Erasmus University Rotterdam

Erasmus School of Economics

Bachelor Thesis International Bachelor Econometrics and Operations Research

A study on Nonlinear charging functions in a Green Mixed Fleet Vehicle Routing Problem with Partial Recharge and Time Windows

Hoang Thi Khue Nguyen (574244)

Supervisor:	Y.N. Hoogendoorn
Second assessor:	R.M. Badenbroek
Date final version:	1st July 2023

Abstract

In the context of green vehicle routing problem, charging functions refer to modeling the relationship between the amount of energy recharged and the state of charge of the electric vehicle's battery, which is typically nonlinear. Nevertheless, most existing studies adopt linear charging functions, raising the question of how well this assumption captures real-world problem. In this paper, we study the impact of nonlinear charging functions within a green vehicle routing problem that involves a mixed fleet of conventional and electrical vehicles, a partial recharge policy, and time windows. We propose a matheuristic combining an iterated local search and a set partitioning formulation. A computational study is conducted based on 288 test instances with different size. We find that nonlinear charging functions provide a better approximation for instances with high level of complexity, as there are six instances with 50 to 100 customers whose solutions found under the linear recharge assumption are infeasible in the nonlinear scenario. Furthermore, we conclude that solving a set partitioning formulation as an improvement step can reduce the solution cost found by the iterated local search procedure by up to 48.49%. Lastly, our matheuristic is able to achieve solutions within a reasonable amount of time.

1 Introduction

The Vehicle Routing Problem (VRP) is one of the most important and well-documented $\mathcal{NP}$-hard combinatorial optimisation problems due to its wide range of application in logistics and transportation. Given a set of customers and a fleet of vehicles, the VRP looks for the optimal design of routes to visit all customers such that a certain objective function is optimised. Many variants of the VRP are intensively studied to tackle real-life situations, such as complex system structures, constraints on customers, or the operation of vehicles and drivers (Vidal, Crainic, Gendreau & Prins, 2013). In recent years, climate change has raised the need for more green policies in logistics and transportation, thereby drawing attention to the environmental externalities of vehicle routing. Consequently, the Green Vehicle Routing Problem (G-VRP), a new variant that integrates environmental aspects into the VRP, was formally introduced by Erdoğan and Miller-Hooks (2012) and has become more and more popular over the years.

In general, the G-VRP aims to decrease the environmental externalities. One can account for the CO_2 emissions generated from conventional vehicles (CVs) by including the aforementioned in the problem objective. Another approach is to utilize the CVs efficiently in terms of speed and load capacity, as these factors influence the amount of emissions the CVs produce. One of the most popular directions is to employ alternative fuel vehicles (AFVs) since they generate lower or zero greenhouse gases compared to conventional counterparts. Most existing AFV-related G-VRP literature focuses on electrical vehicles (EVs) (Sabet & Farooq, 2022), which are tied with the issues such as battery autonomy, battery degradation, or charging time. Furthermore, the use of EVs involves charging stations (CSs), in which also require certain assumptions such as the distribution of CSs or the available recharging technology. The aforementioned issues extend further the variety of EV-related G-VRP variants, allowing for more practical problems.

Nonetheless, the realistic characteristics of the EV-related G-VRP were simplified in the majority of existing literature. Most studies used a linear function to model the relationship between the amount of energy recharged and the state of charge of the vehicle's battery (linear recharging function), whereas the aforementioned relationship is nonlinear in practice (Sabet & Farooq, 2022). Montoya, Guéret, Mendoza and Villegas (2017) found that the linear recharge assumption might result in infeasible routes under nonlinear scenario, and the linear recharge solutions exhibit an average cost rise of 2.7%. They assumed a full recharge policy (i.e., an EV is recharged to its maximum battery upon its departure from a CS), which is a common assumption among G-VRP literature. Felipe, Ortuño, Righini and Tirado (2014) was the first to investigate the effect of partial recharging, and concluded that this policy could significantly improve the overall costs and energy consumption. In their computational study, allowing for partial recharging saves on average 1.15%, 1.51%, and 1.7% costs for instances with 100, 200, and 400 customers.

This paper studies a green mixed fleet VRP with nonlinear partial recharging and time windows (GMFVRP-NLPRTW). In particular, we focus on the influence of nonlinear recharging functions, which are commonly simplified as linear recharging functions in existing G-VRP literature. We study a mixed fleet of conventional internal combustion commercial vehicles (ICCVs) and electric commercial vehicles (ECVs). In addition, we employ a partial recharge policy and account for battery degradation. Furthermore, we consider the direct restriction on the amount of CO_2 emissions as in Macrina, Pugliese, Guerriero and Laporte (2019) - a feature that distinguishes their work from most studies whose focus was on cost and energy consumption (Macrina et al., 2019). We propose a matheuristic ILS-SP which combines an iterated local search (ILS) and a set partitioning (SP) formulation to solve the proposed problem. We conduct a computational study based on 288 test instances of different size to examine the impact of nonlinear recharge and the performance of the ILS-SP. Our results show that, when the number of customers grows, the solution cost found under the nonlinear recharge assumption deviates more frequently with greater variation from those found in the linear scenario. Furthermore, there are six test instances with 50 to 100 customers whose linear recharge solutions become infeasible in the nonlinear scenario. We therefore conclude that the nonlinear recharge functions are more appropriate for modeling instances with high level of complexity. In addition, from the performance of the ILS-SP, we find that it is cost-effective to solve an SP formulation as an improvement step for the ILS procedure, as this can achieve a cost reduction by up to 48.49%. The ILS-SP is able to achieve solutions within a reasonable amount of time for all instances.

The remaining text is organised as follows: In Section 2, we provide the literature that is relevant to our study. In Section 3, we describe the GMFVRP-NLPRTW studied in this paper. In Section 4, we discuss the proposed matheuristic. In addition, we provide a computational study in Section 5 and conclude our findings in Section 6.

2 Literature review

The growing interest that the G-VRP has received is evidenced by the amount of articles dedicated to it and its variants. Recent reviews of these articles include Asghari, Al-e et al. (2021), Moghdani, Salimifard, Demir and Benyettou (2021), and Sabet and Farooq (2022), in which the last one classified G-VRP studies based on the type of vehicles available in the fleet: only CVs, only AFVs, or a mix of CVs and AFVs.

To account for environmental impact of a fleet of CVs, a common approach among the G-VRP studies is to minimise the total amount of CO_2 emissions generated by the CVs. Bektaş and Laporte (2011) was the first the introduce the Pollution Routing Problem (PRP), in which they included in their objective function the greenhouse emissions and fuel consumption of the CVs. They provided a nonlinear integer programming formulation, together with a linearisation procedure. Later studies considered also the vehicle speed in their problem due to its significant correlation with the emissions generated by CVs. Jabali, Van Woensel and De Kok (2012) included an upper bound on the maximum vehicle speed as part of the optimisation and solved an emission-based time-dependent VRP with tabu search. They concluded that restricting the vehicle speed was beneficial for minimising the overall costs. Tajik, Tavakkoli-Moghaddam, Vahdani and Mousavi (2014) studied for the first time a time window pickup-delivery PRP in which uncertain data was addressed. Their objective function included distance travelled, greenhouse emissions, and number of CVs. A distinguishable feature of their study is the use of a robust optimisation in which the vehicle speed was considered as an uncertain parameter. Kramer, Subramanian, Vidal and Lucídio dos Anjos (2015) studied the PRP introduced by Bektaş and Laporte (2011) with a heuristic including an iterated local search (ILS), speed optimisation, and a set partitioning (SP) formulation. They considered the vehicle speeds as decision variables, and these were optimised for each route.

Thanks to the technological advances in electrical mobility in recent years, EVs has become more and more appealing to researchers as an alternative to conventional counterparts

(Kucukoglu, Dewil & Cattrysse, 2021; Ye, He & Chen, 2022). The study on Electric VRP (E-VRP), a variant of the G-VRP that studies a fleet of only EVs, is therefore substantially extended over the years. On the contrary, there is a lack of attention on mixed fleet of CVs and EVs compared to the other G-VRP types (Macrina et al., 2019; Sabet & Farooq, 2022). Gonçalves, Cardoso, Relvas and Barbosa-Póvoa (2011) is one of the earliest to study a mixed fleet G-VRP with pickup and delivery. To account for the charging process of the EVs, they made a simple assumption as follows: Whenever it is necessary to recharge the EVs, an additional duration for recharging is added to the travelling time. Furthermore, the location of CSs was not incorporated explicitly in their study.

Erdoğan and Miller-Hooks (2012) studied a G-VRP employing a fleet of AFVs and developed two constructive heuristics. They assumed a constant fuel consumption rate and a full refueling policy (i.e., all AFVs are refueled to their maximum capacity upon an arrival at any refueling stations). However, the aforementioned assumptions raised the question whether they were good approximation of the real-world problem. Attention has been drawn to the practical aspect of the E-VRP, raising the need for more complex problem variants that are associated to realistic assumptions on energy consumption model, charging policies, and CSs (Montoya et al., 2017).

Goeke and Schneider (2015) was the first G-VRP paper that employed an energy consumption model depending on speed, gradients, and load, for both EVs and CVs. They incorporated time windows and a mixed fleet of EVs and CVs. They considered a full recharge policy with linear recharge, which are common assumptions among E-VRP literature. The former was proven to be less beneficial compared to a partial recharge policy in Felipe et al. (2014), who studied another version of the problem in Erdoğan and Miller-Hooks (2012). They assumed that EVs could be partially recharged with a linear recharging function, and multiple charging technologies were considered at the charging stations (CSs). They concluded that the impact of partial recharge policy was significant in minimising costs and energy, and it helped guaranteeing feasibility for some instances. Macrina et al. (2019) also studied a mixed fleet of EVs and CVs with time windows, partial recharge policy, and linear recharge functions. They employed an emissions model which took into account the distance travelled and the amount of load available. A distinguishable feature in their study was that they directly restricted the amount of CO_2 emissions to a certain level instead of including it in the objective. They proposed an ILS heuristic to solve their proposed problem.

According to Montoya et al. (2017), the charging functions are typically nonlinear in reality. This indicates that the results of most EV-related literature with a linear charging process might become impractical (Sabet & Farooq, 2022). Montoya et al. (2017) was one of the earliest studies that investigated nonlinear (NL) recharging functions. They introduced the E-VRP-NL with full recharge policy and multiple charging technologies. They modeled the recharging procedure by fitting piecewise linear functions on actual charging data, and proposed a metaheuristic combining an ILS with a heuristic concentration that solves an SP formulation. Their computational experiments showed that the solutions when assuming linear charging functions might be too costly or even infeasible. Froger, Mendoza, Jabali and Laporte (2019) extended the aforementioned paper by developing two new formulations to the E-VRP-NL and further enhanced Montoya et al. (2017)'s metaheuristic with three new algorithms.

3 Problem description

This paper studies a green mixed fleet vehicle routing problem with nonlinear partial recharging and time windows (GMFVRP-NLPRTW). We will use the notation from Macrina et al. (2019). We denote by N the set of customers, each of which has a demand q_i (kg), a time window $[t_i^e, t_i^l]$ (hours), and a service duration s_i (hours). The set of charging stations (CSs) is denoted by R. The problem includes a single depot at which all vehicles must start and end their route, and the depot is also a CS in the set R. We denote two artificial depots s and t indicating the start and end of all routes, respectively. The set of all vertices is $V = R \cup N \cup \{s, t\}$, and the set of arcs is $A = \{(i, j) : i, j \in V, i \neq j\}$. For any arc $(i, j) \in A$, the distance is d_{ij} (km), and the travelling time is t_{ij} (hours). All vehicles travel with a constant speed v (km/h), and they should return to the depot within T hours.

The fleet is composed of ICCVs (C) and ECVs (E). For each vehicle type $k \in \{C, E\}$, the number of available vehicles is n^k, and each vehicle has a maximum load capacity of Q^k (kg). The total CO_2 emissions generated by the ICCVs must not exceed an upper bound UB (kg). Additionally, each ECV has a maximum battery capacity of B^E (kWh). The energy consumption on any arc $(i, j) \in A$ is assumed to be proportional to the travelled distance at the rate π (kWh/km). We denote the travelling cost by c_{ij}^k (€/km). If an ECV is assigned to a route, it will be activated at the depot by being charged to full battery. The activation cost w^a (€) is therefore the cost of a full recharge with the technology available at the depot.

Furthermore, partial recharging is allowed at all CSs, each of which is characterised by a charging cost w_j^r (€) and a charging function, for $j \in R$. Following the notation in Montoya et al. (2017), we denote the charging function at a CS $j \in R$ by $g_j(z_j, \Delta_j)$, where the first input z_j is the energy upon arrival at j of an ECV (kWh), the second input Δ_j is the charging time at j (hours), and the output is the energy upon departure from j of the ECV (kWh).

To account for battery degradation, we introduce an additional set of constraints, namely the state of charge (SoC) constraints. In particular, the energy upon arrival at any vertices of an ECV must be at least $10\%B^E$.

As in Macrina et al. (2019), the objective is to minimise the overall costs function (1),

$$f(\eta) = \sum_{i\in R}\sum_{j\in V} w_i^r g_{ij} + \sum_{j\in V} w^a x_{sj}^E + \sum_{i\in V}\sum_{j\in V} c_{ij}^C d_{ij} x_{ij}^C + \sum_{i\in V}\sum_{j\in V} c_{ij}^E d_{ij} x_{ij}^E, \tag{1}$$

where η denotes a solution to the GMFVRP-NLPRTW, x_{ij}^k takes value of 1 if the vehicle of type k travels on arc (i,j) and 0 otherwise, and g_{ij} is the amount of energy recharged at CS i before visiting vertex j. The total costs includes the cost of recharging at the CSs, the cost of using the ECVs, and the costs of travelling for ICCVs and ECVs, respectively.

Lastly, the amount of CO_2 emissions is accounted by an emission function $\varepsilon(\cdot)$, whose input is the load carried by an ICCV to a vertex, as in Macrina et al. (2019). Using the estimation of emission factors for a 10 tonne load capacity vehicle from Úbeda, Faulin, Serrano and Arcelus (2014), we assume that the emission factor $\varepsilon_{ij} = \varepsilon(u_i^C)$ on an arc (i,j) takes value 0.77, 0.83, 0.90, 0.95, or 1.01, if the load carried on the arc belongs to the domain $[0, 0.25Q^C)$, $[0.25Q^C, 0.50Q^C)$, $[0.50Q^C, 0.75Q^C)$, $[0.75Q^C, Q^C)$, or equal to Q^C, respectively. Following Macrina et al. (2019), we directly restrict the amount of CO_2 emissions with an upper bound UB (kg).

