Submitted to *INFORMS Journal on Computing*
manuscript (Please, provide the manuscript number!)

The Electric Vehicle Routing and Overnight Charging Scheduling Problem on a Multigraph

Daniel Yamín
Centro para la Optimización y Probabilidad Aplicada (COPA), Departamento de Ingeniería Industrial, Universidad de los Andes, Bogotá, Colombia, d.yamin@uniandes.edu.co.

Guy Desaulniers
Polytechnique Montréal and GERAD, Montreal, Quebec, Canada, guy.desaulniers@gerad.ca.

Jorge E. Mendoza
HEC Montréal, Montreal, Quebec, Canada, jorge.mendoza@hec.ca.

The electric vehicle routing and overnight charging scheduling problem on a multigraph consists in routing a fleet of electric vehicles (EVs) that must serve a set of customers and scheduling their preceding overnight charging operations to minimize the total travel cost. The problem is defined on a multigraph capturing the alternative ways to travel between two locations and the trade-off between conflicting resources, such as the consumed energy and the traveled distance. The EVs are recharged using a limited number of chargers at the depot and according to a piecewise-linear recharging function. To solve the problem, we design a branch-price-and-cut algorithm that implements state-of-the-art techniques, including the ng-path relaxation, subset-row inequalities, and a specialized labeling algorithm. We report computational results showing that the method consistently solves to optimality instances with up to 50 customers. We also present experiments evaluating the impact of charging scheduling on computational time and multigraph representation on optimal cost.

Key words: electric vehicle routing, charging scheduling, multigraph, branch-price-and-cut, labeling algorithm, city logistics.

1. Introduction

In recent years, logistics companies that deal with goods distribution in urban city centers have progressively transitioned from internal combustion engine vehicles (ICEVs) to more-sustainable electric commercial vehicles (Pelletier et al. 2017). This transition is driven by the fact that electric vehicles (EVs) alleviate some of the unsustainable practices of present-day logistics. For instance, they produce minimal noise and no local greenhouse gas emissions, allowing delivery fleets to comply with noise restrictions and emission targets that are usually imposed in inner-city areas (Davis and Figliozzi 2013). Furthermore, using EVs enables companies to improve their market position by promoting a green brand image, an essential competitive factor due to the increasing environmental consciousness of consumers (Schneider et al. 2014). From a cost perspective, EVs are steadily becoming more attractive as a result of recent technological developments, spiking oil prices, and many incentives offered by local governments, including subsidies (Desaulniers et al. 2016).

In the urgency of replacing fossil fuels with renewable energy sources, EVs have become a realistic alternative to reduce the carbon footprint of city logistics operations. Despite a vast amount of capital investment from automakers and battery manufacturers, EVs still have significant disadvantages over ICEVs, such as a high acquisition cost, limited driving range, slow recharging times, and lack of comprehensive recharging infrastructure (Bektaş et al. 2019, Pelletier et al. 2016). While the acquisition cost is expected to decrease over the following years through economies of scale and technology innovations (Florio et al. 2021), the restrictive autonomy and recharging hassles need to be addressed before the wide adoption of EVs in urban logistics. Moreover, companies that currently use EVs in their daily operations need decision-support tools that account for these limitations.

To deal with EVs' limitations, researchers have investigated two broad approaches. The first one is to plan recharging stops along a route. This approach assumes that existing and widespread recharging infrastructure is in place. However, that is not the case in most urban centers, and due to the growing lack of urban spaces, an adequate charging infrastructure might not be possible soon (Florio et al. 2021). In addition, deliveries are typically carried out during the day when time-dependent energy costs are more expensive, en route charging can yield undesirable driver's idle times as well as cargo security concerns, public charging infrastructure is uncertain to be available, and the autonomy of newly available EVs is typically sufficient to perform delivery routes in urban areas (Pelletier et al. 2018, Kullman et al. 2021, Lebeau et al. 2016). For these reasons, operators usually prefer charging the fleet of vehicles at their facilities and overnight (Morganti and Browne 2018). Thus, the second approach that researchers have investigated, which consists in designing routes that can be completed with a full battery load, can be more realistic in the context of inner-city logistics. Nevertheless, this second approach raises another considerable challenge. Since a company would probably own a limited number of (expensive) chargers, the charging scheduling of EVs at their facilities can be difficult.

When planning EV routing and charging scheduling, their recharging process and energy consumption must be consistent with real-life behaviors. On the one hand, the recharging process defines the relationship between the charging time and the amount of energy recharged. To avoid overcharging the battery, the charging function of an EV comprises a linear and a non-linear component with respect to time (Marra et al. 2012). This non-linear behavior can be represented accurately by a piecewise-linear recharging function (Montoya et al. 2017). On the other hand, the energy consumption of EVs while in motion depends on several factors, including the road grade, ancillary systems, and rolling resistances (Bigazzi and Clifton 2015). In city logistics applications, the recurrent acceleration and deceleration caused by heavy traffic lead to significant energy losses (Asamer et al. 2016). This means that avoiding traffic is an excellent strategy to increase the driving range of an EV. In fact, longer routes that circumvent congested areas may be preferable from an energy consumption perspective, yielding a compromise between the consumed energy and the traveled distance.

In light of the discussion above, this paper introduces the multigraph-based electric vehicle routing and overnight charging scheduling problem with time windows (mE-VRSPTW), in which a fleet of EVs must be routed to serve a set of customers and their overnight charging operations scheduled so that the total traveled distance is minimized. In addition to the usual load and time window constraints, routes must comply with energy requirements. Furthermore, the EVs are recharged prior to performing their routes, using a limited number of chargers located at the depot. The recharging process of each EV occurs non-preemptively in a single charger and according to a piecewise-linear recharging function. The problem is defined on a multigraph, meaning that there are alternative arcs (i.e., paths on the road network) to travel between two locations. This representation captures the trade-off between the consumed energy and the traveled distance. Our contributions to the existing literature are the following:

1. In line with recent literature, we introduce an E-VRP in which the vehicles are recharged according to a piecewise-linear recharging function and using a limited number of chargers. Different from related studies, the EVs are charged at the depot rather than en route, and the problem is defined on a multigraph to capture the trade-off between conflicting resources. Routing and scheduling algorithms considering these real-world characteristics of EVs can facilitate their integration into goods distribution schemes.

2. To solve the problem, we design a branch-price-and-cut (BPC) algorithm that implements state-of-the-art techniques, including the *ng*-path relaxation, subset-row inequalities, and a specialized labeling algorithm. Moreover, we derive problem-specific dominance and branching rules.
3. In addition to the traditional set-partitioning constraints, the master problem handles the capacity constraints of the depot's chargers. Accordingly, we define the pricing problem on a network whose structure accounts for the dual variables from these capacity constraints and ensures that the charge of the EVs is conducted non-preemptively.
4. From a computational perspective, we show that the method consistently solves to optimality instances with up to 50 customers. We also evaluate the impact of charging scheduling on computational time and multigraph representation on optimal cost.

