218.21	0.00	1	218.21	0.00	1	218.21	0.00
RC102-10	4	423.51	4	423.51	0.00	4	423.51	0.00	4	423.51	0.00
RC108-10	3	345.11	3	345.93	0.24	3	345.93	0.24	3	345.93	0.24
RC201-10	3	310.06	3	310.06	0.00	3	310.06	0.00	3	310.06	0.00
RC205-10	2	325.98	2	325.98	0.00	2	325.98	0.00	2	325.98	0.00
C103-15	4	369.32	4	369.45	0.04	4	369.45	0.04	4	369.45	0.04
C106-15	3	275.13	3	275.13	0.00	3	275.13	0.00	3	275.13	0.00
C202-15	3	369.56	3	369.56	0.00	3	369.56	0.00	-	-	-
C208-15	2	300.55	2	300.55	0.00	2	300.55	0.00	2	300.55	0.00
R102-15	5	413.46	5	413.46	0.00	5	413.46	0.00	5	412.78	-0.16
R105-15	4	339.88	4	336.15	-1.11	4	337.36	-0.75	4	336.15	-1.11
R202-15	2	358.22	2	358.22	0.00	2	358.22	0.00	3	359.08	0.24
R209-15	2	293.20	2	293.20	0.00	2	293.20	0.00	2	293.20	0.00
RC103-15	4	394.65	4	394.65	0.00	4	394.65	0.00	4	394.65	0.00
RC108-15	3	375.88	3	378.36	0.65	3	378.36	0.65	3	378.36	0.65
RC202-15	2	394.39	2	394.39	0.00	2	394.39	0.00	2	394.39	0.00
RC204-15	2	310.58	2	310.58	0.00	2	310.58	0.00	2	310.58	0.00
Average					0.12			0.12			0.13

The results from implementing non-linear recharging functions show relatively similar results to those from the linear recharging function; on average, they differ by only 0.12% - 0.13%. Focusing on individual instances, the first and third non-linear recharging functions provide the largest improvement in objective value for R105-15, at 1.11%.



(a) Linear recharging function

(b) Non-linear recharging function

Figure 6: Comparison of routes with different recharging functions for R105-15

Upon inspecting the routes for R105-15 in Figure 6, we infer that in the non-linear case, route 2 manages to incorporate another customer C28 on the route who was not visited on route 2 in the linear case. This is because in the non-linear case, the EV does not need to stop at both stations S3 and S1 to recharge. Instead, it only needs to stop at S3 to top up its battery and it has sufficient time to do so, because to recharge until 95% of the battery capacity, the non-linear recharge rate is quicker than the linear rate (Figure 5). Hence, this enables the EV to meet the time window of customer C28; these changes reduce the overall solution cost for this instance. On the contrary, all non-linear recharging functions provide the worst increase in objective value, 2.54%, than the linear case for C101-10. The non-linear case can accommodate all customers in 3 instead of 4 vehicles used by the linear case, again due to the quicker recharge rate until 95% of the battery capacity. However, in this case reducing the number of vehicles provides a higher solution cost.

Overall, despite these case-by-case differences between linear and non-linear recharging functions, the average effect across all instances is relatively small. This can be attributed to the fact that the critical non-linear recharging segment of 95% - 100% battery capacity, which has a substantially slower recharge rate than the linear rate and thus can significantly alter the routes, is never utilised for any instance. Table 6 illustrates this finding.

Table 6: Each type of non-linear recharging segment used as a percentage of the stations visited

Functions	# Stations used	Non-linear recharging segments used			
		60%-75%	75%-85%	85%-95%	95%-100%
Function 1	104	-	-	26 (25%)	0
Function 2	108	-	40 (37.04%)	0	0
Function 3	98	58 (59.18%)	-	0	0

In fact, for each type of non-linear recharging function, only the initial non-linear segments are used. For example, for non-linear recharging function 1, only 25% of station visits involve recharging the battery to a level between 85% - 95%, and none of the visits involve charging between 95% - 100%. Similar patterns follow for functions 2 and 3. This outcome can be explained by the PR policy which ensures that EVs only charge as much as needed, and also ALNS operators such as full charge station removal, which prioritises the removal of stations with the highest recharge amounts. Consequently, the battery levels never reach the full recharge segment, making the non-linear recharging behaviour not significantly different from the linear case. Therefore, we proceed with studying the effect of multiple chargers by using the standard ALNS-SP with a linear recharge rate.

5.5 Multiple recharging technologies

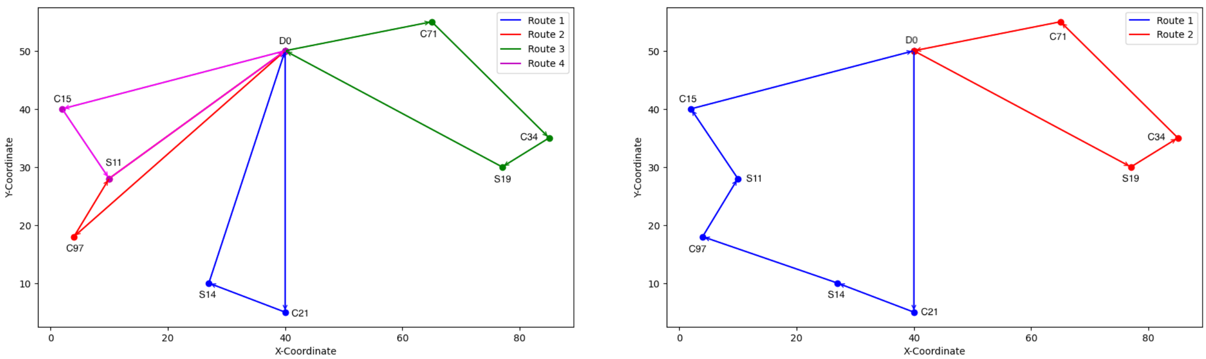
All the original instances considered a single-type charger setting, where different instances had different constant recharging rates, ranging from 0.18 to 3.47 minutes/kWh. To transition into the setting of multiple chargers, we referred to Felipe et al. (2014), wherein they described three recharging technologies: slow (16.67 min/kWh), medium (3 min/kWh) and fast (1.33 min/kWh). However, it is not possible to implement these rates in our case, due to the presence of time windows for customers and the end depot - each route has a limited duration. The above-mentioned rates make most of the instances infeasible due to extensive recharging times violating the duration of the routes. Therefore, we limit the number of infeasible instances to 3 (RC102-10, RC108-10, R102-15), by picking the maximum recharge rate provided in the instances, 3.47 min/kWh, and treating this as the recharge rate for the slowest charger. Based on this rate and according to the values from Felipe et al. (2014), we then scale the medium and fast recharge rates to be 0.62 min/kWh and 0.28 min/kWh respectively. Furthermore, we also use the following recharge costs presented by Felipe et al. (2014): 0.16 €/kWh, 0.176 €/kWh, 0.192 €/kWh for slow, medium and fast technologies respectively.