3.1 Estimating the charging functions

As mentioned in the previous section, the charging function $g_j(z_j, \Delta_j)$ requires two different inputs. Following Montoya et al. (2017) who transformed the charging functions in the same manner as Zündorf (2014), we define a function $\hat{g}_j(x)$ that yields the energy upon departure from the CS j (kWh) of an ECV whose energy is 0 kWh upon arrival at j and it is charged for x hours at j. The inverse $\hat{g}_j^{-1}(x)$ is therefore the charging time at j for a target energy level upon departure of x kWh. We estimate $g_j(z_j, \Delta_j)$ with $\hat{g}_j(\hat{g}_j^{-1}(z_j) + \Delta_j)$.

To estimate $\hat{g}_j(x)$, Zündorf (2014) proposed to use piecewise linear functions and claimed that such estimation was accurate. Montoya et al. (2017) reinforced the aforementioned argument by fitting piecewise linear functions to real world charging data contributed by Uhrig, Weiß, Suriyah and Leibfried (2015). The average absolute error of their estimation is 0.90%, 1.24%, and 1.90% for a CS with charging power of 11, 22, and 44 kW, respectively. Following the aforementioned research, we use piecewise linear functions to approximate the function $\hat{g}_j(x)$.

3.2 A set partitioning formulation

We use a set partitioning (SP) formulation to model our problem. Let $\Omega = \Omega^C \cup \Omega^E$ be the set of all feasible routes, where Ω^C and Ω^E contains all ICCV- and ECV-routes, respectively.

A route $r^k, k \in \{C, E\}$, is considered to be feasible if it starts and ends at the depot, it obeys the time windows of its vertices, and the total load of all customers served by r^k does not exceed the maximum load capacity. Additionally, if $k = E$, then the corresponding ECV must has sufficient energy level to reach each vertex and satisfies the SoC constraints to account for battery degradation, i.e., the battery level must be at least $10\%B^E$ upon arrival at the successor vertex. We introduce the binary decision variable x_r, which takes value 1 if route $r \in \Omega^k$ is chosen for a vehicle of type $k \in \{C, E\}$, and 0 otherwise. The parameters include $a_{i,r}$ which takes value 1 if the route $r \in \Omega^k$ covers customer $i \in N$ and 0 otherwise, c_r the overall cost associated to the route $r \in \Omega^k$, as described in the previous section; and ε_r is the total CO_2 emission generated by the conventional route $r \in \Omega^C$. The SP formulation is as follows:

$$\min \quad \sum_{k\in\{C,E\}} \sum_{r\in\Omega^k} c_r x_r \tag{2a}$$

$$\text{s.t.} \quad \sum_{k\in\{C,E\}} \sum_{r\in\Omega^k} a_{i,r} \cdot x_r = 1, \qquad \forall i \in N, \tag{2b}$$

$$\sum_{r\in\Omega^C} x_r \leq n^C, \tag{2c}$$

$$\sum_{r\in\Omega^E} x_r \leq n^E, \tag{2d}$$

$$\sum_{r\in\Omega^C} \varepsilon_r x_r \leq UB, \tag{2e}$$

$$x_r \in \{0,1\}, \quad \forall r \in \Omega^k, k \in \{C, E\}. \tag{2f}$$

The objective (2a) minimises the overall cost of the chosen routes. Constrains (2b) ensures that each customer is visited exactly once. Constraints (2c)-(2d) limit the number of used ICCVs and ECVs, respectively, to the available vehicles of the fleet. Constraint (2e) restricts the total CO_2 emissions generated by the ICCVs to an upper bound UB. Lastly, constraints (2f) define the values of the decision variables.

4 Methodology

We propose the ILS-SP method, a matheuristic that combines an iterated local search (ILS) and a set partitioning (SP) formulation. We denote a solution, which is a set of conventional and electrical routes, by η. Algorithm 1 presents the framework of our method which consists of the following main procedures. In Section 4.1, we describe our `initialisation` phase. In Section 4.2 and Section 4.3, we introduce the two main components of the ILS, namely the `localSearch` and the `perturbation` procedures. To optimise the charging decisions (including the decisions on which CSs to visit and how much energy the ECV should recharge), we employ a `labeling` algorithm whose mechanism is elucidated in Section 4.4. Lastly, in Section 4.5, we describe the procedure for solving the SP formulation as our improvement procedure.

Algorithm 1 The ILS-SP framework

```
procedure ILS-SP
    $\eta \leftarrow$ initialisation
    $\eta^* \leftarrow$ localSearch($\eta$)
    Add $\eta^*$ to $\Omega$
    while Termination condition is not satisfied do
        $\eta \leftarrow$ perturbation($\eta^*$)
        $\eta^* \leftarrow$ localSearch($\eta$)
        Add $\eta^*$ to $\Omega$
    end while
    $\eta^* \leftarrow$ Solving an SP formulation with $\Omega$
    return $\eta^*$
end procedure
```

4.1 Initialisation

Following the constructive heuristic proposed by Macrina et al. (2019), our `initialisation` procedure consists of three steps: First, we perform a `clustering` algorithm to obtain two disjoint sets of customers, one is served by the ICCVs (set C) and the other by the ECVs (set E). Second, we apply an insertion strategy based on the sequential insertion heuristic proposed by Solomon (1987) to construct the routes travelled by the ICCVs (ICCV-routes). Third, we apply the aforementioned insertion strategy with some modifications to construct the routes travelled by the ECVs (ECV-routes). Algorithm 2 presents the framework of our `initialisation` procedure.

Algorithm 2 The initialisation procedure

```
procedure INITIALISATION
    $C, E \leftarrow$ clustering($N$)
    $\eta_C \leftarrow$ insertionHeuristicForICCV($C$)
    if There are unrouted customers in $C$ then
        Add the unrouted customers in $C$ to $E$
    end if
    $\eta_E \leftarrow$ insertionHeuristicForECV($E$)
    return $\eta' = \eta_C \cup \eta_E$
end procedure
```

Different from Macrina et al. (2019), we perform two preliminary steps before running the ILS-SP. First, we look for the customers that cannot be served by an ECV. In particular, we employ a `labeling` algorithm, which is described in Section 4.4, to find a feasible ECV-route to visit a single customer i, $\forall i \in N$. If the `labeling` algorithm is unable to find any feasible ECV-route for a customer, then the customer is added to the set of vertices that must be served by the ICCVs (C'). However, it might be impossible to route the customers in C' if the limit on the amount of CO_2 emissions is too low. Therefore, the second preliminary step is to guarantee that each instance has at least a feasible solution corresponding to the upper bound UB on the total emissions. We compute the amount of CO_2 emissions generated by initialising an ICCV-route for each vertex in C', denoted by $\varepsilon_{C'}$. If UB is lower than this value, then we set $UB = \varepsilon_{C'}$. With this step, we are guaranteed that all customers in C' are routed with the ICCVs.

4.1.1 Clustering

Given a set of customers N, we perform a `clustering` algorithm to find two disjoint sets of customer, namely set C consists of customers served by the ICCVs, and set E consists of customers served by the ECVs, where $C \cup E = N$. We use the same notation as in Macrina et al. (2019). For $k \in \{C, E\}$, we let b_k be the barycentre of the set k, and d_i^k be the Euclidean distance from customer $i \in N$ to the barycentre of the set k. We define $d_{min}^k = \min_{i \in N} d_i^k$ and $d_{max}^k = \max_{i \in N} d_i^k$ the distances from the barycentre of k to the nearest and furthest customers, respectively. In addition, the smallest and largest customer demands are denoted by q_{min} and q_{max}, respectively. For each customer $i \in N$, we compute the scores p_i^E and p_i^C as follows:

$$p_i^E = 11 - \left(1 + 9 \times \frac{d_i^E - d_{min}^E}{d_{max}^E - d_{min}^E}\right), \tag{3}$$

$$p_i^C = \lambda \times pD_i^C + (1 - \lambda) \times pQ_i, \tag{4}$$

$$pD_i^C = 11 - \left(1 + 9 \times \frac{d_i^C - d_{min}^C}{d_{max}^C - d_{min}^C}\right), \tag{5}$$

$$pQ_i^C = 11 - \left(1 + 9 \times \frac{q_i - q_{min}}{q_{max} - q_{min}}\right), \tag{6}$$

where the parameter $\lambda \in [0, 1]$ puts weights on the distance from customer i to the barycentre of C, as well as the demand of customer i, since the latter directly influences the total emissions.

We initialise the two sets as $C = E = \{s\}$, then iteratively add customers to the set until all customers are assigned. For each iteration, we compute the new barycentres of both sets C and E, the new distances for each customers $j \in N$ (the minimum/maximum distances to the barycentres are therefore also updated), and the scores p_i^C and p_i^E for each unassigned customer $i \in N \setminus (C \cup E)$. Next, we select the customers whose scores are the largest, namely $i_k^* = \text{argmax}_{i \in N \setminus (C \cup E)} \{p_i^k\}$ for $k \in \{C, E\}$. If $i_C^* \neq i_E^*$, then we add customer i_k^* to the set k for $k \in \{C, E\}$. If $i_C^* = i_E^* = i^*$, we add customer i^* to the set k^* which corresponds to the higher score, i.e. $k^* = \text{argmax}_{k \in \{C, E\}} \{p_{i^*}^k\}$. The algorithm terminates when all customers are assigned to either the set C or E, and we remove the depot s from both sets.

Different from Macrina et al. (2019), we will remove the customers in E that must be served by the ICCVs (i.e., the vertices in $E \cap C'$) and add them to C. Next, C is sorted in ascending order of the customers' due time, with those who must be served by the ICCVs being prioritised.

4.1.2 Insertion heuristic for ICCV-routes

As in Macrina et al. (2019), our insertion heuristic is based on the sequential insertion heuristic proposed by Solomon (1987). We provide in the Appendix A the pseudocode for constructing the set of ICCV-routes η_C. In each iteration, the current route r is initialised as $\{s, i', t\}$ where i' is customer who has the earliest due time among the unrouted customers. Next, we iteratively add the best customer u^* into the corresponding best inserting position $p^*_{u^*}$ of the current route. Let $(s, i_1, i_2, ..., i_m, t)$ be the current route, the best unrouted customer u^* and the best inserting position p^*_u are selected based on the criteria from Solomon (1987) as in Macrina et al. (2019). We first find the best insertion position p^*_u for each unrouted customer u by computing the criterion c_1 as follows:

$$c_1(i, u, j) = \gamma_1 c_{11}(i, u, j) + \gamma_2 c_{12}(i, u, j), \tag{7}$$

$$c_{11}(i, u, j) = d_{iu} + d_{uj} - \mu d_{ij}, \tag{8}$$

$$c_{12}(i, u, j) = b_{j_u} - b_j, \tag{9}$$

where b_j (hours) denotes the service start time at customer j in the current route, and b_{j_u} (hours) the service start time at customer j when we insert customer u into the current route at the position right before j. The parameters μ, γ_1, γ_2, and θ are all non-negative, and $\gamma_1 + \gamma_2 = 1$. The best insertion position for each $u \in C$ is defined by $p^*_u = \text{argmin}_{p \in \{1,...m\} \cup \{t\}} \{c_1(i_{p-1}, u, i_p)\}$.

Next, we computing the criterion c_2 for each unrouted customer $u \in C$ as follows:

$$c_2(u, p^*_u) = \theta d_{su} - c_1(i_{p^*_u - 1}, u, i_{p^*_u}). \tag{10}$$

The best customer u^* to be inserted into their corresponding best position $p^*_{u^*}$ is the one with the highest c_2 value, i.e., $u^* = \text{argmax}_{u \in C}\{c_2(u, p^*_u)\}$. If the insertion is feasible, i.e., it satisfies the fleet size, load capacity, time windows, and emissions constraints, then we insert u^* into the position $p^*_{u^*}$ of the current route, and start a new iteration of finding the next customer to insert. Otherwise, we add the current route to the set of ICCV-routes η_C and evaluate the feasibility of initialising a new ICCV-route.

The stopping criteria for the insertion heuristic is when all customers in C are routed, or when there are unrouted customers and the initialisation of a new route violates the emissions constraint. Note that, if the current route cannot serve any customers, i.e. it only contains the depot, then it is considered as invalid, meaning that we cannot initialise any other feasible route. If there are any unrouted customers left after the insertion heuristic for ICCV-routes terminates, we add them to the set E.

Different from Macrina et al. (2019), we take into account the customers that must be served by the ICCVs (i.e., customers in C'). These customers are prioritised in C (after C is sorted) and thereby during the routing for ICCVs. If the insertion heuristic for ICCV-routes terminates when some customers in C' are still unrouted, then we remove the customers that are not in C' in the last ICCV-route until it is feasible (i.e., the emissions constraint is not violated) to initialise a new ICCV-route with the unrouted customers in C'.

4.1.3 Insertion heuristic for ECV-routes

The insertion heuristic for ECV-routes follows the same framework of that for ICCV-routes, except for the insertion stopping criterion. In particular, it is feasible to insert the best customer u^* into the associated best position $p^*_{u^*}$ of the current route if the insertion satisfies the fleet size, load capacity, time windows, and energy constraints. Every time a customer is inserted into the current route, a `labeling` algorithm is applied to find the cheapest feasible insertion of CSs. The insertion heuristic for ECV-routes terminates when all customers in the set E are routed. Since every customer can be routed with at least an ECV, we are guaranteed that the aforementioned procedure always find an ECV solution η_E.

4.2 Local search

As in Macrina et al. (2019), we employ three inter-route local search operators: The ICCV-Relocate, in which a customer of an ICCV-route is relocated to another ICCV-route; the ECV-Relocate which moves a customer of an ECV-route to another ECV-route; and the general Relocate operator which relocates a customer from a route to another. We let the operators exhaustively explore the local search space, e.g., the ICCV-Relocate operator looks for the best customer to insert into the new best position in another ICCV-route such that the total cost of the solution is the minimal.

The `localSearch` procedure is described in Algorithm 3, where we define $f(\eta)$ the cost of a solution η. First, we randomly select one among the aforementioned local search operators to operate on the current solution η. Then, we iteratively apply the selected operator to improve the solution until a new solution cannot reduce the current cost anymore.