The remainder of this paper is organized as follows. Section 2 reviews related literature. Section 3 formally defines the mE-VRSPTW. Section 4 describes a BPC algorithm to solve this problem. Section 5 reports the computational results. Finally, Section 6 concludes the paper and outlines future research.

2. Related Literature

The limitations of EVs over traditional ICEVs have urged researchers to study variants of the vehicle routing problem (VRP, see Toth and Vigo (2014)) in which the vehicles have a limited driving range or can stop at charging stations to restore their batteries. Examples of such variants are VRPs with distance constraints, VRPs with intermediate replenishment facilities, and variable-route vehicle-refueling problems. Problems with these settings are usually framed as electric vehicle routing problems (E-VRPs). Several heuristic methods have been proposed to solve E-VRPs and related problems, e.g., Montoya et al. (2016), Erdoğan and Miller-Hooks (2012), Schneider et al. (2015). However, only a few exact methods have been developed to tackle these problems, probably because of the difficulty in handling the combined routing and scheduling operations. This section reviews the literature on exact methods to solve E-VRPs.

Desaulniers et al. (2016) proposed the first exact method for the E-VRP with time windows by considering four variants: single or multiple recharging stops and only full or partial recharges. In these variants, the recharging time is assumed to be a linear function of the amount of energy recharged. They presented a BPC algorithm in which the master problem is general-purpose and reinforced with valid inequalities. In contrast, the pricing problem is variant-specific, and it is solved by employing sophisticated resource extension functions and a dominance rule that handles time and charge resources. Later on, Desaulniers et al. (2020) improved this algorithm by fixing to zero some variables associated with routes traversing a sequence of two arcs.

Given that the battery charging process of an EV follows a non-linear function with respect to time, Montoya et al. (2017) examined this behavior and found that a piecewise-linear function can accurately represent it. They employed this approximation to tackle the E-VRP with a non-linear recharging function, multiple charging technologies, multiple recharging stops, and partial recharges. They formulated a charging station replication-based mixed-integer linear program (MILP) to solve the problem. However, such a formulation might lead to intractable models and introduce symmetry issues. Their computational experiments revealed that neglecting the non-linear behavior of the battery charging process could lead to infeasible or overly expensive solutions.

Froger et al. (2019) addressed the same problem as Montoya et al. (2017) and proposed two improved MILPs. The first is a charging station replication-based formulation, whereas the second is a path-based formulation. Their computational experiments demonstrated that these new formulations are solved faster by a commercial solver than the one by Montoya et al. (2017), primarily because they improve the linear programming relaxation bound.

Since charging stations usually have a fixed and small number of chargers, recent studies have included capacity constraints at these stations when optimizing routing decisions. Bruglieri et al. (2021) considered the green vehicle routing problem (G-VRP) with multiple refueling stops, only full refuels, and capacitated alternative fuel stations. In this work, it is assumed that when a vehicle visits a station, it refuels for a fixed amount of time and to its maximum capacity. They introduced a path-based MILP, which is solved by a cutting-plane algorithm that separates the fueling stations' capacity constraints when necessary.

Froger et al. (2022) tackled the E-VRP with a non-linear recharging function, multiple charging technologies, multiple recharging stops, partial recharges, and capacitated charging stations. In their problem, the EVs are recharged according to a piecewise-linear function. Similar to their previous work, they formulated a path-based MILP in which the chargers' capacity constraints are enforced using sequencing variables and flow variables and constraints. They solved the formulation via a commercial solver. Yet, they concluded that the latter is not an efficient approach mainly because the linear relaxation of the model is *weak*. Their experiments proved that the solution found when the capacity of charging stations is not explicitly modeled could be infeasible in practical applications with only a limited number of chargers.

Lam et al. (2022) investigated the E-VRP with time windows, a non-linear recharging function, multiple recharging stops, partial recharges, and capacitated charging stations. They proposed a BPC algorithm in which column generation (CG) solves the routing component and constraint programming solves the scheduling component. Iteratively, the method finds a set of optimal routes and checks whether they have a corresponding feasible charging schedule. When that is not the case, a logic-based Benders cut is derived, and that routing solution is eliminated from the search space. To *lift* the combinatorial Benders cuts, they devised a strategy based on conflict analysis. As pointed out by the authors, a drawback of their solution approach is that the master problem completely ignores the chargers' capacity constraints in the absence of Benders cuts.

All of the studies we have mentioned plan recharging stops along a route. However, many urban centers do not have a comprehensive recharging infrastructure, and operators usually prefer to charge EVs at their facilities (Pelletier et al. 2017). Still, the challenges that arise from the charging scheduling of EVs at a transport terminal or a central depot have received less attention than the routing component, even if its solution might help integrate EVs into goods distribution schemes. For this reason, some researchers have studied the EVs' depot charging scheduling problem.

In a pioneer work, Sassi and Oulamara (2014) investigated the problem of assigning a fleet of electric and conventional vehicles to a set of predetermined routes while maximizing EVs' usage and minimizing their charging costs. At the depot, the EVs charge in-between their assigned routes, and there are time-dependent energy costs and grid power capacities. Along the lines of Sassi and Oulamara (2014), Pelletier et al. (2018) addressed the EVs' depot charging scheduling problem by employing a more realistic model of the charging process to avoid overcharging the battery. They considered a multi-day planning horizon and incorporated battery degradation as well as facilities-related demand costs when determining the optimal charging schedule. Both Sassi and Oulamara (2014) and Pelletier et al. (2018) formulated MILPs to solve their problems.

Alvo et al. (2021) studied a bus dispatching problem within a public transport terminal working with a mixed fleet of electric and diesel buses and a restricted number of chargers. In this work, buses must be assigned to a trip schedule and a battery charge plan. At the transport terminal, electric buses are recharged according to a linear function. To solve the problem, they proposed a Benders' type decomposition in which a master problem assigns buses to trip schedules while a subproblem defines a battery charge plan for a given set of bus trip schedules. Zhou et al. (2022) also studied bus dispatch operations within a public transport terminal. They tackled a single public transport route's electric bus charging scheduling problem, considering a non-linear charging function as well as battery degradation costs. They formulated two MILPs that are enhanced using problem-specific valid inequalities.

Different from studies on E-VRPs, in those related to the EVs' depot charging scheduling problem, the vehicles are already preassigned to routes or must be assigned to a predefined set of routes. This means that the routing component is virtually absent in this stream of research. Moreover, exact methods to schedule EV charging at the depot primarily rely on mathematical formulations solved through commercial solvers — see Perumal et al. (2021).