Table 7 below demonstrates sensitivity analysis for the different recharging technologies relative to the slow charger, where the slow, medium and fast chargers and denoted as 1, 2 and 3 respectively. We observe a reduction in objective value for all combinations of chargers relative to the slow charger. The smallest reduction at 6.56% is attributed to MC = {1,2}, when only slow and medium chargers are implemented. On the other hand, the highest reduction at 8.32% stems from MC = 3; by only using the fast charger. To illustrate the benefits of using the fast charger relative to the slow charger, we focus on instance RC108-5 which had the largest decrease in objective value at 55.24%. Figure 7 below showcases that the vehicle following route 1 in the fast charger case is able to combine routes 2, 3 and 4 from the slow charger case. This is because due to faster recharging times, the EV has freed up time which it can use to visit more customers on the route, thereby halving the number of EVs required to cover the same set of customers.

However, a counter-intuitive finding is that even though MC = {1,2,3} offers a larger feasible region for ALNS than just MC = 3, restricting the availability of chargers to MC = 3 reduces the objective value by 0.56% more than if all technologies were available. It could be the case that the fast charger is indeed most cost-effective compared to the slow and medium chargers. However, due to the inclusion of slow and medium chargers, ALNS may be getting trapped in worse solutions involving these technologies, indicating the suboptimality of our ALNS approach for solving EVRPTW-NLPR-MC. Therefore, we can conclude that using multiple charger technologies does have a substantial effect on objective value, but additional destroy-repair operators specific to the case of multiple chargers may need to be added to ALNS to improve its performance in this case.

Table 7: Sensitivity analysis for different recharging technologies relative to the slow charger