It is important to mention that a relocation from or to an ECV-route requires the adjustment of the CSs in that route and the amount of energy being recharged, if any. Therefore, whenever

Algorithm 3 The local search procedure

```
procedure LOCALSEARCH($\eta$)
    lsOperator ← Randomly selected among ICCV-Relocate, ECV-Relocate, and Relocate
    $\eta^* \leftarrow \eta$
    $\eta' \leftarrow$ lsOperator($\eta$)
    while $f(\eta') < f(\eta^*)$ do
        $\eta^* \leftarrow \eta'$
        $\eta' \leftarrow$ lsOperator($\eta$)
    end while
    return $\eta^*$
end procedure
```

a customer is removed from or inserted into an ECV-route, the `labeling` algorithm will be performed to find the cheapest feasible solution. If the aforementioned algorithm cannot find any feasible solution for the case of an insertion into an ECV-route, then the insertion is counted as invalid and thus, the operator will move on to other positions/ routes.

4.3 Perturbation

As in Macrina et al. (2019), we reuse the three relocate operators described in Section 4.2 as perturbation operators to escape the local optimum, with the exception that we accept more expensive solutions to explore different neighborhoods. Furthermore, we employ two additional operators that are not included in Macrina et al. (2019), namely the Random-Permute which randomly selects and destroys an ICCV- and an ECV-route, then routes the customers from the destroyed routes in a new order; and the Cyclic-Exchange operator which randomly selects two routes from the current solution and cyclically exchange a sequence of customers from each route. These two operators allow for randomness to diversify the solution space, which potentially leads to better solutions. We verify this statement by testing the ILS-SP with and without the Random-Permute and Cyclic-Exchange operators. According to our preliminary results, including the additional perturbation operators improves the solution costs for most cases. The mechanisms of the additional operators are described in Appendix B.

During each iteration, a perturbation operator is considered to be feasible if it can perturb the current solution. For example, if the current solution contains only ECV-routes, then the ICCV-Relocate is not a feasible perturbation operator. Every time the `perturbation` procedure is called, a list of perturbation operators is constructed to include only the feasible operators. Then, the algorithm will randomly select one among the operators in the list.

It is important to note that the Random-Permute and the Cyclic-Exchange operators, even though they can be considered as feasible operators at the beginning of the procedure, might not be able to perturb the current solutions (i.e., they cannot find any feasible perturbation). In such case, the `perturbation` procedure will remove the currently-employed operator from the list of feasible operators and randomly select a new one. This step allows the `perturbation` procedure to always perturb the current solution to search for new neighborhoods.

4.4 The Labeling algorithm

Different from Macrina et al. (2019), we employ a `labeling` algorithm based on that of Zhao and Lu (2019) to optimise the decision on inserting/removing the CSs and on the amount of energy recharged for each ECV-route while ensuring the feasibility of the energy constraints. As described in the previous sections, the `labeling` algorithm is used in the insertion heuristic for ECV-routes, the local search and the perturbation procedures. The input for this algorithm includes the set of CSs R, and an ECV-route $r^E = (s, v_1, v_2, ..., v_m, t)$ containing only the depots and customers, in which the load capacity and time windows constraints are satisfied, but the energy constraints might be violated. Our labeling algorithm considers a graph illustrated by Figure 1, and looks for the cheapest s,t-path such that all customers are served in time and the energy constraints are satisfied.

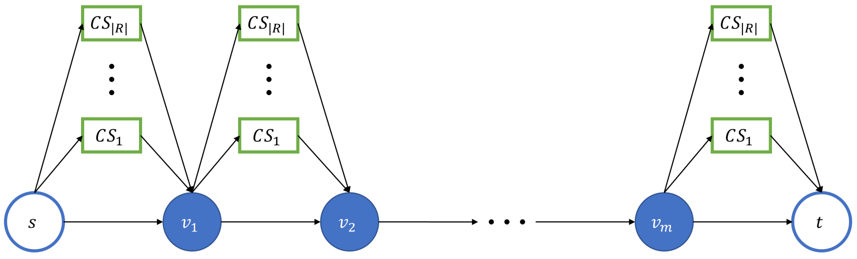


Figure 1: Input graph for the labeling algorithm

Following the notations from Zhao and Lu (2019), we define a label of customer $i \in r^E$ by $l_i = (t_i^a, z_i, f_i, i, l)$, where t_i^a is the arrival time at customer i (hour), z_i is the energy level upon the arrival at i (kWh), f_i is the accumulated cost upon the arrival at i (€), and l is the predecessor label of l_i. Each vertex $i \in r^E$ is associated with a set L_i containing all valid labels of i. Travelling from customer i to the successive customer $j \in r^E$, a new label $l_j = (t_j^a, z_j, f_j, j, l_i)$ is created based on two scenarios: First, the ECV travels directly from i to j, resulting in at most a single new label. Second, the ECV stops by a CS before reaching j, therefore generating at most $|R|$ new labels. The pseudocode for each scenario is provided in Appendix C.

Different from Zhao and Lu (2019) who allows for constraint violations by adding penalty terms to the cost, we only consider valid labels in our algorithm. A label is valid only if it violates neither the time windows nor the energy constraints. Furthermore, a label $l_i = (t_i^a, z_i, f_i, i, l)$ of customer i dominates a label $l'_i = (t_i^{a'}, z'_i, f'_i, i, l')$ of the same customer only if the following conditions hold: $t_i^a \leq t_i^{a'}$, $z_i \geq z'_i$, $f_i \leq f'_i$, and there is at least a strict inequality.

In the second scenario, the ECV travels from customer v_{i-1} to a CS k before reaching customer v_i. Here, it is crucial to consider how much energy the ECV should recharge at k, as this will influence the time windows constraints of all customers served after v_i. We assume that the ECV is recharged as much as needed to travel to the successor vertex of v_i, i.e., customer v_{i+1}. However, if this amount of recharge results in the violation of the time window of v_i, then the ECV is recharged as much as allowed such that it can reach v_i in time.

For notation, we let $v_0 = s$ and $v_{m+1} = t$. The algorithm starts with the initial label $l_0 = (0, B^E, w^a, v_0, \texttt{null})$ of the set L_0, where `null` indicates that l_0 does not have a predecessor label. Then, for each label l of every vertex $v_{i-1}, i \in \{1, ...m+1\}$, we create new labels for the successor vertex v_i based on the aforementioned scenarios. Only valid labels which are not dominated by any other labels of v_i are added to the set L_{v_i}. When all labels are found, we look for the label $l^*_{v_{m+1}} \in L_{v_{m+1}}$ whose cost is the lowest, and trace back to v_0 via the predecessor labels to find the cheapest feasible v_0, v_{m+1}-path. We consider a sequence of customers to be infeasible if the `labeling` algorithm cannot find any v_0, v_{m+1}-path that visits all customers. The pseudocode for the labeling algorithm is provided in Appendix C.

4.5 Solving the Set Partitioning formulation

At the final stage of the ILS-SP, we will solve the Set Partitioning (SP) formulation introduced in Section 3.2. In the remaining of this paper, we refer to this stage as the SP procedure which takes in the set Ω containing all routes found in each ILS iteration. Each of these routes must satisfy the time windows constraints, the SoC constraints, and the load capacity constraint. During the SP procedure, we solve the SP formulation (2) using CPLEX Studio 22.1.0. and the final solution is obtained.

5 Computational study

As in Macrina et al. (2019), the test instances used in this paper are the VRPTW instances introduced in Schneider, Stenger and Goeke (2014), which were originally created by Solomon (1987) based on three different types of customer geographical distribution: clustered (C), random (R), and both clustered and random (RC). Additionally, C1, R1, and RC1 denotes a short scheduling horizon, whereas C2, R2, and RC2 denotes a long scheduling horizon. To solve the E-VRPTW, Schneider et al. (2014) added to each instances a set of 21 CSs whose locations were set randomly and computed new feasible time windows. As in Macrina et al. (2019), we consider three sets of instances. Set 1 consists of the 36 small-sized instances (5, 10, or 15 customers) from Schneider et al. (2014). Set 2 consist of medium-sized instances with 25, 30, or 50 customers. Lastly, Set 3 consists of 15 large-sized instances with 100 customers and short scheduling horizon from Schneider et al. (2014). To achieve Set 2, we keep the 21 CSs and select the first 25, 30, and 50 customers from each large-sized instance in Set 3.

The load capacities of vehicles (of both types) are included in the instance. In addition, we assume that the numbers of available vehicles are infinite. The vehicle speed is included in the instances, such that the feasibility of the instances is ensured. Furthermore, following Solomon (1987), we assume that the travelled time t_{ij} on an arc $(i, j) \in A$ is equal to the distance travelled d_{ij} on this arc, where d_{ij} is determined using the Euclidean distance metric.

To account for the emissions generated, we define UB_{max} as the total emissions in the worst case in which the emissions constraint is relaxed and only ICCVs are employed. For each instance, we find UB_{max} by solving a mixed-integer programming (MIP) formulation with CPLEX Studio 22.1.0. The aforementioned MIP formulation is provided in Appendix D, together with the details regarding the gap we set. The upper bound UB on the amount of CO_2 emissions is set to αUB_{max}, in which $\alpha \in \{0.25, 0.5, 0.75\}$ indicates three level of allowed emissions as in Macrina et al. (2019). An instance denoted as, for example, C101C5 (0.25) means that it is of type C1 (clustered customer geographical distribution with short scheduling horizon), the instance ID is 01, the number of customers is five (C5), and $\alpha = 0.25$.

In the clustering algorithm, we set $\lambda = 0.5$. In the insertion heuristics, we let $\theta = 1$ to account for the case where a new route is initialised for the customer in consideration instead of inserting the aforementioned customers into an existing route, and $\gamma_1 = \gamma_2 = 0.5$ to balance the weights on distance travelled and time windows. As Macrina et al. (2019) found that performing 200 iterations in their ILS heuristic leads to better trade-off between solution quality and executing time compared to 100, 150, and 250 iterations, we also restrict the number of iterations in the ILS-SP to 200. Furthermore, the ECV battery is fixed to 16 kWh for all instances since we use the estimated charging functions found by Montoya et al. (2017). Following our preliminary results, the coefficient of energy consumption $\pi = 1$ kWh/km as in Macrina et al. (2019) leads to infeasible problem for some instances (especially with $\alpha = 0.25$), and $\pi = 0.125$ kWh/km as in Montoya et al. (2017) leads to the ECVs mostly have sufficient energy to visit all customers on their routes, or they require very little amount of recharged energy. Therefore, we set $\pi = 0.25$ kWh/km as this value offers greater insights into the charging decisions of the ILS-SP.

5.1 Charging functions

In the linear scenario, we assume that all CSs employ the moderate technology from Felipe et al. (2014). In particular, the charging speed is assumed to be 20 kWh/h for all CS. In the nonlinear scenario, we make use of the recharging function found in Montoya et al. (2017) for the moderate technology and the battery capacity of 16 kWh. The function consists of the following breakpoints: (0, 0), (0.62, 13.6), (0.77, 15.2), and (1.01, 16), in which the first value indicates the time in hours, and the second value indicates the energy level in kWh. We also use the moderate technology charging cost of 0.176 €/kWh from Felipe et al. (2014) in all instances and charging scenarios since Montoya et al. (2017) did not mention the cost of recharging.

Figure 2 shows the linear and the estimated nonlinear charging functions corresponding to the moderate technology at all CSs. To illustrate the estimation of charging time in the case of nonlinear recharge, we assume an ECV arrives at a CS when its energy level is 8 kWh, and the target level is 14.4 kWh. According to our nonlinear charging function, the corresponding times are 0.36 and 0.70, respectively. The recharging time is therefore $\Delta = 0.70 - 0.36 = 0.34$ (hours).

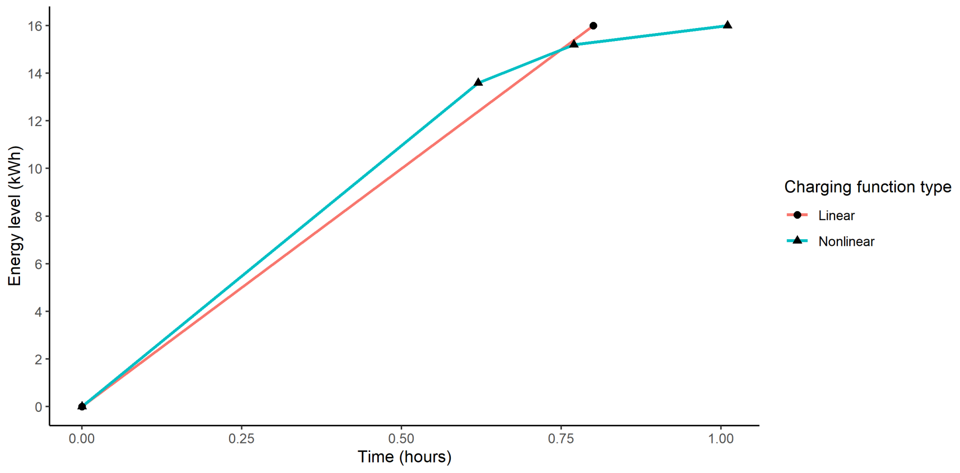


Figure 2: The linear and nonlinear charging functions corresponding to the moderate technology

5.2 Linear recharge

This section discusses the results on all test instances under the linear recharging assumption. We evaluate the ILS-SP based on the solution quality and the execution time. For each instance and level of emissions constraint (α), we obtain the cost of the initial solution (initial cost), the cost of the solution found by the ILS procedure (ILS cost), and the cost of the solution found by solving the SP formulation (SP cost). To examine the performance of the ILS-SP, we introduce two improvement metrics. First, the percentage decreasing from the initial cost to the SP cost, in which we denoted by M_1. Second, the percentage decreasing from the ILS cost to the SP cost, denoted by M_2. Table 1 presents the results obtained with the ILS-SP for the small-sized instances. We report the SP costs (in €) and the running time (in seconds). Below each SP cost, the corresponding metrics M_1 and M_2 is presented in the parentheses, respectively.