VRPs have been traditionally defined over customer-based graphs. In such a graph, the vertices represent locations of interest (e.g., customers or depots), and the arcs represent the optimal path between those locations calculated according to a single criterion (e.g., traveling cost, distance, or time). Needless to say, in practice, there are usually several alternative paths when traveling between two locations. The latter is explicitly taken into account in the stream of research that studies VRPs on *road networks* — see (Ben Ticha et al. 2018). A road network is one that, in addition to the depot and customers' locations, also contains road junctions. Furthermore, a road network can be represented by a multigraph, in which the different arcs between every pair of vertices represent all non-dominated alternative paths to travel between them. Ben Ticha et al. (2017) conducted an empirical analysis for the VRPTW employing a multigraph representation of the road network. Using real-world instances, they found that this representation leads to methods with (unsurprisingly) higher computational times that also deliver better solutions.

The road network representation is particularly interesting in the context of EVs, where several resources compete, such as energy consumption, distance traveled, travel time, and load onboard. To the best of our knowledge, only Florio et al. (2021) have addressed an E-VRP on a road network using an exact solution method. They considered travel speed and energy consumption as time-dependent and stochastic. Accordingly, the energy requirements are treated probabilistically as chance constraints. They proposed a BPC algorithm in which CG is applied on the minimum distance graph, and a completion bound is used to *price* elementary routes efficiently. They did not address the EVs' charging scheduling. Instead, they studied the effects of traffic congestion on energy consumption.

3. Problem Statement

This section formally introduces the mE-VRSPTW. Section 3.1 presents the general definitions. Section 3.2 shows an example of a feasible solution to the problem. Finally, Section 3.3 describes an exponential-size mathematical formulation upon which we build our solution method.

3.1. General Definitions

Let $\mathcal{C} = \{1, 2, \ldots, C\}$ be a set of C customers that need to be served. We define the mE-VRSPTW on a directed multigraph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \{0\} \cup \mathcal{C}$ is the set of vertices, vertex 0 represents the depot, and $\mathcal{A} \subseteq \bigcup_{(i,j)\in\mathcal{V}^2} \mathcal{A}_{(i,j)}$ is the *multiset* of arcs connecting the vertices in $\mathcal{V}$. The set $\mathcal{A}_{(i,j)} = \{(i,j)^k \mid k = 1, \ldots, |\mathcal{A}_{(i,j)}|\}$ represents alternative ways to travel directly from $i \in \mathcal{V}$ to $j \in \mathcal{V}$.

A sufficiently large homogeneous fleet of EVs is available to serve the customers. Every vehicle is associated with a load capacity $Q \in \mathbb{Z}_+$ and is equipped with a battery storing an amount of energy between zero (a minimum amount) and a maximum amount $E \in \mathbb{Z}_+$. All the vehicles may depart from the depot as early as $\underline{T} \in \mathbb{Z}_+$ and must return to the depot no later than $\overline{T} \in \mathbb{Z}_+$, where $\underline{T} < \overline{T}$.

Every vertex $i \in \mathcal{V}$ is associated with a load $q_i \in \{0, 1, \ldots, Q\}$ that must be picked up, and a time window $[\underline{t}_i, \overline{t}_i]$ during which the vehicle visiting that vertex must start service, where $\underline{t}_i, \overline{t}_i \in [\underline{T}, \overline{T}]$ and $\underline{t}_i \leq \overline{t}_i$. The EVs can arrive at a customer location before the time window opening but must wait to begin service. For the depot, we set the load $q_0 = 0$, the time window opening $\underline{t}_0 = \underline{T}$, and the time window closing $\overline{t}_0 = \overline{T}$.

When traveling along an arc $(i,j)^k \in \mathcal{A}$, a vehicle incurs a driving distance $c_{(i,j)^k} \geq 0$, a travel time $t_{(i,j)^k} \geq 0$ (including the service time at vertex i if any), and an energy consumption $e_{(i,j)^k} \in \{0, 1, \ldots, E\}$. Considering that there are multiple ways to travel between two locations, $\underline{c}_{(i,j)}$, $\underline{t}_{(i,j)}$, and $\underline{e}_{(i,j)}$ denote the minimum distance, travel time, and energy consumption, respectively, when traveling from $i \in \mathcal{V}$ to $j \in \mathcal{V}, i \neq j$. We assume that these minimum values satisfy the triangle inequality.

In practice, the energy consumed by EVs primarily depends on the power required to accelerate and overcome rolling resistance, aerodynamic drag, and grade resistance (Davis and Figliozzi 2013). These factors are, in turn, heavily dependent on the vehicle's weight. Also, the vehicle's travel speed could be increased to fulfill time window requirements or reduced to consume less energy. Our problem-setting assumes that travel times and energy consumptions between locations are constant and given. This assumption is suitable to applications where the vehicle's load is nearly negligible compared to the curb weight (e.g., small-package or letter delivery and service vehicles) and the speed is exogenously given (e.g., city logistics). As pointed out by Goeke and Schneider (2015), fixed speed profiles and gradients can be integrated when computing energy consumption. However, modeling the energy consumption as a function of the total vehicle's weight or the speed of the vehicles as a decision variable substantially increases the complexity of the problem – see Desaulniers et al. (2016).

At the depot, there are $B \in \mathbb{Z}_+$ homogeneous chargers. Therefore, at most, B vehicles can be charging simultaneously. The chargers are privately operated, so there is no uncertainty about their availability. In addition, the charging process occurs before the routing of the vehicles. Specifically, a vehicle may start charging at time 1 – that is, $\underline{T}-1$ units of time prior to the depot's opening – and must stop charging before the starting time of its route. This means that a vehicle may delay its departure from the depot to have more time available to recharge. Accordingly, we define the set of discrete timesteps at which the vehicles may charge as $\mathcal{T} = \{1, \ldots, \underline{T}, \ldots, \widetilde{T}-1\}$, where $\widetilde{T} = \lfloor \max_{j \in \mathcal{C}} \{\bar{t}_j - \underline{t}_{(0,j)}\} \rfloor$ is the latest timestep at which a vehicle can depart from the depot to service a customer. As Figure 1 shows, the charging scheduling is a discrete-time problem in which decisions are made at the beginning of each period. In contrast, the routing is a continuous-time problem defined on the interval $[\underline{T}, \overline{T}]$.