Instance	MC = 1		MC = 2			MC = 3			MC = {1,2}			MC = {1,3}			MC = {2,3}			MC = {1,2,3}		
	#Veh	TC	#Veh	TC	Δ%	#Veh	TC	Δ%	#Veh	TC	Δ%	#Veh	TC	Δ%	#Veh	TC	Δ%	#Veh	TC	Δ%
C101-5	3	247.15	3	247.15	0.00	3	247.15	0.00	3	247.15	0.00	3	247.15	0.00	3	247.15	0.00	3	247.15	0.00
C103-5	3	165.67	3	165.67	0.00	3	165.67	0.00	3	165.67	0.00	3	165.67	0.00	3	165.67	0.00	3	165.67	0.00
C206-5	3	236.58	2	236.58	0.00	2	236.21	-0.16	3	236.58	0.00	3	236.58	0.00	2	236.21	-0.16	3	236.58	0.00
C208-5	1	158.48	1	158.48	0.00	1	158.48	0.00	1	158.48	0.00	1	158.48	0.00	1	158.48	0.00	1	158.48	0.00
R104-5	2	136.69	2	136.69	0.00	2	136.69	0.00	2	136.69	0.00	2	136.69	0.00	2	136.69	0.00	2	136.69	0.00
R105-5	3	182.92	2	156.08	-17.19	2	156.08	-17.19	2	156.08	-17.19	2	156.08	-17.19	2	156.08	-17.19	2	156.08	-17.19
R202-5	2	142.65	1	128.78	-10.78	1	128.78	-10.78	2	142.65	0.00	1	128.88	-10.69	1	128.78	-10.78	1	128.88	-10.69
R203-5	2	195.63	1	179.06	-9.26	1	179.06	-9.26	1	179.06	-9.26	1	179.06	-9.26	1	179.06	-9.26	1	179.06	-9.26
RC105-5	3	238.05	3	238.05	0.00	3	238.05	0.00	3	238.05	0.00	3	238.05	0.00	3	238.05	0.00	3	238.05	0.00
RC108-5	4	394.21	2	253.93	-55.24	2	253.93	-55.24	3	316.51	-24.55	2	262.27	-50.31	2	253.93	-55.24	2	262.27	-50.31
RC204-5	1	179.16	1	176.39	-1.57	1	176.39	-1.57	1	176.39	-1.57	1	176.39	-1.57	1	176.39	-1.57	1	176.39	-1.57
RC208-5	1	167.98	1	167.98	0.00	1	167.98	0.00	1	167.98	0.00	1	167.98	0.00	1	167.98	0.00	1	167.98	0.00
C101-10	4	386.76	3	378.01	-2.31	3	378.01	-2.31	3	378.01	-2.31	3	378.01	-2.31	3	378.01	-2.31	3	378.01	-2.31
C104-10	2	273.93	2	266.16	-2.92	1	257.66	-6.31	2	273.93	0.00	2	273.93	0.00	2	266.16	-2.92	2	273.93	0.00
C202-10	2	243.20	2	243.20	0.00	2	243.20	0.00	2	243.20	0.00	2	243.20	0.00	2	243.20	0.00	2	243.20	0.00
C205-10	2	228.28	2	228.28	0.00	2	228.28	0.00	2	228.28	0.00	2	228.28	0.00	2	228.28	0.00	2	228.28	0.00
R102-10	6	361.49	4	262.92	-37.49	4	262.92	-37.49	4	262.92	-37.49	4	262.92	-37.49	4	262.92	-37.49	4	262.92	-37.49
R103-10	3	202.85	3	202.85	0.00	3	202.85	0.00	3	202.85	0.00	3	202.85	0.00	3	202.85	0.00	3	202.85	0.00
R201-10	3	227.28	3	217.68	-4.41	3	217.68	-4.41	3	216.60	-4.93	3	216.60	-4.93	3	217.68	-4.41	3	217.68	-4.41
R203-10	2	232.68	1	218.21	-6.63	1	218.21	-6.63	1	222.64	-4.51	1	218.21	-6.63	1	218.21	-6.63	1	220.53	-5.51
RC102-10	-	-	5	436.05	-	4	412.93	-	5	436.05	-	4	412.93	-	4	412.93	-	4	412.93	-
RC108-10	-	-	3	345.93	-	3	345.93	-	3	345.93	-	3	345.93	-	3	345.93	-	3	345.93	-
RC201-10	3	310.06	3	310.06	0.00	3	310.06	0.00	3	310.06	0.00	3	310.06	0.00	3	310.06	0.00	3	310.06	0.00
RC205-10	3	350.19	2	325.98	-7.43	2	325.98	-7.43	2	325.98	-7.43	2	325.98	-7.43	2	325.98	-7.43	2	325.98	-7.43
C103-15	4	369.32	3	348.45	-5.99	3	348.53	-5.97	3	350.01	-5.52	3	350.02	-5.52	3	348.53	-5.97	3	350.10	-5.49
C106-15	3	275.13	3	271.21	-1.45	3	271.21	-1.45	3	271.21	-1.45	3	271.21	-1.45	3	271.21	-1.45	3	271.21	-1.45
C202-15	3	369.56	3	369.56	0.00	3	369.56	0.00	3	369.56	0.00	3	369.56	0.00	3	369.56	0.00	3	369.56	0.00
C208-15	2	300.41	2	300.55	0.05	2	300.55	0.05	2	300.55	0.05	2	300.55	0.05	2	300.55	0.05	2	300.55	0.05
R102-15	-	-	5	413.46	-	5	412.78	-	5	412.78	-	5	413.46	-	5	413.46	-	5	413.46	-
R105-15	7	443.03	4	338.54	-30.87	4	336.94	-31.49	4	338.54	-30.87	4	336.15	-31.80	4	337.36	-31.33	4	339.88	-30.35
R202-15	3	361.14	2	358.22	-0.81	3	359.08	-0.57	3	360.16	-0.27	3	359.08	-0.57	2	358.22	-0.81	3	360.16	-0.27
R209-15	3	327.83	2	293.20	-11.81	2	293.20	-11.81	2	301.30	-8.80	2	301.30	-8.80	2	293.20	-11.81	2	301.30	-8.80
RC103-15	5	442.85	4	394.65	-12.21	4	393.39	-12.57	4	394.65	-12.21	4	393.39	-12.57	4	394.65	-12.21	4	394.65	-12.21
RC108-15	6	564.77	3	386.58	-46.09	3	378.23	-49.32	3	386.58	-46.09	3	378.23	-49.32	3	378.23	-49.32	3	378.23	-49.32
RC202-15	3	405.44	2	394.39	-2.80	2	394.39	-2.80	3	397.20	-2.08	3	397.20	-2.08	2	394.39	-2.80	3	397.20	-2.08
RC204-15	2	310.58	2	310.58	0.00	2	310.58	0.00	2	310.58	0.00	2	310.58	0.00	2	310.58	0.00	2	310.58	0.00
Average					-8.10			**-8.32**			-6.56			-7.87			-8.21			-7.76



(a) Slow charger (b) Fast charger

Figure 7: Comparison of routes with different chargers for RC108-5

6 Conclusion

In this paper, we introduced the electric vehicle routing problem with time windows, non-linear partial recharging and multiple chargers (EVRPTW-NLPR-MC). We proposed a matheuristic to solve this problem, which combines an adaptive large neighbourhood search with a set partitioning formulation (ALNS-SP). Additionally, we investigated an alternative method by introducing a *labeling algorithm* for station insertion within the framework of ALNS-SP, thereby replacing the station destroy-repair operators. From a modelling perspective, we focused incorporating non-linear recharging functions to estimate recharging times more realistically, and also inspected the effect of using multiple charger technologies on the solution cost. We performed a computational study utilising the instances generated by (Schneider et al., 2014) to evaluate the following: the performance of ALNS versus ALNS-SP, and the performance of the standard versus alternative ALNS-SP with the labeling algorithm. We then modified these instances by adjusting the recharging rates based on the non-linear recharging function specification as well as multiple chargers. Finally, we evaluated ALNS-SP under two scenarios: linear and non-linear recharging, and then conducted sensitivity analysis for multiple chargers.

Firstly, we observed that ALNS-SP always provides the same or lower objective value as ALNS at minimal additional computation time. Hence, averaging an improvement in objective value by 0.53%, and an increase in average runtime by 0.25 seconds, we conclude that ALNS-SP outperforms ALNS. Moreover, ALNS-SP reduced the number of vehicles used for one instance, and increased the number of vehicles used for two instances to achieve a lower objective value. Secondly, the *labeling algorithm* was successfully implemented for 5 different recharging policies observed in literature. We found that the best recharging policy for the *labeling algorithm* was full recharge (FR), contrary to ALNS-SP which implemented partial recharge (PR). We believe that this may be due to the design of the dominance rules within the *labeling algorithm*, prioritising labels with higher arrival battery levels. Overall, the standard ALNS-SP outperformed ALNS-SP with labeling algorithm and FR policy, with 0.43% lower objective value on average and 32 times faster computational time.