Table 1: Solution costs for small-sized instances under linear recharge

	5 customers							**10 customers**							**15 customers**					
	$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$			$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$			$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$	
Instance	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**	**Instance**	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**	**Instance**	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**
C101C5	281.03	1	264.64	0	264.64	0	C101C10	446.6	1	424.01	0	371.17	0	C103C15	404.76	1	392.14	0	381.65	0
	(14.48%; 0%)		*(0%; 12.83%)*		*(12.17%; 4.73%)*			*(26.17%; 0%)*		*(9.86%; 4.36%)*		*(0.08%; 21.41%)*			*(27.56%; 15.29%)*		*(26.21%; 19%)*		*(37.53%; 14.33%)*	
C103C5	172.09	0	164.87	0	164.87	0	C104C10	367.91	0	355.4	0	318.49	0	C106C15	355.23	0	318.88	0	317.6	0
	(14.32%; 0%)		*(0%; 0%)*		*(0%; 0%)*			*(20.67%; 0%)*		*(12.29%; 1.54%)*		*(20.98%; 11.77%)*			*(39.19%; 0.49%)*		*(30.35%; 17.55%)*		*(27.8%; 15.98%)*	
R104C5	168.91	0	143.75	0	143.75	0	R102C10	281.36	0	284.18	0	253.06	0	R102C15	431.52	0	400.53	0	389.05	0
	(16.42%; 0%)		*(18.75%; 0%)*		*(18.75%; 0%)*			*(15.26%; 0%)*		*(22.23%; 0%)*		*(17.22%; 10.06%)*			*(20.95%; 0%)*		*(31.37%; 7.91%)*		*(20.41%; 13.62%)*	
R105C5	191.79	0	188.97	0	156.3	0	R103C10	216.15	1	201.47	0	198.61	0	R105C15	384.3	0	356.04	0	317.92	0
	(0%; 13.18%)		*(0%; 0%)*		*(0%; 15.5%)*			*(35.16%; 1.69%)*		*(3.41%; 8.56%)*		*(11.95%; 7.51%)*			*(19.6%; 0%)*		*(24.5%; 5.92%)*		*(20.69%; 12.22%)*	
RC105C5	257.63	0	254.54	0	250.58	0	RC102C10	421.05	0	384.75	0	384.75	0	RC103C15	452.39	0	416.16	0	397.95	0
	(3.92%; 0%)		*(1.76%; 0%)*		*(0%; 0%)*			*(6.46%; 0%)*		*(19.12%; 0%)*		*(19.12%; 0%)*			*(30.19%; 0.64%)*		*(29.61%; 8.98%)*		*(30.06%; 11.08%)*	
RC108C5	282.17	0	302.03	0	259.88	0	RC108C10	405.35	0	355.55	0	344.77	0	RC108C15	531.45	0	419.45	0	404.39	0
	(0%; 0%)		*(0%; 11.68%)*		*(0%; 0%)*			*(12.6%; 3.15%)*		*(10.19%; 12.52%)*		*(8.37%; 5.46%)*			*(29.03%; 14.04%)*		*(30.07%; 5.26%)*		*(27.05%; 20.79%)*	
C206C5	269.72	0	252.34	0	252.34	0	C202C10	299.64	0	297.58	0	245.24	0	C202C15	521.6	0	471.2	0	488.02	0
	(0%; 15.58%)		*(0%; 0%)*		*(0%; 0%)*			*(26.29%; 6.19%)*		*(27.27%; 3.83%)*		*(0%; 22.03%)*			*(28.8%; 4.15%)*		*(37.46%; 13.57%)*		*(26.59%; 4.56%)*	
C208C5	262.53	0	262.53	0	233.38	0	C205C10	367.91	0	326.43	0	262.22	0	C208C15	514.55	0	312.74	0	308.24	0
	(27.91%; 0%)		*(22.37%; 16.88%)*		*(34.54%; 0%)*			*(19.58%; 11.67%)*		*(17.75%; 21.78%)*		*(10.81%; 35.63%)*			*(20.72%; 10.73%)*		*(16.44%; 26.13%)*		*(21.51%; 22.48%)*	
R202C5	189.4	0	183.65	0	176.92	0	R201C10	251.4	0	234.5	0	224.31	0	R202C15	461.54	1	396.09	0	392.55	0
	(0%; 0%)		*(0%; 0%)*		*(18.16%; 0%)*			*(28.53%; 4.77%)*		*(17.84%; 1.27%)*		*(18.28%; 0%)*			*(35.54%; 2.78%)*		*(32.8%; 6.96%)*		*(31.1%; 6.24%)*	
R203C5	221.12	0	254.57	0	206.99	0	R203C10	289.74	0	277.48	0	259.88	0	R209C15	438.47	1	350.66	1	316.87	0
	(6.66%; 0.13%)		*(4.29%; 0%)*		*(21.74%; 0%)*			*(37.09%; 0%)*		*(20.4%; 12.3%)*		*(40.57%; 0%)*			*(30.01%; 2.33%)*		*(39.27%; 0.21%)*		*(41.71%; 3.06%)*	
RC204C5	251.8	0	251.8	0	212.57	0	RC201C10	356.91	0	335.2	0	315.34	0	RC202C15	508.38	0	450.98	0	433.13	0
	(1.44%; 0%)		*(23.21%; 0%)*		*(28.06%; 1.24%)*			*(10.58%; 19.86%)*		*(18.2%; 2.65%)*		*(7.59%; 18.93%)*			*(29.49%; 2.73%)*		*(31.65%; 7.99%)*		*(35.34%; 12.28%)*	
RC208C5	212.12	0	212.12	0	222.6	0	RC205C10	451.01	0	425.01	0	350.17	0	RC204C15	482.73	1	404.88	2	324.83	1
	(25.58%; 0%)		*(27.59%; 0%)*		*(5.39%; 0%)*			*(27.23%; 0%)*		*(17.02%; 15.85%)*		*(26.1%; 24.44%)*			*(29.76%; 2.83%)*		*(28.18%; 10.35%)*		*(30.34%; 28.72%)*	
Average	230.03	0.08	227.98	0	212.07	0	Average	346.25	0.17	325.13	0	294	0	Average	457.24	0.33	390.81	0.25	372.68	0.08
	(9.23%; 2.41%)		*(8.16%; 3.45%)*		*(11.57%; 1.79%)*			*(22.14%; 3.94%)*		*(16.3%; 7.06%)*		*(15.09%; 13.1%)*			*(28.4%; 4.67%)*		*(29.83%; 10.82%)*		*(29.18%; 13.78%)*	

The ILS-SP is capable of improving both the initial and the ILS solutions for the majority of the small-sized instances. Considering the case of five customers, we find 10 among 36 instances where the `initialisation` procedure performs relatively well, in the sense that it directly yields the SP solution (i.e., 0% decreasing in cost for both metrics). When the number of customers increases to 10 and then 15, the values for M_1 and M_2 metrics also increase. These findings indicate that for instances with larger size and higher complexity, the `initialisation` procedure becomes less suitable to optimise the routing, while the SP procedure demonstrates a greater impact on improving the ILS solutions.

In addition, we obtain for the majority of the small-sized instances lower costs for higher values of α. The underlying cause for the aforementioned finding is likely the higher expenses associated with employing ECVs, primarily due to the activating and recharging costs at CSs, together with the needs for more ECVs when stricter upper bounds on the total CO_2 emissions (lower α) are imposed. Our matheuristic is also time-efficient as it finds the optimal solution within a second for most small-sized instances.

To analyse the effect of long and short scheduling horizon, we consider the average performance of the ILS-SP on each type of horizon for the small-sized instances in Table 2. Overall, the ILS procedure achieves higher improvement on the initial solutions (i.e., higher M_1 values) in the case of long scheduling horizon for every instance size and level of emissions restriction.

Table 2: The performance based on short and long scheduling horizon for small-sized instances

		$\alpha = 0.25$		$\alpha = 0.5$		$\alpha = 0.75$	
		Cost	**Time**	**Cost**	**Time**	**Cost**	**Time**
Short scheduling horizon	$\lvert N\rvert = 5$	225.60 *(8.19%; 2.2%)*	0.17	219.80 *(3.42%; 4.09%)*	0.00	206.67 *(5.15%; 3.37%)*	0.00
	$\lvert N\rvert = 10$	356.40 *(19.39%; 0.81%)*	0.33	334.23 *(12.85%; 4.5%)*	0.00	311.81 *(12.95%; 9.37%)*	0.00
	$\lvert N\rvert = 15$	426.61 *(27.75%; 5.08%)*	0.17	383.87 *(28.69%; 10.77%)*	0.00	368.09 *(27.26%; 14.67%)*	0.00
	Average	336.21 *(18.44%; 2.7%)*	0.22	312.63 *(14.99%; 6.45%)*	0.00	295.52 *(15.12%; 9.14%)*	0.00
Long scheduling horizon	$\lvert N\rvert = 5$	234.45 *(10.27%; 2.62%)*	0.00	236.17 *(12.91%; 2.81%)*	0.00	217.47 *(17.98%; 0.21%)*	0.00
	$\lvert N\rvert = 10$	336.10 *(24.88%; 7.08%)*	0.00	316.03 *(19.75%; 9.61%)*	0.00	276.19 *(17.23%; 16.84%)*	0.00
	$\lvert N\rvert = 15$	487.88 *(29.05%; 4.26%)*	0.50	397.76 *(30.97%; 10.87%)*	0.50	377.27 *(31.1%; 12.89%)*	0.17
	Average	352.81 *(21.4%; 4.65%)*	0.17	316.65 *(21.21%; 7.76%)*	0.17	290.31 *(22.1%; 9.98%)*	0.06

Table 3 shows the results of the medium-sized instances. As the complexity of the instances increases, the SP cost becomes more expensive whereas the effectiveness of the `initialisation` procedure declines. On average, the instances with the strictest upper bound on the emissions, i.e., $\alpha = 0.25$, tend to achieve lower level of improvement. Nevertheless, the SP procedure reduces the ILS cost for the majority of the instances, emphasising its benefit as an improvement step in our matheuristic. Even though the executing time increases with respect to the size of the instance, the ILS-SP is still capable of solving the medium-sized instances within a minute.

Table 3: Solution costs for medium-sized instances under linear recharge

	25 customers							**30 customers**							**50 customers**					
	$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$			$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$			$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$	
Instance	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**	**Instance**	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**	**Instance**	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**
C101C25	306.75	5	321.24	4	303.94	4	C101C30	339.63	7	332.66	6	342.45	6	C101C50	584.67	19	481.08	17	499.46	18
	(36.03%; 14.01%)		*(21.73%; 6.8%)*		*(31.18%; 18.91%)*			*(22.57%; 8.03%)*		*(28.01%; 3.21%)*		*(25.97%; 17.12%)*			*(19.56%; 5.56%)*		*(25.32%; 10.26%)*		*(35.73%; 5.47%)*	
C102C25	300.99	7	301.16	6	301.47	7	C102C30	333.97	10	333.97	10	282.57	10	C102C50	650.46	30	541.71	33	498.06	33
	(43.86%; 14.8%)		*(38.31%; 22.47%)*		*(43.76%; 23.58%)*			*(56.71%; 8.15%)*		*(62.73%; 0%)*		*(64.3%; 2.32%)*			*(44.73%; 12.96%)*		*(55.55%; 11.01%)*		*(56.48%; 11.26%)*	
C103C25	296.95	10	296.95	10	297.23	10	C103C30	333.34	12	318.73	13	298.8	14	C103C50	577.88	43	568.31	42	489.7	40
	(42.6%; 3.82%)		*(43.81%; 7.36%)*		*(42.16%; 12.17%)*			*(55.8%; 3.55%)*		*(57.91%; 0.8%)*		*(55.41%; 11.6%)*			*(52.1%; 6.54%)*		*(43.62%; 11.61%)*		*(52.2%; 15.03%)*	
C104C25	270.23	11	264.95	12	265.56	12	C104C30	326.77	16	286.77	21	285.99	18	C104C50	570.59	48	522.04	50	497.43	53
	(35.67%; 14.09%)		*(31.07%; 14.98%)*		*(26.53%; 11.99%)*			*(31.2%; 10.22%)*		*(41.97%; 12.06%)*		*(37.79%; 11.58%)*			*(26.18%; 22.56%)*		*(27.63%; 30.95%)*		*(41.8%; 12.2%)*	
C105C25	306.13	6	291.83	5	291.62	6	C105C30	365.73	8	325.89	9	325.33	9	C105C50	561.41	24	481.23	23	481.4	24
	(32.85%; 0.15%)		*(22.82%; 15.33%)*		*(40.83%; 5.3%)*			*(37.61%; 0.11%)*		*(44.38%; 2.32%)*		*(37.32%; 0%)*			*(35.28%; 2.9%)*		*(33.16%; 10.86%)*		*(27.15%; 10.73%)*	
R101C25	642.07	3	613.12	3	603.96	3	R101C30	715.85	6	698.96	6	689.65	5	R101C50	1131.24	14	1100.58	13	1062.58	13
	(30.93%; 3.02%)		*(34.1%; 0.38%)*		*(33.42%; 1.72%)*			*(30.01%; 4.83%)*		*(31.75%; 3.6%)*		*(30.4%; 2.42%)*			*(35.86%; 3.78%)*		*(37.85%; 7.13%)*		*(38.06%; 7.47%)*	
R102C25	557.89	8	526.39	8	530.46	8	R102C30	636.85	13	615.94	13	589.55	13	R102C50	1084.31	25	966.48	26	968.99	26
	(33.34%; 1.81%)		*(32.99%; 7.04%)*		*(35.3%; 5.25%)*			*(32.78%; 1.94%)*		*(27.84%; 2.32%)*		*(26.64%; 5.75%)*			*(39.53%; 0.5%)*		*(42.33%; 5.29%)*		*(48.14%; 5.32%)*	
R103C25	520.83	12	496.21	11	480.29	11	R103C30	587.45	18	520.4	17	526.23	18	R103C50	888.53	45	814.26	44	839.29	39
	(22.96%; 2.55%)		*(29.16%; 5.24%)*		*(28.93%; 9.37%)*			*(15.41%; 3.94%)*		*(23.19%; 8.23%)*		*(14.15%; 14.5%)*			*(38.08%; 4.84%)*		*(43.05%; 5.6%)*		*(44.31%; 2.71%)*	
R104C25	522.02	12	506.76	12	461.43	10	R104C30	536.81	19	507.25	19	486.78	17	R104C50	860.28	54	825.96	53	837.8	51
	(18.02%; 3.62%)		*(23.33%; 1.79%)*		*(24.57%; 8.56%)*			*(15%; 13.53%)*		*(24.28%; 9.11%)*		*(19.83%; 14.73%)*			*(31.68%; 5.85%)*		*(29.03%; 9.41%)*		*(31.61%; 2.18%)*	
R105C25	557.09	4	541.98	4	531.19	4	R105C30	615.58	7	603.26	7	557.96	6	R105C50	1067.49	19	1029.76	19	966.45	19
	(21.82%; 5.98%)		*(12.98%; 3.71%)*		*(15.56%; 4.81%)*			*(23.57%; 2.79%)*		*(21.57%; 0%)*		*(22.18%; 5.93%)*			*(30.24%; 0.59%)*		*(31.11%; 3.61%)*		*(31.14%; 3.21%)*	
RC101C25	569.41	2	500.06	2	490.5	2	RC101C30	1163.67	5	799.46	4	786.09	3	RC101C50	1382.83	14	1084.55	10	992.66	7
	(1.23%; 20.04%)		*(3.96%; 14.26%)*		*(0.82%; 17.29%)*			*(0.67%; 10.82%)*		*(12.02%; 3.37%)*		*(12.57%; 3.76%)*			*(35.88%; 5.05%)*		*(41.68%; 6.35%)*		*(42.05%; 10.07%)*	
RC102C25	596.6	5	469.31	3	454.98	2	RC102C30	1078.96	7	1056.22	7	684.78	3	RC102C50	1280.72	16	964.39	12	900.74	16
	(30.94%; 3.81%)		*(31.17%; 20.26%)*		*(30.14%; 5.24%)*			*(18.36%; 8.97%)*		*(13.56%; 7.18%)*		*(41.14%; 10.21%)*			*(35.53%; 0.07%)*		*(30.75%; 13.84%)*		*(30.98%; 14.72%)*	
RC103C25	596.07	6	516.51	4	394.35	2	RC103C30	1054.97	6	759.26	5	626.66	4	RC103C50	1310.85	25	839.58	23	794.83	14
	(34.54%; 4.58%)		*(24.82%; 13.59%)*		*(16.21%; 17.81%)*			*(15.96%; 8.33%)*		*(11.69%; 29.21%)*		*(27.34%; 12.29%)*			*(26.73%; 21.76%)*		*(57.66%; 14.91%)*		*(50.54%; 14.3%)*	
RC104C25	586.91	7	509.16	5	392.13	4	RC104C30	1051.49	7	873.4	7	916.43	7	RC104C50	1611.1	24	830.68	23	696.77	20
	(28.44%; 1.85%)		*(31.75%; 3.33%)*		*(20.13%; 4.07%)*			*(12.85%; 7.1%)*		*(19.61%; 18.27%)*		*(13.21%; 8.49%)*			*(24.24%; 4.16%)*		*(44.34%; 33.25%)*		*(41.52%; 43.97%)*	
RC105C25	741.31	4	658.8	3	485.18	2	RC105C30	1214.02	5	1066.99	5	919.88	4	RC105C50	1747.64	19	940.5	10	830.09	10
	(27.09%; 2.48%)		*(27.7%; 4.44%)*		*(14.83%; 14.55%)*			*(16.36%; 2.34%)*		*(9.43%; 7.91%)*		*(13.49%; 20.61%)*			*(20.36%; 6.29%)*		*(28.22%; 13.62%)*		*(39.33%; 18.79%)*	
Average	491.42	6.8	454.3	6.13	418.95	5.8	Average	690.34	9.73	606.61	9.93	554.61	9.13	Average	1020.67	27.93	799.41	26.53	757.08	25.53
	(29.35%; 6.44%)		*(27.31%; 9.4%)*		*(26.96%; 10.71%)*			*(25.66%; 6.31%)*		*(28.66%; 7.17%)*		*(29.45%; 9.42%)*			*(33.07%; 6.89%)*		*(38.09%; 12.51%)*		*(40.74%; 11.83%)*	