Every vehicle initially (at time 1) has zero energy. This assumption is not restrictive as an energy level of zero does not mean that the battery is empty but rather represents a minimal state of charge. The battery is recharged according to a concave piecewise-linear function f with $P \in \mathbb{N}$ pieces, where $\mathcal{P} = \{1, \ldots, P\}$ is the *ordered* set of pieces. Each piece $p \in \mathcal{P}$ begins at an energy level $\phi_{p-1} \geq 0$ and ends at an energy level $\phi_p \geq 0$, satisfying

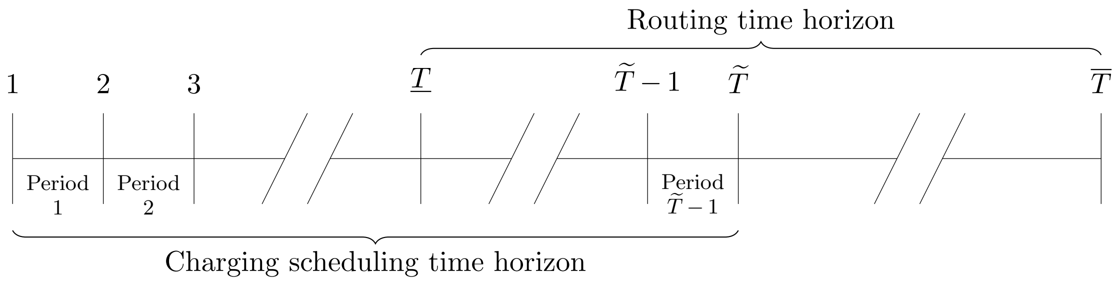


Figure 1 Routing and charging scheduling time horizons.

$0 = \phi_0 < \phi_1 < \ldots < \phi_P = E$. Associated with each piece $p \in \mathcal{P}$, there is a recharging rate $\theta_p > 0$ (measured in energy per timestep) such that $\theta_1 > \theta_2 > \ldots > \theta_P > 0$. Figure 2 presents the three-piece piecewise-linear function considered for the mE-VRSPTW. Given an energy level $e \in \{0, 1 \ldots, E\}$, the inverse recharging function $f^{-1}(e)$ maps to the recharge duration to achieve such a level (since the EVs initially have zero energy) and can be computed as

$$f^{-1}(e) = \sum_{p=1}^{p'} \frac{1}{\theta_p}\big(\min\{e, \phi_p\} - \phi_{p-1}\big), \tag{1}$$

where $p' \in \mathcal{P}$ denotes the piece in the piecewise-linear function that comprises the energy level e — that is, $\phi_{p'-1} \leq e \leq \phi_{p'}$. We can precompute function (1) given that the energy capacity and the energy consumption are assumed to be integer values. Once more, this assumption is not restrictive as any amount of energy can be approximated by a scaled integer value. Finally, EVs can charge their battery once and in a single charger. Figure 3 shows a charging schedule with two chargers, three EVs, and five timesteps that violates these two conditions. In this schedule, EV 1 (blue) preempts its recharge to share a charger, whereas EV 2 (green) moves to a different charger.

The mE-VRSPTW consists in routing the fleet of EVs and scheduling their preceding overnight charging such that: (1) each customer is visited exactly once by a single vehicle and within its time window, (2) each route starts and ends at the depot, (3) each route is load- and energy-feasible, (4) the number of EVs simultaneously charging at the depot is at most the number of available chargers, and (5) a vehicle receives no more than one continuous recharge (preemption is prohibited) and in a single charger. The objective of the mE-VRSPTW is to minimize the total distance traveled by the EVs. We summarize the notation for the problem statement in Table 2 in Appendix A.

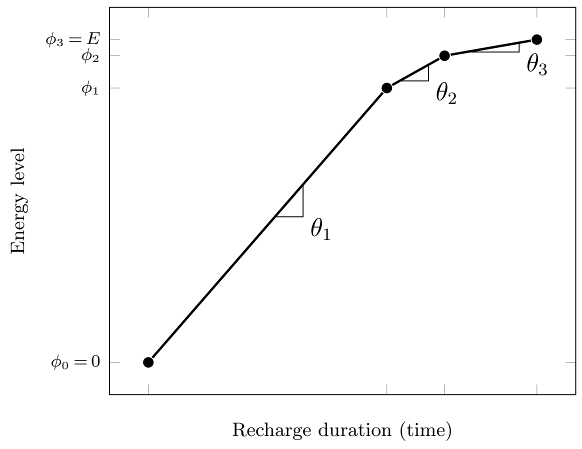


Figure 2 **The amount of energy charged as a function of the duration spent recharging according to Montoya et al. (2017).**

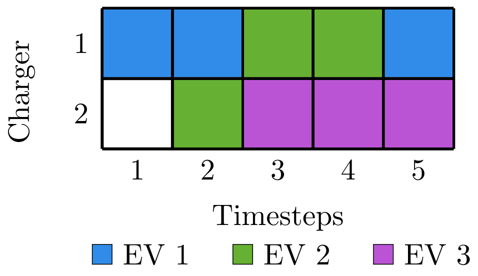
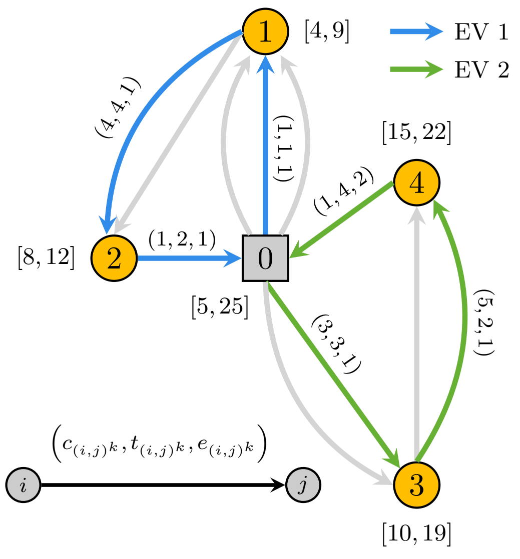


Figure 3 **Infeasible charging schedule. Adapted from Lam et al. (2022).**

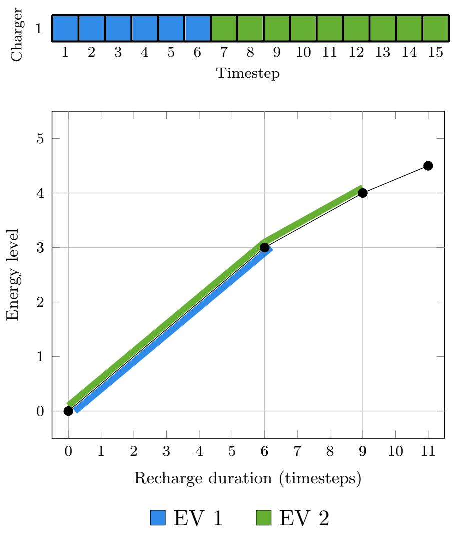
3.2. Illustrative Example

Figure 4 provides an example of a feasible solution for a small instance of the mE-VRSPTW. In this illustrative instance, there are $C = 4$ customers, and two EVs with a load capacity $Q = 10$ and equipped with a battery storing a maximum amount of energy $E = 5$. The load associated with each customer $i \in \{1, 2, 3, 4\}$ is $q_i = 5$. Figure 4(a) shows the time window associated with each vertex and the distance, travel time, and energy consumption associated with each used arc. There is $B = 1$ charger located at the depot and the recharging process occurs according to the piecewise-linear function depicted in Figure 4(b).