Furthermore, we find that non-linear recharging provides very similar objective values to linear recharging, differing only by 0.12% on average. An investigation of the battery recharge quantities at stations showed that the quantities never hit 95% - 100% battery capacity for any instance. This can be explained by the PR policy ensuring that EVs do not charge more than necessary, and also the ALNS station operators, such as full charge station removal, which removes stations with the highest recharge quantities. Since only this critical 95% - 100% battery segment had a substantially slower recharge rate than the linear rate and was never utilised, we conclude that for our set-up, non-linear recharging is not significantly different than the linear recharging case. Proceeding with ALNS-SP and linear recharging, our sensitivity analysis for multiple chargers showed that introducing these different technologies does indeed affect the solution cost. Fast chargers are the dominant technology, since they reduce the objective value by 8.32% on average, despite being the most expensive technology. This outcome can be explained by the fact that quicker recharging times allow EVs to service more customers on the route, thereby reducing the number of vehicles used and also the total cost of the solution. However, there is on average 0.56% difference in cost savings when using fast chargers versus all three chargers. Since the feasible region provided by fast chargers is contained within the feasible region for all three chargers, this indicates the suboptimality of our ALNS-SP for solving the multiple chargers case.

Therefore, directions for future research from a methodological perspective include improving the performance of ALNS-SP for multiple chargers, by incorporating operators specific to this case, such that even if all three technologies are available, the fast charger should always be used if it is indeed the optimal recharging technology. Secondly, the *labeling algorithm* showed potential through achieving relatively similar objective values to the standard ALNS-SP; hence, one could investigate improving the computational time of the algorithm. This could be executed through creating labels more efficiently, such as by generating bidirectional labels simultaneously from both the start and end of the route. One could also modify the *labeling algorithm* to study the effect of combining various recharge policies within label creation on the solution cost. Lastly, in this study we explore electric vehicles with a constant rate of energy consumption. Hence, one might consider extending this study to a heterogeneous fleet case with a more realistic model for energy consumption which includes traffic congestion, waiting times, vehicle load and vehicle speed.

References

Bruglieri, M., Colorni, A. & Lue, A. (2014). The vehicle relocation problem for the one-way electric vehicle sharing: an application to the milan case. *Procedia-Social and Behavioral Sciences*, *111*, 18–27.

Conrad, R. G. & Figliozzi, M. A. (2011). The recharging vehicle routing problem. In *Proceedings of the 2011 industrial engineering research conference* (Vol. 8).

Demir, E., Bektaş, T. & Laporte, G. (2012). An adaptive large neighborhood search heuristic for the pollution-routing problem. *European journal of operational research*, *223*(2), 346–359.

Dönmez, S., Koç, Ç. & Altıparmak, F. (2022). The mixed fleet vehicle routing problem with partial recharging by multiple chargers: Mathematical model and adaptive large neighborhood search. *Transportation Research Part E: Logistics and Transportation Review*, *167*, 102917.

Erdoğan, S. & Miller-Hooks, E. (2012). A green vehicle routing problem. *Transportation research part E: logistics and transportation review*, *48*(1), 100–114.

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

Froger, A., Mendoza, J. E., Jabali, O. & Laporte, G. (2019). Improved formulations and algorithmic components for the electric vehicle routing problem with nonlinear charging functions. *Computers & Operations Research*, *104*, 256–294.

Goeke, D. & Schneider, M. (2015). Routing a mixed fleet of electric and conventional vehicles. *European Journal of Operational Research*, *245*(1), 81–99.

Golden, B. L., Raghavan, S. & Wasil, E. A. (2008). *The vehicle routing problem: latest advances and new challenges* (Vol. 43). Springer Science & Business Media.

Gunawan, A., Widjaja, A. T., Vansteenwegen, P. & Yu, V. F. (2020). A matheuristic algorithm for solving the vehicle routing problem with cross-docking. In *Learning and intelligent optimization: 14th international conference, lion 14, athens, greece, may 24–28, 2020, revised selected papers 14* (pp. 9–15).

Hiermann, G., Puchinger, J., Ropke, S. & Hartl, R. F. (2016). The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal of Operational Research*, *252*(3), 995–1018.

Hõimoja, H., Rufer, A., Dziechciaruk, G. & Vezzini, A. (2012). An ultrafast ev charging station demonstrator. In *International symposium on power electronics power electronics, electrical drives, automation and motion* (pp. 1390–1395).

Karakatič, S. (2021). Optimizing nonlinear charging times of electric vehicle routing with genetic algorithm. *Expert Systems with Applications*, *164*, 114039.

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.

Keskin, M. & Çatay, B. (2018). A matheuristic method for the electric vehicle routing problem with time windows and fast chargers. *Computers & operations research*, *100*, 172–188.

Lee, C. (2021). An exact algorithm for the electric-vehicle routing problem with nonlinear charging time. *Journal of the Operational Research Society*, *72*(7), 1461–1485.

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

Paschero, M., Anniballi, L., Del Vescovo, G., Fabbri, G. & Mascioli, F. M. F. (2013). Design and implementation of a fast recharge station for electric vehicles. In *2013 ieee international symposium on industrial electronics* (pp. 1–6).

Pelletier, S., Jabali, O., Laporte, G. & Veneroni, M. (2017). Battery degradation and behaviour for electric vehicles: Review and numerical analyses of several models. *Transportation Research Part B: Methodological*, *103*, 158–187.

Ropke, S. & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, *40*(4), 455–472.

Schneider, M., Stenger, A. & Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. *Transportation science*, *48*(4), 500–520.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, *35*(2), 254–265. Retrieved 2024-05-17, from http://www.jstor.org/stable/170697

Toth, P. & Vigo, D. (2002). *The vehicle routing problem.* SIAM.

Touati-Moungla, N. & Jost, V. (2012). Combinatorial optimization for electric vehicles management. *Journal of Energy and Power Engineering*, *6*(5), 738–743.

Uhrig, M., Weiß, L., Suriyah, M. & Leibfried, T. (2015). E-mobility in car parks–guidelines for charging infrastructure expansion planning and operation based on stochastic simulations. In *Evs28 international electric vehicle symposium and exhibition* (pp. 1–12).

Wang, H. & Cheu, R. L. (2013). Operations of a taxi fleet for advance reservations using electric vehicles and charging stations. *Transportation research record*, *2352*(1), 1–10.

Zhao, M. & Lu, Y. (2019). A heuristic approach for a real-world electric vehicle routing problem. *Algorithms*, *12*(2), 45.