The results for the large-sized instances with 100 customers are presented in Table 4. The average costs are around two times higher than that of the 50-customer instances for each level of restriction on the total emissions. The average improvement on the initial costs is also higher than that in the medium-sized instances, indicating that the ILS-SP can handle well problems with high level of complexity. All values corresponding to the M_2 metric are positive, once again highlighting the advantage of the SP procedure. Our matheuristic terminates within around five minutes for the large-size instances.

Table 4: Solution costs for large-sized instances under linear recharge

	$\alpha = 0.25$			$\alpha = 0.50$			$\alpha = 0.75$		
Instance	**Cost**		**Time**	**Cost**		**Time**	**Cost**		**Time**
C101C100	1717.89	*(28.56%; 12.11%)*	98	1539.14	*(46.29%; 5.35%)*	94	1266.98	*(40.05%; 13.71%)*	116
C102C100	1416.68	*(58.52%; 9.11%)*	177	1188.43	*(53.31%; 19.18%)*	180	1115.87	*(58.63%; 18.31%)*	150
C103C100	1460.38	*(37.53%; 28.96%)*	245	1252.47	*(47.49%; 23.09%)*	267	1223.89	*(50.2%; 19.91%)*	234
C104C100	1297.86	*(40.31%; 31.61%)*	319	1290.6	*(42.53%; 15.83%)*	340	1058.29	*(48.57%; 20.92%)*	208
C105C100	1552.89	*(43.6%; 2.4%)*	108	1510.85	*(43.73%; 8.89%)*	117	1411.44	*(49.63%; 4.49%)*	135
R101C100	1999.87	*(38.14%; 2.35%)*	52	1903.86	*(36.58%; 3.78%)*	52	1796.4	*(36.54%; 4.38%)*	45
R102C100	1794.22	*(41.1%; 2.74%)*	98	1576.99	*(45.94%; 6.83%)*	100	1571.58	*(42.22%; 5.95%)*	94
R103C100	1538.49	*(42.84%; 1.74%)*	163	1485.89	*(41.51%; 4.11%)*	161	1379.27	*(44.03%; 7.86%)*	179
R104C100	1392.93	*(29.72%; 6.72%)*	243	1199.64	*(27.74%; 7.81%)*	227	1194.12	*(33.5%; 3.68%)*	206
R105C100	1669.35	*(36.04%; 3.81%)*	78	1561.77	*(39.71%; 3.11%)*	76	1431	*(43.03%; 1.4%)*	69
RC101C100	2341.69	*(40.24%; 8.36%)*	83	2146.4	*(42.78%; 8.25%)*	71	1889.14	*(48.08%; 6.67%)*	71
RC102C100	2093.67	*(36.91%; 19.43%)*	116	2085.64	*(35.7%; 12.65%)*	112	1835.82	*(35%; 25.13%)*	98
RC103C100	2047.61	*(32.57%; 19.75%)*	136	1838.16	*(41.99%; 13.12%)*	140	1721.27	*(36.06%; 20.16%)*	138
RC104C100	2115.14	*(39.75%; 0.08%)*	188	2076.35	*(36.47%; 1.98%)*	187	1452.14	*(41.84%; 25.26%)*	200
RC105C100	1903.49	*(46.57%; 12.11%)*	115	1758.71	*(40.94%; 23.7%)*	102	1634.62	*(55.36%; 2.38%)*	69
Average	1756.14	*(39.49%; 10.75%)*	147.93	1627.66	*(41.51%; 10.51%)*	148.4	1465.46	*(44.18%; 12.01%)*	134.13

Table 5 summarises the results for all classes of instances on each level of emissions restriction. It is clear that the SP cost and executing time increase with respect to the complexity of the instances. In addition, the ILS-SP finds more expensive costs for low values of α in all classes of instances. Indeed, imposing lower upper bounds on the CO_2 emissions leads to the need for more ECVs, which are more expensive than ICCVs due to the activation and recharging costs.

The aforementioned claim is supported by the results in Table 6, in which we present the impact of the ECVs on the solution costs for medium- and large-sized instances. The lower the value of α, the higher contribution to the overall costs of the ECVs. We also find that the influence of the ECVs decreases for more complex instances. In particular, the ECVs contribute on average 94.38% to the solution costs of the 25-customer instances, whereas this value drops to 82.84% for the case of 100 customers.

Table 5: Summary for all classes of instances under linear recharge

Small	α	**Cost**	**Time**	**Medium**	α	**Cost**	**Time**	**Large**	α	**Cost**	**Time**
	0.25	230.03	0.08		0.25	491.42	6.8		0.25	1756.15	147.93
$\lvert N \rvert = 5$	0.5	227.98	0	$\lvert N \rvert = 25$	0.5	454.30	6.13	$\lvert N \rvert = 100$	0.5	1627.66	148.4
	0.75	212.07	0		0.75	418.95	5.8		0.75	1465.46	134.13
	Average	223.36	0.03		Average	454.89	6.24		Average	1616.42	143.49
	0.25	346.25	0.17		0.25	690.34	9.73				
$\lvert N \rvert = 10$	0.5	325.13	0	$\lvert N \rvert = 30$	0.5	606.61	9.93				
	0.75	294.00	0		0.75	554.61	9.13				
	Average	321.79	0.06		Average	617.19	9.60				
	0.25	457.24	0.33		0.25	1020.67	27.93				
$\lvert N \rvert = 15$	0.5	390.81	0.25	$\lvert N \rvert = 50$	0.5	799.41	26.53				
	0.75	372.68	0.08		0.75	757.08	25.53				
	Average	406.91	0.22		Average	859.05	26.67				

Table 6: Impact of ECVs on the solution costs under linear recharge for the medium- and large-sized instances

	$\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 0.75$
$\lvert N \rvert = 25$	94.38%	72.61%	59.69%
$\lvert N \rvert = 30$	91.95%	72.31%	56.76%
$\lvert N \rvert = 50$	90.09%	60.16%	44.93%
$\lvert N \rvert = 100$	82.84%	63.55%	44.58%

5.3 Non-linear recharge

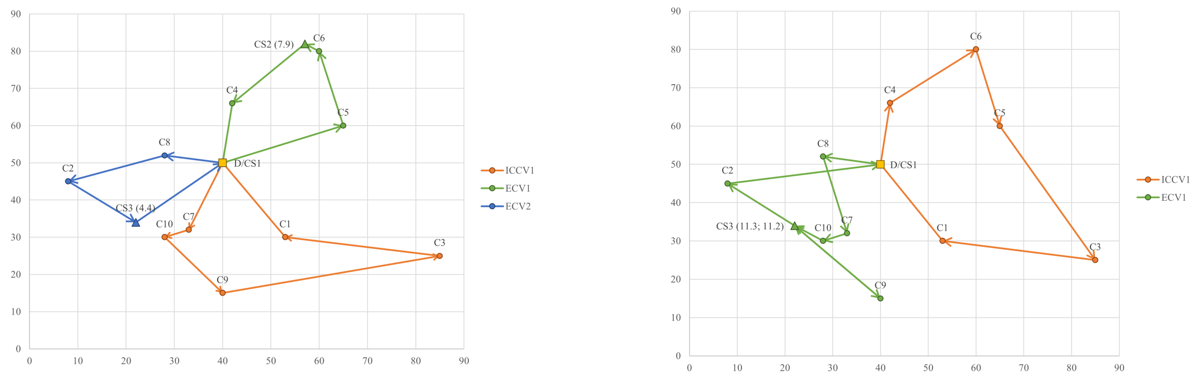
In this section, we present and analyse the results on all test instances under the assumption of nonlinear recharge. To evaluate the impact of the nonlinear recharging function compared to the linear counterpart, we transform the linear recharge solutions, i.e., the solutions obtained from performing the ILS-SP under the linear recharge assumption, as follows: For each test instance, we perform the `labeling` algorithm on the customer sequence of each ECV-route in the linear recharge solution, in which the algorithm considers the nonlinear charging function employed in this computational study. The new linear recharge solutions are considered in this section for comparison.

Table 7 shows the optimal costs and executing times for the small-sized instances. Again, the values in parentheses are the percentage of costs that the ILS procedure is able to reduce from the initialised solution (M_1), and that the SP procedure is able to improve from the ILS solution (M_2), respectively. Furthermore, each of the SP costs emphasised in bold indicates that it deviates from the SP cost of the corresponding linear recharge solution.