Figure 4(a) presents a feasible routing solution for the illustrative instance. EV 1 visits customers 1 and 2 (blue arcs), and EV 2 visits customers 3 and 4 (green arcs). The EVs do not use the gray arcs, but these show several ways to travel between two locations. By leaving the depot at times 7 and 16, respectively, EV 1 and EV 2 service their customers within their time window. Both EVs carry a load equal to 10, satisfying the load capacity. Figure 4(b) presents a feasible charging schedule. EV 1 charges during six timesteps, which translates to three units of energy. Subsequently, EV 2 charges during nine timesteps, which translates to four units of energy. Thus, both EVs have sufficient energy to complete their routes. Unlike most classical VRP variants, in the mE-VRSPTW, there are coupling constraints between routes because their corresponding vehicles interact at the depot during the charging process.



(a) Routing solution.



(b) Charging scheduling solution.

Figure 4 **Example of a feasible solution to the mE-VRSPTW.**

3.3. Set-Partitioning Formulation

The mathematical formulation of the mE-VRSPTW is a route-based set-partitioning model. A route is defined by the customers that it visits, the arcs that it traverses, and the timesteps during which it charges. For that purpose, let $a_i^r \in \mathbb{Z}_+$ be a constant representing the number of times that route r visits customer $i \in \mathcal{C}$, $a^r_{(i,j)^k} \in \mathbb{Z}_+$ be a constant representing the number of times that route r traverses arc $(i,j)^k \in \mathcal{A}$, and $b_t^r \in \{0,1\}$ be a constant indicating whether route r charges during timestep $t \in \mathcal{T}$. The cost c_r of a route r corresponds to its total distance traveled, that is,

$$c_r = \sum_{(i,j)^k \in \mathcal{A}} c_{(i,j)^k} a^r_{(i,j)^k}. \tag{2}$$

For a route r to be load-feasible, it must hold that

$$\sum_{i \in \mathcal{C}} q_i a_i^r \leq Q.$$

For a route to be feasible with respect to the time windows, it must arrive at its visiting vertices prior to their time window closing. Finally, for a route r to be energy-feasible, its charge must be conducted non-preemptively, and its energy level when departing the depot must be sufficient, namely,

$$f^{-1}\left(\sum_{(i,j)^k \in \mathcal{A}} e_{(i,j)^k} a^r_{(i,j)^k} \right) \leq \sum_{t \in \mathcal{T}} b_t^r. \tag{3}$$

Let $\mathcal{R}$ be the set of all possible routes satisfying load, time window, and energy constraints and $\lambda_r \in \{0,1\}$ be a variable indicating whether route $r \in \mathcal{R}$ is selected in the solution. A mathematical formulation for the mE-VRSPTW is given as follows:

$$\min \quad \sum_{r\in\mathcal{R}} c_r\lambda_r \tag{4a}$$

$$\text{s.t.} \quad \sum_{r\in\mathcal{R}} a_i^r\lambda_r = 1 \qquad \forall i \in \mathcal{C}, \tag{4b}$$

$$\sum_{r\in\mathcal{R}} b_t^r\lambda_r \le B \qquad \forall t \in \mathcal{T}, \tag{4c}$$

$$\lambda_r \in \{0,1\} \qquad \forall r \in \mathcal{R}. \tag{4d}$$

The objective (4a) minimizes the total cost of all chosen routes. Constraints (4b) guarantee that every customer is visited exactly once by a single EV. Constraints (4c) enforce the depot chargers' capacity. Finally, constraints (4d) establish the binary domain of the variables.

Since there is potentially a vast number of routes in $\mathcal{R}$, the model (4) is typically too large to be solved using an integer programming solver. Instead, we dynamically add variables and valid inequalities using a BPC algorithm. We refer to model (4) as the *integer master problem.*

4. Branch-Price-and-Cut Algorithm

This section presents a BPC algorithm to solve the integer master problem (4). A BPC algorithm is a branch-and-bound (BB) algorithm in which CG computes the lower bounds, and cutting-planes are incorporated to strengthen the linear relaxations in the search tree. A linear relaxation of the integer master problem is called a master problem (MP, see §4.1), and CG solves it. The CG algorithm iterates between a MP restricted to a subset of route variables, the so-called restricted master problem (RMP), and a pricing problem (PP). The RMP is a linear program that, given a subset of routes, determines the proportion of each route in a candidate solution. The role of the PP (see §4.2) is to find promising and feasible routes (including their charging schedules) to add to the RMP or prove that none exists. For this reason, the PP handles the intra-route constraints, namely, the load, time window, and energy requirements. Once the PP cannot find an improving route, the current RMP solution can be extended (by setting to zero all nongenerated variables) to yield an optimal solution to the MP. The cost of this optimal (and possibly fractional) solution is a lower bound on the optimal value. When the solution found is fractional, valid inequalities can be incorporated to strengthen the linear relaxation and obtain better lower bounds (see §4.3), or integrality can be enforced by branching (see §4.4). Otherwise, the routes (and charging schedules) selected by the MP form a feasible solution to model (4). We present the outline and setup of the BPC algorithm in Section 4.5 and summarize its notation in Table 3 in Appendix B.

4.1. The Master Problem

The MP at the root node of the BB search tree is the linear relaxation of model (4). Together, constraints (4b) and $\lambda_r \geq 0$ ensure that $0 \leq \lambda_r \leq 1$ for all $r \in \mathcal{R}$. During the BB search, valid inequalities and branching decisions are incorporated into the linear program.

The CG algorithm finds the optimal MP solution by iteratively solving the RMP, which is obtained by replacing the set $\mathcal{R}$ in the MP with a relatively small subset $\mathcal{R}' \subseteq \mathcal{R}$. Solving the RMP yields an optimal primal solution and a complementary dual solution $(\alpha_i)_{i\in\mathcal{C}}, (\beta_t)_{t\in\mathcal{T}}$, where $\alpha_i \in \mathbb{R}$ is the dual variable associated with constraint (4b) indexed by $i \in \mathcal{C}$ and $\beta_t \leq 0$ is the dual variable associated with constraint (4c) indexed by $t \in \mathcal{T}$. Extending this primal solution to the MP by setting $\lambda_r = 0$ for all $r \in \mathcal{R} \setminus \mathcal{R}'$ leads to an optimal MP solution if the reduced cost of every route $r \in \mathcal{R} \setminus \mathcal{R}'$ is nonnegative. Therefore, using the dual solution, the PP tries to find routes with a negative reduced cost. When such routes are found, they are added to the subset of routes $\mathcal{R}'$ in the RMP, and the CG algorithm iterates. Otherwise, the CG algorithm stops with an optimal MP solution.