Zündorf, T. (2014). Electric vehicle routing with realistic recharging models. *Unpublished Master's thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany*.

A ALNS versus ALNS-SP

Table 8: Comparison of results obtained with ALNS and ALNS-SP on the small-size instances.

Instance	ALNS			ALNS-SP			
	#Veh	TD	Time (sec)	#Veh	TD	Time (sec)	Δ%
C101-5	3	247.15	4.19	3	247.15	4.35	0.00
C103-5	3	165.67	5.17	3	165.67	5.30	0.00
C206-5	3	236.58	7.38	3	236.58	7.47	0.00
C208-5	1	158.48	9.39	1	158.48	9.48	0.00
R104-5	2	136.69	3.52	2	136.45	3.59	-0.18
R105-5	2	156.08	4.05	2	156.08	4.12	0.00
R202-5	1	128.78	6.96	1	128.78	7.02	0.00
R203-5	1	179.06	10.26	1	179.06	10.31	0.00
RC105-5	3	238.05	4.47	3	238.05	4.54	0.00
RC108-5	2	253.93	3.71	2	253.93	3.80	0.00
RC204-5	1	176.39	9.34	1	176.39	9.40	0.00
RC208-5	1	167.98	5.00	1	167.98	5.07	0.00
C101-10	4	393.56	16.83	4	383.56	17.26	-2.61
C104-10	2	273.24	19.11	2	273.93	19.38	0.25
C202-10	2	243.20	34.04	2	243.20	34.25	0.00
C205-10	2	228.28	16.05	2	228.28	16.28	0.00
R102-10	4	262.92	9.51	4	258.03	9.66	-1.90
R103-10	3	202.85	10.45	3	202.85	10.69	0.00
R201-10	3	228.36	42.51	3	217.68	42.76	-4.90
R203-10	1	218.21	30.25	1	218.21	30.33	0.00
RC102-10	4	423.51	10.43	4	423.51	10.64	0.00
RC108-10	3	345.11	12.92	3	345.11	13.12	0.00
RC201-10	3	310.06	37.64	3	310.06	37.85	0.00
RC205-10	2	325.98	32.84	2	325.98	33.03	0.00
C103-15	4	369.32	48.73	4	369.32	49.48	0.00
C106-15	3	275.13	29.59	3	275.13	30.06	0.00
C202-15	3	369.56	75.20	3	369.56	75.66	0.00
C208-15	2	300.55	73.06	2	300.55	73.49	0.00
R102-15	5	423.46	30.15	5	413.46	30.54	-2.42
R105-15	4	344.16	27.61	4	339.88	27.98	-1.26
R202-15	2	361.10	209.59	2	358.22	210.09	-0.80
R209-15	2	293.20	125.78	2	293.20	126.17	0.00
RC103-15	4	394.65	24.02	4	394.65	24.43	0.00
RC108-15	3	395.71	31.48	3	375.88	31.88	-5.27
RC202-15	2	394.39	140.63	2	394.39	141.13	0.00
RC204-15	2	310.58	274.84	2	310.58	275.32	0.00
Average			39.91			40.16	-0.53