Table 7: Solution costs for small-sized instances under nonlinear recharge

	5 customers							**10 customers**							**15 customers**					
	$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$			$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$			$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$	
Instance	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**	**Instance**	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**	**Instance**	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**
C101C5	281.03	1	264.64	0	264.64	0	C101C10	446.6	1	424.01	0	**402.3**	0	C103C15	404.76	1	**389.53**	0	**367.32**	0
	(14.48%; 0%)		*(0%; 12.83%)*		*(12.17%; 4.73%)*			*(26.17%; 0%)*		*(9.86%; 4.36%)*		*(11.28%; 4.06%)*			*(27.56%; 15.29%)*		*(29.04%; 16.33%)*		*(38.78%; 15.87%)*	
C103C5	172.09	0	164.87	0	164.87	0	C104C10	367.91	0	355.4	0	**310.8**	0	C106C15	355.23	0	**342.82**	0	317.6	0
	(14.32%; 0%)		*(0%; 0%)*		*(0%; 0%)*			*(20.67%; 0%)*		*(12.29%; 1.54%)*		*(20.98%; 13.9%)*			*(39.48%; 0%)*		*(33.87%; 6.64%)*		*(25.48%; 18.59%)*	
R104C5	168.91	0	143.75	0	143.75	0	R102C10	281.36	0	284.18	0	253.06	0	R102C15	**424.41**	0	400.53	0	389.05	0
	(16.42%; 0%)		*(18.75%; 0%)*		*(18.75%; 0%)*			*(15.26%; 0%)*		*(22.23%; 0%)*		*(17.22%; 10.06%)*			*(20.31%; 2.44%)*		*(30.93%; 8.51%)*		*(29.54%; 2.42%)*	
R105C5	191.79	0	188.97	0	156.3	0	R103C10	216.15	0	201.47	0	198.61	0	R105C15	384.3	0	356.04	0	317.92	0
	(0%; 13.18%)		*(0%; 0%)*		*(0%; 15.5%)*			*(35.16%; 1.69%)*		*(3.41%; 8.56%)*		*(11.95%; 7.51%)*			*(19.6%; 0%)*		*(24.5%; 5.92%)*		*(20.69%; 12.22%)*	
RC105C5	257.63	0	254.54	0	250.58	0	RC102C10	421.05	0	384.75	0	384.75	0	RC103C15	452.39	0	**444.62**	0	**395.86**	0
	(3.92%; 0%)		*(1.76%; 0%)*		*(0%; 0%)*			*(6.46%; 0%)*		*(19.12%; 0%)*		*(19.12%; 0%)*			*(30.19%; 0.64%)*		*(33.54%; 2.75%)*		*(38.35%; 10.55%)*	
RC108C5	282.17	0	302.03	0	259.88	0	RC108C10	405.35	0	355.55	0	344.77	0	RC108C15	**537.21**	0	**400.39**	0	**405.73**	0
	(0%; 0%)		*(0%; 11.68%)*		*(0%; 0%)*			*(12.6%; 3.15%)*		*(10.19%; 12.52%)*		*(8.37%; 5.46%)*			*(29.03%; 13.11%)*		*(36.76%; 0%)*		*(27.05%; 20.53%)*	
C206C5	269.72	0	252.34	0	252.34	0	C202C10	299.64	0	297.58	0	245.24	0	C202C15	521.6	0	471.2	0	488.02	0
	(0%; 15.58%)		*(0%; 0%)*		*(0%; 0%)*			*(26.29%; 6.19%)*		*(27.27%; 3.83%)*		*(0%; 22.03%)*			*(28.8%; 4.15%)*		*(39.89%; 10.08%)*		*(26.59%; 4.56%)*	
C208C5	262.53	0	262.53	0	233.38	0	C205C10	367.91	0	326.43	0	**259.51**	0	C208C15	514.55	0	**313.19**	0	308.24	0
	(27.91%; 0%)		*(35.48%; 0%)*		*(34.54%; 0%)*			*(19.58%; 11.67%)*		*(17.75%; 21.78%)*		*(10.81%; 36.3%)*			*(20.72%; 10.73%)*		*(16.44%; 26.02%)*		*(21.51%; 22.48%)*	
R202C5	189.4	0	183.65	0	176.92	0	R201C10	251.4	0	**234.8**	0	224.31	0	R202C15	461.54	1	**392.69**	0	392.55	0
	(0%; 0%)		*(0%; 0%)*		*(18.16%; 0%)*			*(17.25%; 17.75%)*		*(17.84%; 1.14%)*		*(18.28%; 0%)*			*(35.54%; 2.78%)*		*(32.8%; 7.76%)*		*(31.1%; 6.24%)*	
R203C5	221.12	0	254.57	0	206.99	0	R203C10	289.74	0	277.48	0	259.88	0	R209C15	438.47	1	**366.9**	1	**316.62**	0
	(6.66%; 0.13%)		*(4.29%; 0%)*		*(21.74%; 0%)*			*(37.09%; 0%)*		*(20.4%; 12.3%)*		*(40.57%; 0%)*			*(30.01%; 2.33%)*		*(26.03%; 14.27%)*		*(43.49%; 0.08%)*	
RC204C5	251.8	0	251.8	0	212.57	0	RC201C10	356.91	0	335.2	0	315.34	0	RC202C15	508.38	0	450.98	0	433.13	0
	(1.44%; 0%)		*(23.21%; 0%)*		*(28.06%; 1.24%)*			*(10.58%; 19.86%)*		*(18.2%; 2.65%)*		*(7.59%; 18.93%)*			*(29.49%; 2.73%)*		*(31.65%; 7.99%)*		*(37.76%; 8.88%)*	
RC208C5	212.12	0	212.12	0	222.6	0	RC205C10	451.01	0	425.01	0	350.17	0	RC204C15	482.73	1	404.88	2	324.83	1
	(25.58%; 0%)		*(27.59%; 0%)*		*(5.39%; 0%)*			*(27.23%; 0%)*		*(23.42%; 8.82%)*		*(26.1%; 24.44%)*			*(29.76%; 2.83%)*		*(28.18%; 10.35%)*		*(30.34%; 28.72%)*	
Average	230.03	0.08	227.98	0	212.07	0	Average	346.25	0.08	**325.16**	0	**295.73**	0	Average	**457.13**	0.33	**394.48**	0.25	**371.41**	0.08
	(9.23%; 2.41%)		*(9.26%; 2.04%)*		*(11.57%; 1.79%)*			*(21.2%; 5.03%)*		*(16.83%; 6.46%)*		*(16.02%; 11.89%)*			*(28.37%; 4.75%)*		*(30.3%; 9.72%)*		*(30.89%; 12.6%)*	

For the instances with five customers, we find that the solutions, and therefore the costs, from both charging scenarios are the same. The nonlinear recharge solution costs start to deviate when we solve the instances with 10 and 15 customers. Figure 3a and Figure 3b illustrates the optimal routes found by the ILS-SP under the assumption of linear and nonlinear recharge, respectively, for instance C104C10 (0.75). At each CS in the ECV routes, the values in the parentheses indicate the amount of energy recharged in kWh. Our matheuristic found completely different optimal routes for each assumption of the charging function.



(a) Linear charging function (b) Nonlinear charging function

Figure 3: Illustration of the optimal routes found for C104C10 ($\alpha = 0.75$)

Table 8 shows the results obtained for the medium-sized instances. Here, it is clear that there are more bolded values, indicating that the ILS-SP finds different solutions under the nonlinear assumption more frequently for instances whose size are larger. The aforementioned finding suggests that nonlinear recharge functions provide a better approximation of real-world problems. In the case of 100 customers, in which the results are presented in Table 9, we find for all instances different solutions compared to that found under the linear recharge assumption. The advantage of the SP procedure remains noteworthy as it improves the solution cost for the majority of the instances. In the case of instance RC104C50 ($\alpha = 0.75$), the SP procedure decreases the ILS cost by 48.49%, which is also the best improvement among all results.

It is important to mention that we find six instances whose linear recharge solutions consists of an infeasible ECV-route, i.e., the `labeling` algorithm cannot construct any feasible s, t-path that visits all customers in the original one given the set of CSs in the instances. In particular, these instances include R102C50 (0.75), RC104C50 (0.25), R104C100 (0.5), RC102C100 (0.5), RC103C100 (0.75), and RC104C100 (0.5). Furthermore, these six instances all have 50 to 100 customers whose geographical distribution is (partially) random, while the linear recharge solution of each instance with clustered customer locations remains feasible for all cases. The aforementioned findings suggests that linear recharge functions might lead to infeasible solutions when problem instances are highly complex.

Table 8: Solution costs for medium-sized instances under nonlinear recharge

	25 customers							30 customers							50 customers					
	$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$			$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$			$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$	
Instance	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**	**Instance**	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**	**Instance**	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**
C101C25	**312.87**	6	321.24	4	**304.07**	4	C101C30	339.63	8	**340.18**	7	**308.51**	6	C101C50	584.67	20	481.08	19	**506.95**	18
	(36.03%; 12.29%)		*(21.73%; 6.8%)*		*(31.18%; 18.87%)*			*(22.57%; 8.03%)*		*(28.01%; 1.02%)*		*(32%; 18.71%)*			*(19.81%; 5.27%)*		*(25.32%; 10.26%)*		*(34.65%; 5.63%)*	
C102C25	300.99	7	**301.94**	7	**314.64**	7	C102C30	**336.9**	11	333.97	10	282.57	11	C102C50	**579.91**	35	**551.7**	34	**552.39**	33
	(43.86%; 14.8%)		*(38.49%; 22.04%)*		*(44.5%; 19.18%)*			*(59.37%; 1.26%)*		*(62.73%; 0%)*		*(64.3%; 2.32%)*			*(50.03%; 14.17%)*		*(54.57%; 11.33%)*		*(50.55%; 13.38%)*	
C103C25	**305.52**	9	296.95	10	297.23	10	C103C30	333.34	13	318.73	15	**294.76**	13	C103C50	**576.18**	44	**578.9**	41	**529.9**	39
	(40.37%; 4.75%)		*(43.81%; 7.36%)*		*(42.16%; 12.17%)*			*(55.8%; 3.55%)*		*(57.91%; 0.8%)*		*(53.59%; 16.2%)*			*(52.1%; 6.82%)*		*(43.62%; 9.96%)*		*(50.14%; 11.86%)*	
C104C25	**270.76**	12	264.95	12	**267.98**	11	C104C30	326.77	17	**285.07**	23	**286.42**	19	C104C50	**536.82**	55	**569.54**	52	**503.67**	65
	(35.93%; 13.57%)		*(31.07%; 14.98%)*		*(25.69%; 12.2%)*			*(31.2%; 10.22%)*		*(43.94%; 9.51%)*		*(38.38%; 10.59%)*			*(43.19%; 5.33%)*		*(44.51%; 1.76%)*		*(44.86%; 6.17%)*	
C105C25	306.13	6	291.83	5	291.62	6	C105C30	365.73	8	**324.34**	8	325.33	9	C105C50	**558.05**	26	**508.68**	25	**477.98**	26
	(32.85%; 0.15%)		*(22.82%; 15.33%)*		*(40.83%; 5.3%)*			*(37.61%; 0.11%)*		*(44.64%; 2.34%)*		*(37.32%; 0%)*			*(35.37%; 3.34%)*		*(35.81%; 1.89%)*		*(32.27%; 4.67%)*	
R101C25	**644.29**	4	**614.56**	4	**614.26**	4	R101C30	**723.7**	6	**692.16**	6	**684.47**	5	R101C50	**1140.4**	15	**1127.46**	14	**1066.3**	13
	(31.51%; 1.87%)		*(34.1%; 0.15%)*		*(33.32%; 0.2%)*			*(30.01%; 3.78%)*		*(31.38%; 5.04%)*		*(29.5%; 4.38%)*			*(35.86%; 3%)*		*(37.85%; 4.86%)*		*(38.06%; 7.15%)*	
R102C25	**562.02**	9	**548.14**	9	**533.39**	8	R102C30	**637.75**	13	**633.85**	13	**592.36**	12	R102C50	**1081.4**	29	**962.3**	30	**933.32**	27
	(33.34%; 1.09%)		*(32.99%; 3.2%)*		*(35.3%; 4.73%)*			*(32.78%; 1.8%)*		*(24.96%; 3.34%)*		*(26.64%; 5.3%)*			*(39.53%; 0.76%)*		*(42.33%; 5.71%)*		*(48.32%; 8.49%)*	
R103C25	520.83	12	**468.09**	12	**467.27**	12	R103C30	**539.02**	17	**531.98**	18	**522.19**	18	R103C50	**923.3**	46	**855.94**	47	**856.59**	52
	(22.96%; 2.55%)		*(29.16%; 10.61%)*		*(27.76%; 13.26%)*			*(15.41%; 11.86%)*		*(23.19%; 6.19%)*		*(19.7%; 9.3%)*			*(34.71%; 3.54%)*		*(39.77%; 6.18%)*		*(44.31%; 0.7%)*	
R104C25	**488.5**	14	**454.23**	12	**454**	11	R104C30	536.81	20	**491.16**	19	**482.89**	20	R104C50	860.28	59	**804**	56	**771.08**	50
	(18.02%; 9.81%)		*(23.33%; 11.97%)*		*(24.57%; 10.03%)*			*(15%; 13.53%)*		*(24.28%; 11.99%)*		*(26.17%; 8.15%)*			*(31.68%; 5.85%)*		*(29.03%; 11.82%)*		*(34.95%; 5.35%)*	
R105C25	**573.94**	5	**525.24**	4	**519.12**	4	R105C30	**614.75**	7	**594.63**	7	557.96	7	R105C50	**1079.12**	20	**1046.56**	22	**1013.22**	19
	(21.82%; 3.14%)		*(12.98%; 6.68%)*		*(15.95%; 6.53%)*			*(23.57%; 2.92%)*		*(21.57%; 1.43%)*		*(22.18%; 5.93%)*			*(26.62%; 4.47%)*		*(31.11%; 2.03%)*		*(29.66%; 0.67%)*	
RC101C25	**635.01**	3	**501.12**	2	**491.49**	2	RC101C30	**1231.92**	5	**799.52**	4	**787.14**	3	RC101C50	**1306.17**	14	**1021.61**	10	**983.29**	9
	(1.23%; 10.82%)		*(3.96%; 14.08%)*		*(0.82%; 17.12%)*			*(0.67%; 5.59%)*		*(12.02%; 3.37%)*		*(12.57%; 3.63%)*			*(33.64%; 13.35%)*		*(42.78%; 10.1%)*		*(42.05%; 10.92%)*	
RC102C25	596.6	5	469.31	3	**456.58**	5	RC102C30	**914.13**	7	**1068.97**	5	**588.63**	3	RC102C50	**1208.21**	22	**967.49**	16	**948.87**	13
	(30.94%; 3.81%)		*(31.17%; 20.26%)*		*(27.54%; 8.32%)*			*(17.38%; 23.79%)*		*(17.81%; 1.21%)*		*(51.69%; 5.96%)*			*(35.35%; 6%)*		*(35.98%; 6.51%)*		*(30.98%; 10.17%)*	
RC103C25	596.07	7	516.51	5	394.35	3	RC103C30	1054.97	7	759.26	5	626.66	4	RC103C50	**1010.53**	27	**921.29**	23	**775.31**	17
	(34.54%; 4.58%)		*(24.82%; 13.59%)*		*(16.21%; 17.81%)*			*(15.96%; 8.33%)*		*(11.69%; 29.21%)*		*(27.34%; 12.29%)*			*(26.75%; 39.67%)*		*(55.72%; 10.73%)*		*(50.14%; 17.08%)*	
RC104C25	586.91	7	509.16	5	**391.92**	4	RC104C30	1051.49	7	**880.25**	7	**727.3**	7	RC104C50	**1437.22**	28	**842.14**	29	**661.66**	20
	(28.44%; 1.85%)		*(31.75%; 3.33%)*		*(20.13%; 4.13%)*			*(11.68%; 8.32%)*		*(19.61%; 17.63%)*		*(13.21%; 27.38%)*			*(24.34%; 14.4%)*		*(44.12%; 32.6%)*		*(39.59%; 48.49%)*	
RC105C25	**713.42**	4	658.8	3	**504.44**	2	RC105C30	**1227.73**	5	1066.99	6	**1052.07**	4	RC105C50	**1103.19**	19	**931.23**	11	**828.45**	11
	(27.61%; 5.48%)		*(27.7%; 4.44%)*		*(14.83%; 11.16%)*			*(16.36%; 1.24%)*		*(9.43%; 7.91%)*		*(13.49%; 9.2%)*			*(20.36%; 40.85%)*		*(28.28%; 14.4%)*		*(39.33%; 18.95%)*	
Average	**494.26**	7.33	**449.47**	6.47	**420.16**	6.2	Average	**682.31**	10.07	**608.07**	10.2	**541.28**	9.4	Average	**932.36**	30.6	**811.33**	28.6	**760.6**	27.47
	(29.3%; 6.04%)		*(27.33%; 10.32%)*		*(26.72%; 10.73%)*			*(25.69%; 6.96%)*		*(28.88%; 6.73%)*		*(31.21%; 9.29%)*			*(33.96%; 11.12%)*		*(39.39%; 9.34%)*		*(40.66%; 11.31%)*	