4.2. The Pricing Problem

The PP identifies (or *prices*) routes to add to the RMP. For that purpose, it considers the routing of a single vehicle and handles the intra-route constraints. However, the dual variables $(\beta_t)_{t\in\mathcal{T}}$ of constraints (4c) provide information about the chargers capacity at the depot. Given a solution to the RMP, the PP consists in identifying routes with negative reduced cost, that is, routes $r \in \mathcal{R}$ such that

$$c_r - \sum_{i\in\mathcal{C}} \alpha_i a_i^r - \sum_{t\in\mathcal{T}} \beta_t b_t^r < 0. \tag{5}$$

To account for all feasible routes (and their corresponding charging schedules), the PP works on a directed multigraph $\mathcal{G}_\mathrm{P} = (\mathcal{V}_\mathrm{P}, \mathcal{A}_\mathrm{P})$. The set of vertices is defined as $\mathcal{V}_\mathrm{P} =$

$\{s\} \cup \mathcal{T} \cup \{\underline{0}\} \cup \mathcal{C} \cup \{\overline{0}\}$, where s is a dummy source vertex, there is a vertex representing each charging time $t \in \mathcal{T}$, $\underline{0}$ is the source depot vertex, there is a vertex for each customer $i \in \mathcal{C}$, and $\overline{0}$ is the sink depot vertex. The set of arcs is defined as $\mathcal{A}_{\mathrm{P}} = \mathcal{A} \cup \mathcal{A}_1 \cup \mathcal{A}_2 \cup \mathcal{A}_3$, where

- $\mathcal{A} = \{(\underline{0}, j)^k \mid j \in \mathcal{C}, k = 1, \ldots, |\mathcal{A}_{(\underline{0},j)}|\} \cup \{(i, \overline{0})^k \mid i \in \mathcal{C}, k = 1, \ldots, |\mathcal{A}_{(i,\overline{0})}|\} \cup \{(i,j)^k \in \mathcal{C} \times \mathcal{C} \mid i \neq j,\, q_i + q_j \leq Q,\, \underline{t}_i + t_{(i,j)^k} \leq \overline{t}_j,\, \underline{e}_{(\underline{0},i)} + e_{(i,j)^k} + \underline{e}_{(j,\overline{0})} \leq E,\, k = 1, \ldots, |\mathcal{A}_{(i,j)}|\}$ is the multiset of arcs connecting the vertices of $\mathcal{V} = \{\underline{0}\} \cup \mathcal{C} \cup \{\overline{0}\}$,
- $\mathcal{A}_1 = \{(s,t)^k \mid t \in \mathcal{T}, k = 1\}$ is the set of arcs connecting the dummy source with charging time vertices,
- $\mathcal{A}_2 = \{(t, t+1)^k \mid t \in \mathcal{T}, t < \widetilde{T}, k = 1\}$ is the set of arcs connecting successive charging time vertices, and
- $\mathcal{A}_3 = \{(t, \underline{0})^k \mid t \in \mathcal{T}, k = 1\}$ is the set of arcs connecting charging time vertices with the source depot vertex.

For $i \in \mathcal{V}_{\mathrm{P}} \setminus \mathcal{V}$, we set $q_i = 0$, $\underline{t}_i = \underline{T}$, and $\overline{t}_i = \overline{T}$. For $(i,j)^1 \in \mathcal{A}_{\mathrm{P}} \setminus \mathcal{A}$, we set $c_{(i,j)^1} = t_{(i,j)^1} = e_{(i,j)^1} = 0$. With these definitions, a modified cost $r_{(i,j)^k} \in \mathbb{R}$ can be assigned to each arc $(i,j)^k \in \mathcal{A}_{\mathrm{P}}$:

$$r_{(i,j)^k} = \begin{cases} c_{(i,j)^k} - \alpha_i, & (i,j)^k \in \mathcal{A}, \\ -\beta_i, & (i,j)^k \in \mathcal{A}_1 \cup \mathcal{A}_2 \cup \mathcal{A}_3, \end{cases} \tag{6}$$

where $\alpha_{\underline{0}} = 0$ and $\beta_s = 0$ for notation conciseness. Figure 5 shows the routes in the solution depicted in Figure 4 mapped to the corresponding multigraph $\mathcal{G}_{\mathrm{P}}$ of the PP. Note that some of the vertices in $\mathcal{V}_{\mathrm{P}}$ and arcs in $\mathcal{A}_{\mathrm{P}}$ are not depicted. Note, also, that a path from s to $\overline{0}$ passing through $\overline{0}$ defines a route (and its charging schedule).

The PP is an elementary shortest path problem with resource constraints (ESPPRC) from the dummy source s to the depot sink $\overline{0}$ in which the weight of the arcs is set to the modified cost. Given that the ESPRRC is NP-hard in the strong sense (Dror 1994), many researchers have relaxed (totally or partially) the elementarity requirements when solving the PP. One can relax the elementarity in the PP since constraints (4b) forbids integer solutions containing nonelementary routes. When the elementarity is totally relaxed, the resulting problem is the shortest path problem with resource constraints (SPPRC), in which any cycle might be part of a generated path.

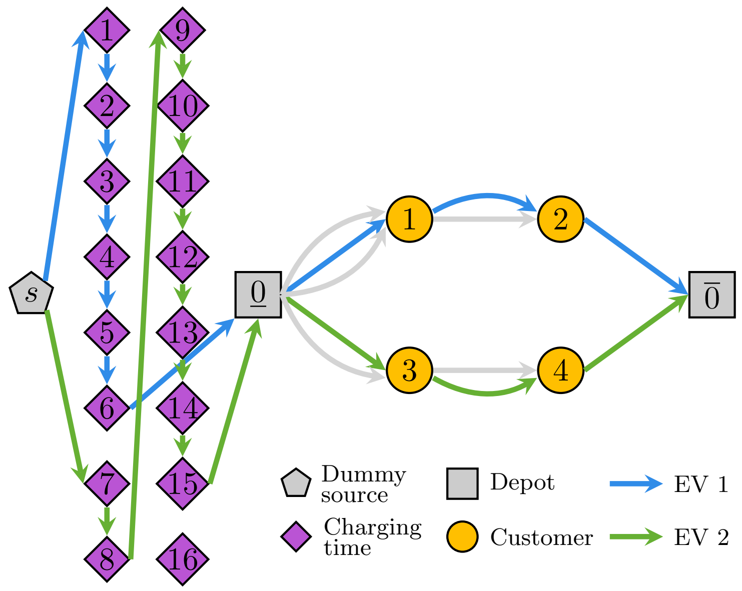


Figure 5 Illustrative pricing problem multigraph.

On the one hand, the SPPRC can be solved efficiently by a pseudopolynomial-time algorithm (Desrochers et al. 1992). On the other hand, solving the SPPRC as the PP may significantly reduce the quality of the lower bounds computed by the MP (Feillet et al. 2004). To balance between the computational demand and the quality of the lower bounds, we solve the *ng*-SPPRC as the PP (Baldacci et al. 2011).