B Results for large-size instances under different recharge strategies

Table 9: EVRPTW results for different recharge strategies

Instance	FR		PR (q free)			PR (q = 0.3)			PR (q = 0.4)			PR (q = 0.5)		
	#Veh	TD	#Veh	TD	Δ% %	#Veh	TD	Δ% %	#Veh	TD	Δ% %	#Veh	TD	Δ% %
c101	13	1090.17	13	1086.26	-0.36	13	1080.98	-0.85	13	1082.48	-0.71	13	1090.17	0
c102	12	1058.27	12	1035.08	-2.24	12	1037.72	-1.98	12	1035.89	-2.16	12	1045.20	-1.25
c103	12	1028.10	11	1010.10	-	11	1012.91	-1.50	11	1035.69	-	11	1038.99	-
c104	11	1024.66	11	958.25	-6.93	11	961.58	-6.56	11	970.41	-5.59	11	965.47	-6.13
c105	12	1071.59	12	1034.25	-3.61	12	1048.83	-2.17	12	1048.31	-2.22	12	1062.13	-0.89
c106	12	1081.95	12	1048.71	-3.17	12	1063.35	-1.75	12	1060.84	-1.99	12	1071.03	-1.02
c107	13	1035.67	12	1030.68	-	12	1020.26	-1.51	12	1018.76	-1.66	12	1018.86	-1.65
c108	13	1044.86	12	1043.59	-	11	999.99	-	12	1033.29	-1.12	12	1030.13	-1.43
c109	12	1164.07	12	1035.28	-12.44	12	1079.24	-7.86	12	1039.53	-11.98	12	1041.39	-11.78
c201	7	762.19	7	744.69	-2.35	7	760.44	-0.23	7	760.44	-0.23	7	758.17	-0.53
c202	6	751.29	6	733.97	-2.36	6	746.96	-0.58	6	746.96	-0.58	6	747.26	-0.54
c203	5	728.52	5	711.72	-2.36	5	724.32	-0.58	5	724.32	-0.58	5	724.17	-0.60
c204	7	775.25	7	767.50	-1.01	7	774.48	-0.10	7	775.25	0.00	7	775.25	0.00
c205	7	764.48	7	752.66	-1.57	7	761.28	-0.42	7	753.77	-1.42	7	760.98	-0.46
c206	7	765.78	7	755.58	-1.35	7	769.32	0.46	7	757.52	-1.09	7	765.78	0.00
c207	6	769.05	6	758.81	-1.35	6	772.61	0.46	6	769.05	0.00	6	769.05	0.00
c208	7	760.57	7	750.44	-1.35	7	756.64	-0.52	7	756.64	-0.52	7	760.57	0.00
r101	19	1639.38	19	1623.47	-0.98	19	1598.46	-2.56	19	1617.70	-1.34	19	1639.87	0.03
r102	18	1483.72	18	1423.64	-4.22	18	1438.83	-3.12	18	1439.24	-3.09	18	1437.15	-3.24
r103	15	1292.87	15	1247.34	-3.65	14	1245.67	-	15	1248.42	-3.56	15	1265.16	-2.19
r104	13	1070.00	12	1079.31	-	12	1139.57	-	12	1067.44	-0.24	12	1064.04	-0.56
r105	16	1397.94	16	1386.98	-0.79	16	1396.54	-0.10	16	1395.01	-0.21	16	1411.77	0.98
r106	16	1301.48	15	1303.88	-	15	1313.17	0.89	15	1323.58	1.67	15	1311.45	0.76
r107	13	1195.59	13	1165.41	-2.59	13	1180.02	-1.32	13	1177.46	-1.54	13	1180.37	-1.29
r108	12	1098.64	12	1078.26	-1.89	12	1093.17	-0.50	12	1091.54	-0.65	12	1095.35	-0.30
r109	14	1257.05	14	1239.45	-1.42	14	1235.79	-1.72	14	1247.32	-0.78	14	1236.40	-1.67
r110	14	1165.61	13	1163.01	-	13	1113.59	-	13	1163.52	-0.18	13	1160.04	-0.48
r111	14	1173.95	14	1149.35	-2.14	14	1167.64	-0.54	14	1159.11	-1.28	14	1165.67	-0.71
r112	12	1085.62	12	1063.71	-2.06	12	1081.84	-0.35	12	1052.06	-3.19	12	1084.65	-0.09
r201	11	1185.82	11	1186.65	0.07	11	1187.36	0.13	11	1196.23	0.87	11	1183.81	-0.17
r202	9	1081.53	9	1081.53	0.00	9	1084.02	0.23	9	1083.91	0.22	9	1084.78	0.30
r203	7	924.88	7	924.88	0.00	7	925.99	0.12	7	927.01	0.23	7	925.07	0.02
r204	4	753.02	4	753.02	0.00	4	732.59	-	4	735.29	-	4	773.04	2.59
r205	7	1037.24	7	1037.24	0.00	7	1041.61	0.42	7	1047.51	0.98	7	1040.57	0.32
r206	9	1007.08	9	1007.08	0.00	9	1019.72	1.24	9	1021.90	1.45	9	1010.31	0.32
r207	6	851.71	6	850.94	-0.09	6	854.18	0.29	7	811.63	-	6	854.10	0.28
r208	5	757.33	5	757.33	0.00	5	762.67	0.70	5	758.09	0.10	5	758.77	0.19
r209	7	951.24	7	947.92	-0.35	7	952.19	0.10	7	952.19	0.10	7	951.71	0.05
r210	7	927.48	7	926.37	-0.12	7	936.28	0.94	7	935.62	0.87	7	935.43	0.85
r211	6	835.21	5	830.52	-	5	835.21	0.00	5	842.54	0.87	5	841.52	0.75
rc101	17	1780.70	17	1731.02	-2.87	16	1835.79	-	17	1742.37	-2.20	17	1755.25	-1.45
rc102	17	1572.33	16	1575.39	-	16	1877.04	-	16	1865.04	-	16	1868.04	-
rc103	3	1428.64	3	1405.45	-1.65	3	1428.64	0.00	3	1422.24	-0.45	3	1421.53	-0.50
rc104	3	1286.36	3	1255.11	-2.49	3	1286.36	0.00	3	1283.15	-0.25	3	1278.56	-0.61
rc105	6	1536.43	6	1521.07	-1.01	6	1531.84	-0.30	6	1530.31	-0.40	6	1532.30	-0.27
rc106	16	1490.37	15	1493.57	-	15	1519.04	-	15	1534.04	-	15	1498.79	-
rc107	13	1343.57	13	1321.76	-1.65	13	1333.43	-0.76	13	1326.06	-1.32	13	1322.02	-1.63
rc108	5	1287.18	5	1259.72	-2.18	5	1261.70	-2.02	5	1316.94	2.26	5	1287.95	0.06
rc201	10	1320.78	10	1320.78	0.00	10	1335.47	1.10	10	1336.15	1.15	10	1329.29	0.64
rc202	10	1225.13	10	1183.36	-3.53	11	1224.56	-	10	1228.26	-	10	1218.74	-
rc203	8	1001.34	8	1001.34	0.00	8	1021.67	1.99	8	1022.82	2.10	8	1015.45	1.39
rc204	7	904.16	7	904.16	0.00	7	903.80	-0.04	7	911.08	0.76	7	902.81	-0.15
rc205	9	1171.18	9	1155.81	-1.33	9	1169.43	-0.15	9	1147.34	-	9	1153.93	-
rc206	8	1127.81	8	1130.98	0.28	8	1126.12	-0.15	8	1128.72	0.08	8	1129.51	0.15
rc207	7	984.95	7	983.97	-0.10	7	987.13	0.22	7	982.50	-0.25	7	983.48	-0.15
rc208	7	875.04	7	873.29	-0.20	7	882.89	0.89	7	877.76	0.31	7	879.88	0.55
Average					-1.69			-0.64			-0.79			-0.62

C Results for ALNS with Labeling Algorithm

Table 10: Comparison of runtimes for different recharge strategies under ALNS with Labeling Algorithm