Table 9: Solution costs for large-sized instances under nonlinear recharge

	$\alpha = 0.25$		$\alpha = 0.50$		$\alpha = 0.75$	
Instance	**Cost**	**Time**	**Cost**	**Time**	**Cost**	**Time**
C101C100	**1667.55** *(39.04%; 0.03%)*	115	**1443.88** *(47.65%; 8.9%)*	109	**1235.6** *(41.43%; 13.86%)*	98
C102C100	**1424.35** *(58.52%; 8.62%)*	164	**1273.53** *(53.31%; 13.4%)*	182	**1193.77** *(58.75%; 12.37%)*	152
C103C100	**1947.96** *(37.53%; 5.24%)*	212	**1454.56** *(50.49%; 6.87%)*	237	**1121.31** *(56.5%; 16%)*	200
C104C100	**1799.1** *(40.31%; 5.2%)*	321	**1141.7** *(49.44%; 15.36%)*	333	**1101.43** *(48.35%; 18.04%)*	321
C105C100	**1595.59** *(43.37%; 0.13%)*	149	**1488.89** *(43.73%; 10.21%)*	185	**1300.18** *(54.87%; 1.82%)*	170
R101C100	**1860.34** *(39.85%; 6.58%)*	58	**1857.54** *(36.58%; 6.13%)*	60	**1763.68** *(36.42%; 6.3%)*	47
R102C100	**1706.82** *(41.21%; 7.3%)*	110	**1623.09** *(44.12%; 7.24%)*	117	**1523.57** *(42.22%; 8.82%)*	101
R103C100	**1567.51** *(41.38%; 2.37%)*	202	**1457.64** *(40.46%; 7.09%)*	201	**1343.07** *(46.87%; 5.49%)*	206
R104C100	**1455.34** *(27.67%; 5.3%)*	273	**1226.1** *(26.62%; 7.21%)*	271	**1197.31** *(34.31%; 2.23%)*	233
R105C100	**1685.39** *(36.04%; 2.89%)*	87	**1615.98** *(37.88%; 2.7%)*	88	**1481.23** *(41.56%; 0.51%)*	86
RC101C100	**2353.38** *(40.05%; 8.19%)*	79	**1954.39** *(42.78%; 16.46%)*	74	**2159.38** *(42.37%; 3.9%)*	87
RC102C100	**2517.15** *(37.71%; 1.9%)*	123	**1877.35** *(35.6%; 21.5%)*	107	**1727.22** *(45.82%; 15.49%)*	96
RC103C100	**2392.22** *(34.82%; 3%)*	147	**1865.54** *(41.4%; 12.72%)*	162	**1646.64** *(38.93%; 20.08%)*	174
RC104C100	**2235.56** *(35.96%; 0.64%)*	192	**1955.96** *(37.78%; 5.72%)*	206	**1486.28** *(40.72%; 24.96%)*	196
RC105C100	**2507.08** *(36.15%; 3.12%)*	117	**1737.9** *(41.25%; 24.2%)*	129	**1630.57** *(55.36%; 2.62%)*	130
Average	**1914.36** *(39.31%; 4.03%)*	156.6	**1598.27** *(41.94%; 11.05%)*	164.07	**1460.75** *(45.63%; 10.17%)*	153.13

We also provide Table 10 which summarises average costs on every level of emissions restrictions for each class of instances under the nonlinear recharge scenario. Furthermore, the values in parentheses indicate the (absolute) deviation in percentage from the corresponding costs when we assume linear recharge.

Table 10: Summary for all classes of instances under nonlinear recharge

$\|N\|$	α	**Cost**	**Time**	$\|N\|$	α	**Cost**	**Time**	$\|N\|$	α	**Cost**	**Time**
	0.25	230.03 *(0%)*	0.08		0.25	494.26 *(2.06%)*	7.33		0.25	1914.36 *(11.49%)*	156.6
5	0.5	227.98 *(0%)*	0	25	0.5	449.47 *(1.6%)*	6.47	100	0.5	1598.27 *(5.4%)*	164.07
	0.75	212.07 *(0%)*	0		0.75	420.16 *(1.25%)*	6.2		0.75	1460.75 *(4.56%)*	153.13
	Average	223.36 *(0%)*	0.03		Average	454.63 *(1.64%)*	6.67		Average	1657.79 *(7.15%)*	157.93
	0.25	346.25 *(0%)*	0.08		0.25	682.31 *(2.18%)*	10.07				
10	0.5	325.15 *(0.01%)*	0	30	0.5	608.07 *(1.07%)*	10.2				
	0.75	295.73 *(0.99%)*	0		0.75	541.28 *(4.23%)*	9.4				
	Average	322.38 *(0.33%)*	0.03		Average	610.55 *(2.49%)*	9.89				
	0.25	457.13 *(0.23%)*	0.33		0.25	932.36 *(6.77%)*	30.6				
15	0.5	394.48 *(2.1%)*	0.25	50	0.5	811.33 *(3.27%)*	28.6				
	0.75	371.41 *(0.39%)*	0.08		0.75	760.60 *(3.7%)*	27.47				
	Average	407.67 *(0.91%)*	0.22		Average	834.76 *(4.58%)*	28.89				

On average, instances with larger size achieve higher deviation in costs, suggesting that the linear recharge assumption becomes less appropriate as an estimation for problem instances with high level of complexity. Furthermore, we conclude that the nonlinear recharge assumption does not exhibit disadvantage compared to the linear scenario in terms of executing time due to two reasons. First, the ILS-SP also obtains the solutions for large-sized instances within roughly five minutes under the nonlinear scenarios. Second, the executing time of the ILS-SP under the linear recharge assumption is only 10 to 20 seconds faster on average.

In addition, we analyse the impact of the CSs in the solutions found by the ILS-SP. Table 11 presents the proportion of the ECV-routes that visit at least one CS grouped by the level of emissions restriction and the size of the instances. For all cases, more than half of the ECV-routes in the optimal solutions require at least a visit to the CSs. This finding is consistent with that found in Montoya et al. (2017), in which they stated that allowing ECVs to recharge during their routes was crucial for optimising the ECV routing with nonlinear recharge.

Table 11: Proportion of the ECV-routes that visit at least one CS

	Number of customers						
α	**5**	**10**	**15**	**25**	**30**	**50**	**100**
0.25	90.48%	87.88%	82.50%	86.25%	81.44%	86.36%	84.02%
0.5	88.89%	76.19%	80.77%	79.63%	77.33%	85.23%	71.52%
0.75	84.62%	81.82%	75.00%	78.05%	75.00%	85.71%	66.97%

In Table 12, we show the proportion of the ECV-routes which is partially recharged among those that visit at least one CS. The proportion varies from 69.56% to 100%, emphasising the importance of partial recharge in our problem and also in practice. Partial recharge is beneficial in terms of costs since it avoids charging more than necessary, and in terms of customer time windows since the ECVs can be charged as much as needed to reach the targeted customers in time. It is therefore more flexible for the ILS-SP to route the customers on ECV-routes under partial recharge policy.

Table 12: Proportion of partially recharged ECV-routes among those that visit at least one CS

	Number of customers						
α	**5**	**10**	**15**	**25**	**30**	**50**	**100**
0.25	75.68%	84.00%	88.24%	95.40%	77.17%	90.45%	88.52%
0.5	71.88%	82.86%	94.44%	94.92%	79.27%	96.70%	92.31%
0.75	69.57%	80.00%	95.00%	100.00%	83.64%	98.28%	97.73%

Lastly, we analyse the maximum number of visits to a CS for each test instances. We find that this number varies from 0 (i.e., the ECVs in the solution does not visit to the CSs) to 11. This finding suggests that restricting the number of CS-visits to a certain number, which is a common practice in the VRP studies, might not be beneficial for finding the optimal routing. Therefore, our ILS-SP is superior in the sense that it allows as many visits to the same CS as needed, giving more flexibility to the matheuristic in routing the customers.

6 Conclusions

In this paper, we introduce a green mixed fleet vehicle routing problem with nonlinear partial recharge and time windows (GMFVRP-NLPRTW). We focus on the impact of nonlinear recharging functions, which refers to the nonlinear relationship between the amount of energy recharged and the state of charge of the vehicle's battery. Our problem accounts for the CO_2 emissions generated by conventional vehicles (ICCVs) by limiting the total amount of emissions to a certain value. We propose a matheuristic that combines an iterated local search and a set partitioning formulation (ILS-SP) to solve the aforementioned problem. In addition, we conduct a computational study with 288 test instances whose size can be classified into small, medium, and large. As our focus is on the nonlinear recharge functions, we execute the ILS-SP under two scenarios: linear recharge and nonlinear recharge.

We find that when the number of customers grows, the deviation of costs found under the nonlinear recharge assumption compared to that under the linear scenario becomes more frequent and exhibits greater variation. The linear solutions of six instances with 50 to 100 customers are also found to be infeasible under the nonlinear scenario. The aforementioned findings indicate that the more complex the problem instance is, the less appropriate the linear recharge assumption becomes. Furthermore, the solutions found under the linear recharge assumption remain feasible under the nonlinear scenario for 97.92% of our instances. A possible explanation is that the partial recharge policy helps avoid the violation of time windows constraints (the potential cause for infeasibility of the solutions with linear recharge). We conclude that nonlinear charging functions should be considered for problem instances with high level of complexity, as they provide a better approximation compared to the linear counterparts.

In addition, we analyse the routes travelled by an electric vehicle (ECV) under the nonlinear recharge assumption. We find that the partial recharge policy is beneficial in practice to avoid charging more than necessary, and it offers flexibility for the ILS-SP to route the customers. We also find that the optimal solution for a test instance can require up to 11 visits to the same charging station (CS), emphasizing a strength of our ILS-SP in which it does not limit the number of CS-visits to a certain value. Furthermore, we conclude that it is beneficial to solve the SP formulation as an improvement step after the ILS procedure. This is due to the fact that the SP procedure improves the costs found at the end of the ILS procedure for 84.72% of our instances in the nonlinear scenario, and it can achieve a cost reduction of up to 48.49%. Lastly, our ILS-SP is time-efficient in the sense that it achieves the solutions within a reasonable amount of time.

Potential future research includes developing a new method to solve the GMFVRP-NLPRTW to compare with our study. One might be interested in adding guidance for searching new neighborhoods and local optimum in the solution space, as our choice of perturbation and local search operator in each iteration is random. New recharge policies should also be considered, as we only let the ECVs recharge enough to reach the customers subsequent to the following node. Although our problem allows for multiple charging technologies, we do not analyse the effect of this characteristic in our computational study. Therefore, another research direction is to study the impact of employing multiple technologies. Lastly, we assume in this study a constant energy consumption rate for the ECVs, which depends only on the distance travelled. One might consider a more realistic model of energy consumption that involves vehicle load, vehicle speed, or traffic congestion.

References

Asghari, M., Al-e, S. M. J. M. et al. (2021). Green vehicle routing problem: A state-of-the-art review. *International Journal of Production Economics*, *231*, 107899.

Bektaş, T. & Laporte, G. (2011). The pollution-routing problem. *Transportation Research Part B: Methodological*, *45*(8), 1232–1250.

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.

Gonçalves, F., Cardoso, S. R., Relvas, S. & Barbosa-Póvoa, A. (2011). Optimization of a distribution network using electric vehicles: A vrp problem. In *Proceedings of the io2011-15 congresso da associação portuguesa de investigação operacional, coimbra, portugal* (Vol. 2011, pp. 18–20).

Jabali, O., Van Woensel, T. & De Kok, A. (2012). Analysis of travel times and co2 emissions in time-dependent vehicle routing. *Production and Operations Management*, *21*(6), 1060–1074.

Kramer, R., Subramanian, A., Vidal, T. & Lucídio dos Anjos, F. C. (2015). A matheuristic approach for the pollution-routing problem. *European Journal of Operational Research*, *243*(2), 523–539.

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

Macrina, G., Pugliese, L. D. P., Guerriero, F. & Laporte, G. (2019). The green mixed fleet vehicle routing problem with partial battery recharging and time windows. *Computers & Operations Research*, *101*, 183–199.

Moghdani, R., Salimifard, K., Demir, E. & Benyettou, A. (2021). The green vehicle routing problem: A systematic literature review. *Journal of Cleaner Production*, *279*, 123691.

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

gical, *103*, 87–110.

Sabet, S. & Farooq, B. (2022). Green vehicle routing problem: State of the art and future directions. *IEEE Access*.

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.

Tajik, N., Tavakkoli-Moghaddam, R., Vahdani, B. & Mousavi, S. M. (2014). A robust optimization approach for pollution routing problem with pickup and delivery under uncertainty. *Journal of Manufacturing Systems*, *33*(2), 277–286.

Úbeda, S., Faulin, J., Serrano, A. & Arcelus, F. J. (2014). Solving the green capacitated vehicle routing problem using a tabu search algorithm. *Lecture Notes in Management Science*, *6*(1), 141–149.

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).

Vidal, T., Crainic, T. G., Gendreau, M. & Prins, C. (2013). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research*, *231*(1), 1–21.

Ye, C., He, W. & Chen, H. (2022). Electric vehicle routing models and solution algorithms in logistics distribution: A systematic review. *Environmental Science and Pollution Research*, *29*(38), 57067–57090.

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 Pseudocode for the Insertion Heuristic for ICCV-routes

In Section 4.1.2, we describe the insertion heuristic for creating the set of ICCV-routes in the initial solution. Here, we present in Algorithm 4 the pseudocode for the aforementioned procedure.