Currently, state-of-the-art BPC algorithms for vehicle routing rely on the *ng*-path relaxation to solve their pricing procedure (Costa et al. 2019). In the *ng*-path relaxation, each customer $i \in \mathcal{C}$ has a neighborhood $\mathcal{N}_i \subseteq \mathcal{C}$ containing the $\Delta \in \mathbb{Z}_+$ closest customers to i (in terms of distance, time, or energy) and customer i itself. A *ng*-path allows a cycle starting and ending at vertex $j \in \mathcal{C}$ if and only if there exists a vertex $i \in \mathcal{C}$ in the cycle for which $j \notin \mathcal{N}_i$. Therefore, such a cycle is prohibited if and only if $j \in \mathcal{N}_i$ for every vertex $i \in \mathcal{C}$ it contains. Naturally, these cycling restrictions only apply to customer vertices.

In the *ng*-path relaxation, setting $\Delta = 0$ results in the SPPRC whereas setting $\Delta = C - 1$ results in the ESPPRC. Thus, even if the PP considers nonelementary routes, the so-called elementary bound can be reached by dynamically enlarging the neighborhoods. Along the lines of Contardo et al. (2015), we initialize the sets $\mathcal{N}_i, i \in \mathcal{C}$ with a small number of closest customers. Then, if the PP only finds nonelementary routes, the neighborhoods of the customers in a cycle are enlarged by adding the vertex where the cycle starts. We repeat this process until an elementary route with a negative reduced cost is found or a maximum neighborhood size $\overline{\Delta}$ is reached.

To solve the ng-SPPRC, we use a backward labeling algorithm — see Irnich and Desaulniers (2005). Our labeling algorithm performs a backward search since it determines the latest time the vehicle may depart from the depot (favoring more time to recharge) and the charging time required to perform the route. In turn, this information allows labels to build feasible charging schedules efficiently. Upon reaching the depot source vertex $\overline{0}$, labels know which of the charging time vertices they can visit (i.e., those prior to the route's departure time from the depot) and the minimum number of charging time vertices they need to visit (in order to have the necessary energy to perform the route). The algorithm starts with a subpath at the depot sink vertex $\overline{0}$ at time $\overline{T}$ and performs a backward search. At each iteration, the method extends an existing subpath through all the incoming arcs from its current vertex, thus building more subpaths. The algorithm evaluates all subpaths until at least one feasible complete path (from the dummy source s to the depot sink vertex $\overline{0}$ passing through the depot source vertex $\underline{0}$) with a negative reduced cost is found. Such a path corresponds to a feasible route (and charging schedule) that must be added to the set $\mathcal{R}'$ in the RMP.

Every subpath from an intermediate vertex $i \in \mathcal{V}_{\mathrm{P}}$ to the depot sink vertex $\overline{0}$ is encoded by a *label* ℓ_i that stores the reduced cost and resource consumptions of the subpath. Formally, a label ℓ_i is associated with a subpath starting at vertex i and stores the cumulative reduced cost $r(\ell_i) \in \mathbb{R}$, the remaining load capacity $q(\ell_i) \in \mathbb{Z}_+$, the latest time for starting service $t(\ell_i) \in [\underline{T}, \overline{T}]$, the remaining energy capacity $e(\ell_i) \in \mathbb{Z}_+$, the number of timesteps required to charge $b(\ell_i) \in \mathbb{Z}_+$, a resource $\mathbb{I}_c^u(\ell_i) \in \{0,1\}$ indicating whether customer $c \in \mathcal{C}$ is unreachable in the remainder of the subpath as a result of resource limitations, and a resource $\mathbb{I}_c^{ng}(\ell_i) \in \{0,1\}$ indicating whether visiting customer $c \in \mathcal{C}$ violates the ng-path cycling restrictions. When a subpath's resource consumption is infeasible or does not comply with the ng-path cycling restrictions, the corresponding label is discarded. To avoid enumerating all feasible routes, the labeling algorithm discards non-useful subpaths verifying dominance between labels.

To initialize the labeling algorithm, we create a label $\ell_{\overline{0}}$ representing the sink depot vertex with $r(\ell_{\overline{0}}) = 0$, $q(\ell_{\overline{0}}) = Q$, $t(\ell_{\overline{0}}) = \overline{T}$, $e(\ell_{\overline{0}}) = E$, $b(\ell_{\overline{0}}) = 0$, and $\mathbb{I}_c^u(\ell_{\overline{0}}) = \mathbb{I}_c^{ng}(\ell_{\overline{0}}) = 0$ for all $c \in \mathcal{C}$. At each iteration, the algorithm selects an unprocessed label ℓ_j and extends it backwardly to every incoming arc $(i,j)^k \in \mathcal{A}_{\mathrm{P}}$, obtaining new labels in the form of ℓ_i.

Algorithm 1 presents the label extension procedure. Lines 1, 2, and 3 check whether the extension is feasible. When the extension is to a customer vertex, Line 1 discards the label if that customer has been marked as unreachable or visiting it violates the *ng*-path cycling restrictions. The arc traversed from the charging time vertices to the depot source vertex determines the charging end time. Therefore, Line 2 discards the label when such time is either insufficient to charge the necessary energy or is later than the latest time for starting the route. Lastly, Line 3 verifies that once a label reaches the dummy source vertex, it has charged sufficient energy.

Lines 4-9 compute the resource consumptions of the label. Lines 4-7 update, respectively, the subpath's reduced cost, remaining load capacity, latest time for starting service, and remaining energy capacity. A unit time charge occurs when the extension is to a charging time vertex (Line 8). Otherwise, Line 9 computes the number of timesteps required to charge according to the consumed energy and the piecewise-linear recharging function. In Line 9, we round up the required charging time since we have a discrete-time scheduling problem while the inverse recharging function could map to a non-integer value. This rounding up ensures that the route is still energy-feasible – see constraint (3) – and leads to a more efficient dominance rule, as we will show later.

Once the resource consumptions have been computed, Line 10 checks whether the label extension is *actually* feasible. Specifically, when the latest time for starting service is earlier than the time window opening, the energy capacity has been exceeded, the necessary charge duration is greater than the remaining time available, or the time required to charge has already been completed, the label is discarded.