Instance	FR			PR (q = 0.4)			PR (q = 0.6)			PR (q = 0.8)			PR (2 customers)		
	#Veh	TD	Time (sec)	#Veh	TD	Time (sec)	#Veh	TD	Time (sec)	#Veh	TD	Time (sec)	#Veh	TD	Time (sec)
C101-5	3	247.15	42.02	3	247.15	30.05	3	247.15	59.04	3	247.15	48.15	4	250.04	4.62
C103-5	2	165.67	70.36	2	165.67	102.09	2	165.67	76.24	2	165.67	61.35	3	165.67	31.12
C206-5	2	221.97	423.37	3	244.30	119.97	3	236.58	528.55	2	221.97	593.60	2	226.89	292.73
C208-5	1	164.34	174.87	2	189.06	703.64	1	164.34	238.01	1	164.34	166.13	1	174.82	22.57
R104-5	2	136.69	6.06	2	136.69	39.27	2	136.69	8.29	2	136.69	10.33	2	136.69	7.10
R105-5	2	156.08	6.34	3	182.92	36.12	3	182.92	14.24	2	156.08	12.85	3	165.40	44.78
R202-5	1	128.88	497.73	2	143.39	912.98	1	128.88	493.93	1	128.88	638.32	2	143.04	193.28
R203-5	1	179.06	850.38	2	208.19	467.37	1	191.61	607.33	1	179.06	690.97	1	197.99	265.26
RC105-5	3	238.05	22.08	3	238.05	94.98	3	238.05	15.16	3	238.05	17.87	3	238.05	4.02
RC108-5	2	253.93	21.35	2	264.92	262.67	2	253.93	24.32	2	253.93	33.84	2	253.93	10.54
RC204-5	2	176.00	381.07	2	185.44	965.09	2	185.16	387.54	2	176.00	478.36	2	175.99	80.47
RC208-5	1	167.98	469.54	1	182.79	863.50	1	167.98	346.53	1	167.98	618.31	1	174.38	130.30
C101-10	4	385.78	112.42	4	409.65	652.65	4	385.78	187.04	4	385.78	151.93	4	397.56	25.33
C104-10	2	273.93	261.24	2	300.50	2018.68	2	276.09	963.02	2	271.20	578.53	2	308.51	213.83
C202-10	2	242.09	2782.09	2	245.46	4949.99	2	243.21	2561.92	2	242.09	3012.46	3	270.06	771.55
C205-10	2	228.28	468.65	2	228.91	967.41	2	228.90	570.67	2	228.28	478.69	2	234.52	197.79
R102-10	3	249.19	70.45	3	255.88	299.99	3	255.88	83.51	3	249.19	67.45	4	262.49	24.12
R103-10	3	191.33	73.40	3	196.26	527.79	3	193.69	108.64	3	193.69	71.40	3	199.62	23.42
R201-10	3	211.50	1783.21	3	227.41	4661.91	3	218.37	1517.21	3	210.81	1983.05	3	239.07	247.23
R203-10	1	222.64	2605.33	1	245.46	3278.13	1	235.15	2278.96	1	222.64	2835.23	2	253.36	4363.10
RC102-10	4	423.44	12.26	4	433.29	60.12	4	423.51	20.72	4	423.44	18.36	5	434.52	8.52
RC108-10	3	347.90	23.43	3	368.11	190.39	3	352.09	45.36	3	347.90	28.52	3	347.89	27.93
RC201-10	3	305.41	2758.73	3	310.98	4429.76	3	309.99	2146.96	3	305.41	2652.90	4	341.65	669.11
RC205-10	2	330.00	923.46	2	354.32	983.21	2	337.89	753.17	2	333.00	947.78	3	368.72	299.75
C103-15	4	361.86	573.65	4	368.11	974.63	4	361.86	897.42	4	363.86	673.78	4	369.32	411.97
C106-15	3	271.21	138.51	4	313.95	431.53	3	271.21	199.62	3	275.19	338.54	4	313.17	90.99
C202-15	3	367.60	2367.61	4	398.22	3369.21	3	376.79	2244.05	3	369.30	2567.62	4	418.91	2519.47
C208-15	3	298.85	2487.46	3	325.75	3432.21	2	308.45	1596.93	3	299.87	2697.23	3	356.03	813.31
R102-15	5	394.64	177.01	6	438.88	478.37	6	416.12	229.18	6	409.96	285.04	6	431.08	111.04
R105-15	4	336.15	172.67	4	340.17	479.65	4	336.15	223.08	4	336.15	271.45	4	352.39	141.35
R202-15	3	356.84	9223.20	3	381.43	10253.31	3	376.59	9777.56	3	359.08	9203.27	3	411.57	11537.38
R209-15	3	296.41	8700.82	3	312.79	9451.67	3	309.10	9806.53	3	296.42	8975.20	4	351.86	5521.99
RC103-15	4	388.64	38.55	4	399.64	97.32	4	388.64	76.86	4	388.64	78.54	5	395.22	54.26
RC108-15	3	370.25	94.44	3	382.27	195.49	3	387.93	224.87	3	370.25	94.44	5	445.73	60.40
RC202-15	3	397.14	3580.08	3	398.41	4381.19	2	394.39	4119.83	2	394.39	3780.28	4	435.96	2474.04
RC204-15	2	309.72	5995.33	2	312.12	6135.37	2	310.58	7004.20	2	309.72	5921.43	2	365.14	7035.26
Average			1344.14			1869.38			1401.01			1418.98			1075.83

D Programming code

The ALNS-SP matheuristic with and without labelling algorithm was implemented in Java, involving the following Java classes:

D.1 Fundamental classes:

1. `Customer`: This class represents a customer vertex in a given route; hence it is an object of the Route class. It implements the Vertex interface and describes various properties such as location, demand, time windows, service time, and levels of cargo and battery of the EV when arriving and leaving this customer vertex. `Depot` and `Station` classes operate in the same manner.

2. `CustomerZoneAssignment`: This class assigns objects from the Customer class to different zones based on their coordinates. It does so by viewing the entire area covered by all customers, diving it into a grid and then assigning each customer to a specific zone in the grid. The `Zone` class represents a zone containing a collection of customers.

3. `Instance`: This class contains all the relevant data points for the VRP problem, by parsing the input file and initialising customers, stations, depots and their copies. It also tracks the assignment of customers to different zones based on their location.

4. `Route`: This class handles the routing of a vehicle, managing the insertion and removal of customers and stations and correctly updating the battery levels, cargo load and time windows associated with these operations. It also keeps a track of state variables such at the current battery level of the vehicle, the total distance travelled on the route / total cost accumulated so far. Lastly, Route contains an additional constructor for creating route copies, such that routes from ALNS can be stored and post-processed with SP in Gurobi.

D.2 Classes for constructing a solution:

1. `InitialSolution`: This class constructs the initial solution for ALNS by firstly initialising a `Route` object with the customer closest to the depot and the adding customers based on cheapest feasible insertions such that time windows are respected. Moreover, it also adds stations based on the `GreedyStationInsertion` class in case a customer insertion causes negative battery levels. If the customer cannot be added in the route, they are placed in a new route.