Algorithm 4 The insertion heuristic framework for the ICCV-routes

```
procedure insertionHeuristicForICCV(C)
    Initialise the set of all ICCV-routes $\eta_C$
    stopCriteria ← false
    while stopCriteria is false do
        $r \leftarrow (s, i', t)$, where $i' \leftarrow \text{argmin}_{i \in C}\{t_i^l\}$
        if Adding $r$ to $\eta_C$ violates the emissions constraint then
            If there are unrouted customers in $C'$, then remove the customers that are not in $C'$ in the last
            ICCV-route until the emissions constraint is satisfied, then initialise a new ICCV-route with
            the unrouted customers in $C'$.
            Otherwise, stopCriteria ← true
        end if
        stopInserting ← false
        while stopInserting is false do
            $p_u^* \leftarrow \min_{p \in \{1,\dots m\} \cup \{t\}}\{c_1(i_{p-1}, u, i_p)\}\ \forall u \in C$
            $u^* \leftarrow \max_{u \in C}\{c_2(i(u), u, j(u))\}$
            if Inserting $u^*$ into the position $p_{u^*}^*$ of the current route is feasible then
                Insert $u^*$ and remove this customer from the set $C$
            else
                stopInserting ← true
            end if
        end while
        Add the current route $r$ to $\eta_C$
        if $C$ is empty then
            stopCriteria ← true
        end if
    end while
    return $\eta_C$
end procedure
```

B Perturbation operators

As mentioned in Section 4.3, the ILS-SP employs two additional perturbation operators, Random-Permute and Cyclic-Exchange, beside the three Relocate operators. In this section, we describe the mechanisms of the additional operators.

First, the Random-Permute is considered only if there are at least one ICCV- and one ECV-routes. For each type of vehicles, we randomly selected a route from the current solution. These two routes are then removed from the solution and destroyed, and their customers are stored in ascending order of their due time. We first initialise an ICCV-route that visits the customer with the earliest due time and is feasible with respect to the current solution. We then look for the cheapest feasible customer to insert into the position right after the current customer (i.e., in the end of the current route). The insertion of new customers terminates when the algorithm cannot find any feasible customer, after which the ECV-routes are created in the same manner. In the end, we add the new ICCV- and ECV-routes to the current solution and terminate the perturbation procedure.

Second, the Cyclic-Exchange considers two randomly selected routes, each can either be an ICCV- or an ECV-route. Let the the first route consists of k customers and the second route l customers. We consider exchanging two sequences of customers with lengths ranging from 1 to $\lfloor k/2 \rfloor$ and from 1 to $\lfloor l/2 \rfloor$ for route 1 and route 2, respectively. For each combination of sequences, we verify the feasibility of the new routes and the new solution cost they generate. We then choose the combination of sequences that generates the lowest cost and perturb the current solution.

C The Labeling algorithm

In this section, we present the pseudocode for our `labeling` algorithm, whose framework is shown in Algorithm 5. The algorithm takes in a sequence of customers to be visited (denoted by $(v_1, v_2, ...v_m)$), the starting and ending depots (denoted by v_0 and v_{m+1}, respectively), and the set R of all CSs.

Algorithm 5 The labeling algorithm

```
procedure LABELING($\{v_0, v_1, v_2, ..., v_m, v_{m+1}\}, R$)
    Initialise $L_{v_i} = \emptyset$, $\forall i \in \{0, 1, ...m+1\}$
    Add the initial label $l_{v_0} = (0, B^E, w^a, v_0, \texttt{null})$ to $L_{v_0}$
    for $i \in \{1, ...m+1\}$ do
        Consider going from customer $v_{i-1}$ to customer $v_i$
        for $l \in L_{v_{i-1}}$ do
            Create new labels $l_{v_i}$, whose predecessor is $l$, for customer $v_i$ based on the two scenarios
            Remove the invalid labels
            if Any new label $l_{v_i}$ dominates any existing labels in $L_{v_i}$ then
                Add $l_{v_i}$ to $L_{v_i}$ and remove the dominated labels from $L_{v_i}$
            end if
        end for
    end for
    return The $v_0, v_{m+1}$-path whose cost is the minimal
end procedure
```

New labels are created based on two scenarios: Either the ECV travels directly from a customer to another, or it visits a CS in between the customers. To illustrate the scenarios, we consider a sequence of customers $v_{i-1} \to v_i \to v_{i+1}$ belong to the input sequence of customers. Consider a label $l_{v_{i-1}} = (t^a_{v_{i-1}}, z_{v_{i-1}}, f_{v_{i-1}}, v_{i-1}, l) \in L_{v_{i-1}}$ of customer v_{i-1}. We create new labels for v_i as follows: In the first scenario, the ECV travels directly from v_{i-1} to v_i. Thus, we create new labels as shown in Algorithm 6. The label is invalid if the arrival time at v_i violates the due time of v_i, or the energy upon arrival at v_i drops below $10\%B^E$.

Algorithm 6 Creating new label: travel directly from i to j

```
procedure createNewLabel1($v_{i-1}, v_i, l_{v_{i-1}}$)
    $t^a_{v_i} \leftarrow \max\{e_{v_{i-1}}, t^a_{v_{i-1}}\} + s_{v_{i-1}} + t_{v_i v_j}$
    $z_{v_i} \leftarrow z_{v_{i-1}} - \pi d_{v_i v_j}$
    $f_{v_i} \leftarrow f_{v_{i-1}} + c^E_{v_i v_j} d_{v_i v_j}$
    if $t^a_{v_i} > b_{v_i}$ or $z_{v_i} < 0.1B^E$ then
        return null
    end if
    return New label $l_{v_i} = (t^a_{v_i}, z_{v_i}, f_{v_i}, j, l_{v_{i-1}})$
end procedure
```

In the second scenario, we consider the possibility of inserting a CS k in between v_{i-1} and v_i. Algorithm 7 shows the procedure of creating new labels for v_i under this scenario. To compute the maximum amount of recharge g_1, we take into account also the time windows of the next customer after v_i (i.e., customer v_{i+1}) such that with the amount of charge g_1, we can still reach the next customer in time. To compute the required amount of recharge g_2, we employ a policy where we only charge enough to reach the customer after v_i. If it is impossible, then we will recharge as much as we can (i.e., recharge an amount of g_1).

Algorithm 7 Creating new label: travel from i to j via a CS k

```
procedure createNewLabel2($v_{i-1}, v_i, k, l_{v_{i-1}}$)
    $t^a_k \leftarrow \max\{e_{v_{i-1}}, t^a_{v_{i-1}}\} + s_{v_{i-1}} + t_{v_i k}$
    $z_k \leftarrow z_{v_{i-1}} - \pi d_{v_i k}$
    if $t^a_k > b_k$ or $z_k < 0.1B^E$ then
        return null
    end if
    $g_1 \leftarrow$ compute the maximum amount of recharge
    $g_2 \leftarrow$ compute the required amount of recharge
    $g \leftarrow \min\{g_1, max\{g_2 - z_k, 0\}\}$
    $f_k \leftarrow f_{v_{i-1}} + c^E_{v_i k} d_{v_i k} + w_k g$
    $l_k \leftarrow (t^a_k, z_k, f_k, k, l_{v_{i-1}})$
    $t_c \leftarrow$ charging time corresponding to the amount of recharge $g$
    $t^a_{v_{v_{i-1}}} \leftarrow t^a_k + t_c + t_{k v_j}$
    $z_{v_{v_{i-1}}} \leftarrow z_k + g - \pi d_{k v_j}$
    $f_{v_{v_{i-1}}} \leftarrow f_k + c^E_{k v_j} d_{k v_j}$
    if $t^a_{v_i} > b_{v_i}$ or $z_{v_i} < 0.1B^E$ then
        return null
    end if
    return New label $l_{v_i} = (t^a_{v_i}, z_{v_i}, f_{v_i}, j, l_k)$
end procedure
```

D The Mixed Integer Program formulation for finding the maximum emissions

In this section, we provide the Mixed Integer Program (MIP) formulation for solving the G-VRP with time windows in which the fleet in consideration only consists of ICCVs. As we do not consider ECVs, we redefine the set of vertices $V = N \cup \{s, t\}$ consisting of the customers, the starting and the ending depots. The set of arcs A is also redefined such that it excludes all arcs that are incident to the CSs. Our decision variables include x_{ij}^C which takes value of 1 if the ICCV travels on arc $(i, j) \in A$ and 0 otherwise, u_i^C denoting the load remaining after serving customer i (kg), and τ_i denoting the arrival time at customer i (hour). The MIP is as follows:

$$
\begin{aligned}
\min \quad & \sum_{i\in V}\sum_{j\in V} c_{ij}^C d_{ij} x_{ij}^C && && \text{(11a)}\\
\text{s.t.} \quad & \sum_{j\in V} x_{ij}^C = 1, && \forall i \in N, && \text{(11b)}\\
& \sum_{j\in V\setminus\{s\}} x_{ij}^C - \sum_{j\in V\setminus\{t\}} x_{ji}^C = 0, && \forall i \in V, && \text{(11c)}\\
& \sum_{j\in V} x_{sj}^C \le n^C, && && \text{(11d)}\\
& \sum_{j\in V\setminus\{s\}} x_{si}^C - \sum_{j\in V\setminus\{t\}} x_{jt}^C = 0, && && \text{(11e)}\\
& u_i^C + q_j x_{ij}^C - Q^C(1 - x_{ij}^C) \le u_j^C, && \forall i \in V \setminus \{s,t\}, j \in V \setminus \{s\}, && \text{(11f)}\\
& u_s^C = Q^C, && && \text{(11g)}\\
& \tau_i^C + (t_{ij} + s_i)x_{ij}^C - T(1 - x_{ij}^C) \le \tau_j^C, && \forall i \in V, j \in V, && \text{(11h)}\\
& t_i^e \le \tau_i \le t_i^l, && \forall i \in V, && \text{(11i)}\\
& \sum_{i\in V}\sum_{j\in V} \varepsilon_{ij} d_{ij} x_{ij}^C \le UB, && && \text{(11j)}\\
& x_{ij}^C \in \{0,1\}, && \forall i \in V, j \in V, && \text{(11k)}\\
& u_i^C \ge 0, && \forall i \in V, && \text{(11l)}\\
& \tau_i \ge 0, && \forall i \in V. && \text{(11m)}
\end{aligned}
$$

The objective function (11a) is the total distance travelled by the ICCVs. Constraints (11b) ensure that each customer is visited by a single ICCV. Constraints (11c) present the flow conservation. Constraint (11d) ensures that the number of ICCVs employed does not exceed that available in the fleet. Constraint (11e) ensures that every route starts and ends at the depot. Constraints (11f) defines the decision variables u_i^C. Constrain (11g) sets the initial load before the ICCVs visit any customers. Constraints (11h) define the decision variables τ_i. Constraints (11i) ensure that the time windows of every customer are satisfied. Constraint (11j) restricts the total amount of emissions generated by the ICCVs to a certain level UB. Lastly, constraints (11k)-(11m) define the domains of the decision variables.

We solve the aforementioned MIP formulation using CPLEX Studio 22.1.0. with the gap being set as follows: For the small-sized instances, we set the gap to 0. For the medium- and large-sized instances, if the solving time exceed 10 minutes, then we set the gap to 0.2. This value will be increased to 0.4, 0.5, and then 0.6 each time the algorithm cannot find a solution within 10 minutes for the current gap. After the algorithm terminates, we compute the total emissions generated by the found solutions and set UB_{max} to this value.

E Programming Code

We implement our ILS-SP matheuristic in Java. The structure of our Java project consists of the following classes:

`Vertex`: An object of this class stores the relevant information about a vertex, namely type (depot, customer, or CS), ID, coordinates, demand, time windows, service time. If the vertex is the depot or a CS, then the corresponding object also stores the charging function (either linear or nonlinear) and the charging cost. The class `Vertex` stores the following variables: the predecessor vertex, the distance travelled up to this vertex, the load remaining after serving this vertex, the service start time, the energy level upon arrival, the energy recharged, and the emissions generated up to this vertex. The main methods of this class include setting a predecessor vertex, setting the amount of energy recharged, retrieving the emission factor, retrieving the maximum amount of energy that can be recharged at this vertex, and retrieving the charging time given a desired amount of charge.

`Instance`: An object of this class stores all the vertices belongs to a test instance, together with other information provided by the instance such as the load capacity, the battery capacity, and the vehicle speed. The main methods of this class include retrieving the set of customers N, the set of CSs R, and the set of all vertices V.

`ICCVRoute`: An object of this class represents an ICCV-route, which is stored as an `ArrayList` of objects belonging to the class `Vertex`. The parameter stored in this class is the upper bound on the emissions. Initialising an `ICCVRoute` object requires the starting and ending depot, and a customer. The main methods of this class include inserting a vertex/ a sequence of vertices, removing a vertex/ a sequence of vertices, retrieving a vertex given a certain position in the route, verifying the feasibility of the corresponding ICCV-route, and retrieving the total cost or emissions of the corresponding ICCV-route.

`ECVRoute`: An object of this class represents an ECV-route, which is stored as an `ArrayList` of objects belonging to the class `Vertex`. The parameter stored in this class is battery capacity of the corresponding ECV. Initialising an `ECVRoute` object requires the starting and ending depot, and a customer. The main methods of this class are those of the `ICCVRoute` class but with different mechanisms.

`Label`: An object of this class represents a label in the labeling algorithm. A `Label` object stores a vertex, the arrival time at the vertex, the energy level upon arrival at the vertex, the accumulated cost upon arrival at the vertex, and the predecessor label. The main method of this class is verifying if a given label is dominant to this label.

`ECVLabeling`: An object of this class represents the labeling algorithm on a graph. Initialising an `ECVLabeling` object requires the stating and ending depots, a sequence of customers to be visited, and the set of CSs R. The main method of this class is finding the best ECV-route (in the sense that it achieves the minimal cost), which return either the best found cost or ∞ if no feasible ECV-route can be found. The mechanism of the aforementioned method is described in Section 4.4 and Section C. We also implement for this class a method which retrieves the best ECV-route, if any (return `null` if no feasible ECV-route is found).

`ILSSP`: An object of this class represents the ILS-SP algorithm on a test instance. Initialising an `ILSSP` object requires a test instance, an upper bound UB on the amount of CO_2 emissions, and a value for the level of restriction α. Upon creating an `ILSSP` object, we perform the preliminary steps mentioned in Section 4.1 and adjust the value of UB if necessary. The main method of this class is running the ILS-SP on the input test instance, which is based on the four main procedures: `initialisation`, `localSearch`, `perturbation`, and `solveSP` which solves the SP formulation after the termination of the ILS procedure using CPLEX. The mechanism of each procedure is described in the previous sections. We also implement a method to return the solution found by the ILS-SP, whose output includes the objective cost, the total emissions generated by the ICCVs, the ICCV-routes, and the ECV-routes with the amount of energy recharged at the CSs, if any.