2. `ALNS`: This class implements a metaheuristic algorithm for solving EVRPTW-PR, by dynamically selecting destroy-repair operator for customers and stations based on adaptive weights, and accepting solutions based on the SA criteria.

3. `GurobiSP`: Provides the MIP model set partitioning, to process the solutions from ALNS and search for a lower objective value through combining routes from across different solutions. It outputs the objective value, runtime and solution routes for a given instance.

4. `Gurobi`: Provides the MIP model for partial and full recharging cases, to solve the smaller instances to optimality (or until the runtime limit is reached). It outputs the objective value, runtime and solution routes for a given instance.

D.3 Destroy-repair operator classes for ALNS:

1. **Customer removal operator classes**: These set of classes handle the removal of customers, check for and remove any associated charging stations, as well as empty routes from the solution. `RandomCustomerRemoval` randomly selects and removes customers from a given solution; the number of customers to be removed is simulated randomly from a uniform distribution. `WorstDistanceCustomerRemoval` removes customers based on their distance costs, and `WorstTimeRemoval` removes customers based on how close they are served to their starting time. `ShawRemoval` removes customer based on how similar they are in terms of time windows, demand levels and their proximity. `TimeBasedRemoval`, `DemandBasedRemoval` and `ProximityBasedRemoval` replicate `ShawRemoval` with varying values for the Shaw parameters. `ZoneRemoval` removes customers if they are present within a randomly selected zone. `GreedyRouteRemoval` implements a heuristic for destroying an existing solution by removing routes in a greedy manner, such that it removes the routes that serve the fewest customers; aiming to incorporate the customers from the removal list into the other existing routes. `RandomRouteRemoval` applies the same logic, but removes routes in a random manner, where the number of routes to be removed is drawn from a bounded uniform distribution.

2. **Customer insertion operator classes**: These classes implement various insertion heuristics for customers, such that customers in the removal list (resulting from a previous destroy operation) can be re-inserted back into the routes in a manner which minimises some criteria. For `GreedyCustomerInsertion` customers were inserted into the existing solution in greedy manner, such that the additional distance incurred by the insertions are minimised. `Regret2CustomerInsertion` and `Regret3CustomerInsertion` utilise the Regret-2 and Regret-3 criteria, while `TimeBasedCustomerInsertion` prioritises minimising the increase in route duration when inserting customers. Lastly, `ZoneInsertionCustomer` reinserts customers only into a subset of routes, determined by selecting a zone randomly and picking routes which cover customers present in that zone.

3. **Station removal operator classes**: These classes implement station removal operators which destroy the previous solution by removing charging stations based on certain criteria. The number of stations to be removed is decided based on a random draw from a uniform distribution. `RandomStationRemoval` randomly removes a subset of stations. `WorstDistanceStationRemoval` removes stations based on distance costs, and `WorstChargeStationRemoval` removes stations based on the battery level when arriving at a station, such that highest battery arrival stations are removed first. For `FullChargeStationRemoval`, the class removes stations based on the amount of battery that is recharged at the station, prioritising the removal of highest recharge quantity stations.

4. **Station insertion operator classes**: These class are required to insert the best feasible charging station into a route whenever there is a vertex with a negative battery level. `GreedyStationInsertion` attempts to insert the best station preferably at arc prior to that vertex, else it also considers the preceding arcs if station insertion is infeasible. It does so until there is no negative battery customer left. Else, the insertion is declared infeasible. `GreedyStationInsertionComparison` functions similarly, but it prefers to add the best station to the arc or two arcs prior to the negative battery vertex; whichever is cheaper. Else, it implements greedy station insertion on preceding arcs. `BestStationInsertion` finds the cheapest station that can be inserted feasibly before a negative battery customer by considering all previous arcs until a depot / station is encountered. Note that these classes consider one `Route` object at a time. Therefore, we adapt these classes for ALNS by creating `GreedyStationInsertionALNS`, `GreedyStationInsertionComparisonALNS` and `BestStationInsertionALNS`, where the corresponding station insertion operators are implemented in an iterative manner for all `Route` objects within a solution.

D.4 Labeling algorithm:

1. `LabellingAlgorithm`: This class implements a *labeling algorithm* for inserting charging stations into a route to ensure that the battery constraints are met. It does so by iteratively building a series of labels and applying rules to check if they are valid or have been dominated. It then picks the least cost label at the end depot and builds the final route by following the arcs back to the start depot, including adding stations on the route.

E List of abbreviations

Table 11: Abbreviations and Full Descriptions

Abbreviation	Full Description
GHG	Greenhouse gas
ICEV	Internal combustion engine vehicles
EV	Electric vehicle
EVRP	Electric Vehicle Routing Problem
GVRP	Green Vehicle Routing Problem
EVRPTW	Electric Vehicle Routing Problem with Time Windows
FR	Full recharging
PR	Partial recharging
NL	Non-linear
EVRPTW-PR	Electric Vehicle Routing Problem with Time Windows and Partial Recharging
EVRPTW-NLPR-MC	Electric Vehicle Routing Problem with Time Windows, Non-linear Partial Recharging and Multiple Chargers
MILP	Mixed-integer linear program
ALNS	Adaptive Large Neighbourhood Search
SP	Set partitioning
ALNS-SP	Adaptive Large Neighbourhood Search with Set Partitioning
RVRP	Recharging Vehicle Routing Problem
SoC	State of Charge
CS	Charging station
TS	Tabu Search
VNS	Variable Neighborhood Search
SA	Simulated annealing
EVRP-NL	Electric Vehicle Routing Problem with Non-linear recharging
EVRPTW-NL	Electric Vehicle Routing Problem with Time Windows and Non-linear recharging
CR	Customer Removal
CI	Customer Insertion
SR	Station Removal
SI	Station Insertion
RRR	Random Route Removal
GRR	Greedy Route Removal
C	Clustered
R	Randomly distributed
RC	Clustered and randomly distributed