When the extension is to a customer vertex, Lines 11-17 mark customers that are unreachable in the remainder of the subpath by resource limitations or cycling restrictions. Lines 13 and 15 update the unreachable customers by using lower bounds on the travel time and energy consumption (their minimum values). Since we have multiple arcs to travel between customers, the lower bounds determine when a customer is certainly unreachable. However, a customer not marked as unreachable might be unreachable when traveling along a specific arc. Thus, once the label is extended, we need to check if the extension is actually feasible (as in Line 10). Afterward, Lines 16 and 17 identify the customers that cannot be visited due to the *ng*-path cycling restrictions. A customer can be visited only

Algorithm 1: Label extension.

```
Input: $\ell_j$, label representing a subpath starting at vertex $j$; $(i,j)^k \in \mathcal{A}_{\mathrm{P}}$, arc along which the extension is performed.
Output: $\ell_i$, label representing a subpath starting at vertex $i$.
/* check whether the extension is feasible */
1  if $i \in \mathcal{C} \wedge \big(\mathbb{I}^u_i(\ell_j) = 1 \vee \mathbb{I}^{ng}_i(\ell_j) = 1\big)$ then exit
2  if $i \in \mathcal{T} \wedge j = \underline{0} \wedge \big(i < b(\ell_j) \vee i \geq t(\ell_j)\big)$ then exit
3  if $i = s \wedge b(\ell_j) > 0$ then exit
/* update the resource consumptions */
4  $r(\ell_i) \leftarrow r(\ell_j) + r_{(i,j)^k}$
5  $q(\ell_i) \leftarrow q(\ell_j) - q_i$
6  $t(\ell_i) \leftarrow \min\{t(\ell_j) - t_{(i,j)^k}, \bar{t}_i\}$
7  $e(\ell_i) \leftarrow e(\ell_j) - e_{(i,j)^k}$
8  if $i \in \mathcal{T}$ then $b(\ell_i) \leftarrow b(\ell_j) - 1$
9  else $b(\ell_i) \leftarrow \lceil f^{-1}\big(E - e(\ell_i)\big)\rceil$
/* check whether the extension is *actually* feasible */
10 if $t(\ell_i) < \underline{t}_i \vee e(\ell_i) < 0 \vee b(\ell_i) \geq t(\ell_i) \vee b(\ell_i) < 0$ then exit
/* mark customers that are unreachable by resource or cycling constraints */
11 if $i \in \mathcal{C}$ then
12     for $c \in \mathcal{C}$ do
           /* unreachable customers */
13         $\mathbb{I}^u_c(\ell_i) \leftarrow \mathbb{I}^u_c(\ell_j)$
14         if $c \neq i \wedge \mathbb{I}^u_c(\ell_i) = 0 \wedge \big(q(\ell_i) - q_c < 0 \vee t(\ell_i) - \underline{t}_{(c,i)} < \underline{t}_c \vee e(\ell_i) - \underline{e}_{(c,i)} < 0\big)$ then
15             $\mathbb{I}^u_i(\ell_i) \leftarrow 1$
           /* ng-path cycling restrictions */
16         if $c = i \vee \big(\mathbb{I}^{ng}_c(\ell_j) = 1 \wedge c \in \mathcal{N}_i\big)$ then $\mathbb{I}^{ng}_c(\ell_i) \leftarrow 1$
17         else $\mathbb{I}^{ng}_c(\ell_i) \leftarrow 0$
18 return $\ell_i$
```

if it has not been visited before or does not belong to the neighborhood of a vertex visited after its last visit. Finally, Line 18 returns a new, extended label.

When a feasible label extension is found by Algorithm 1, the labeling method verifies dominance between labels associated with the same vertex to discard non-useful subpaths. On the one hand, if an existing label dominates the new label, the new label is discarded. On the other hand, if the new label dominates an existing label, the existing label is discarded. In the case that ℓ_i and ℓ'_i dominate each other, we keep one of them to guarantee the exactness of the method. In our labeling algorithm, the dominance rule depends on whether the label's vertex is a customer or a charging time. Definition 1 establishes the dominance rule when the label's vertex is a customer.

Definition 1. In a customer vertex $i \in \mathcal{C}$, a label ℓ_i dominates another label ℓ'_i if the following conditions hold:

$$r(\ell_i) \leq r(\ell'_i), \tag{7a}$$

$$q(\ell_i) \geq q(\ell'_i), \tag{7b}$$

$$t(\ell_i) \geq t(\ell'_i), \tag{7c}$$

$$e(\ell_i) \geq e(\ell'_i), \tag{7d}$$

$$\max\{\mathbb{I}^u_c(\ell_i), \mathbb{I}^{ng}_c(\ell_i)\} \leq \max\{\mathbb{I}^u_c(\ell'_i), \mathbb{I}^{ng}_c(\ell'_i)\} \quad \forall c \in \mathcal{C}. \tag{7e}$$

Conditions (7) imply that any feasible extension of the dominated label ℓ'_i is also a feasible extension of the dominant label ℓ_i with an equal or better reduced cost. Since the number of customers that cannot be visited because of the ng-path cycling restrictions is less than or equal to neighborhoods' size Δ, discarding labels is easier than in the labeling algorithm for the ESPPRC (the dominance condition is less restrictive). Definition 2 establishes the dominance rule when the label's current vertex is a charging time.

Definition 2. In a charging time vertex $i \in \mathcal{T}$, a label ℓ_i dominates another label ℓ'_i if the following conditions hold:

$$r(\ell_i) \leq r(\ell'_i), \tag{8a}$$

$$b(\ell_i) \leq b(\ell'_i). \tag{8b}$$

In a charging time vertex, dominance is verified only by the reduced cost and timesteps required to charge since the load, time, energy, unreachable customers, and cycling restrictions become irrelevant. Finally, condition (8b) is much more likely to be satisfied when the charging time is defined in terms of the number of timesteps required to charge (an integer value) rather than the charge duration as defined by the inverse recharging function (1) (a nonnegative value). In any case, as we have a discrete-time scheduling problem, EVs can only charge during the entirety of a timestep.

4.2.1. Heuristic Labeling. In the CG algorithm, solving exactly the PP is only necessary to prove the optimality of the current solution. Consequently, one or several heuristic algorithms can be invoked first, hoping to rapidly find a negative reduced cost route (Desaulniers et al. 2008). The exact labeling algorithm is invoked only when the heuristic algorithms fail to find a negative reduced cost route.

We derive a heuristic labeling algorithm based on the (exact) labeling algorithm explained before. This heuristic algorithm differs from the exact algorithm in two aspects. First, to alleviate the computational demand of handling a multigraph, for every pair of vertices $i, j \in \mathcal{V}$ such that $|\mathcal{A}_{(i,j)}| \geq 2$, we only keep the arc $(i,j)^k \in \mathcal{A}_{(i,j)}$ with the lowest cost $c_{(i,j)^k}$. Second, the heuristic labeling uses a simpler (and heuristic) dominance rule than stated in Definition 1 as it disregards condition (7e).

4.3. Valid Inequalities

Reinforcing the MP with valid inequalities significantly enhances the performance of branch-and-price algorithms (Desaulniers et al. 2011). The idea is to incorporate cutting-planes (*cuts*) to the linear relaxations encountered in the search tree to obtain tighter lower bounds and reduce the size of the search tree.

First, we strengthen the MP at the root node with the following well-known rounded capacity inequality:

$$\sum_{r \in \mathcal{R}} \lambda_r \geq \left\lceil \sum_{i \in \mathcal{C}} q_i / Q \right\rceil, \tag{9}$$

and we denote by $\pi \geq 0$ the associated dual variable. We further (dynamically) strengthen the MP with subset-row cuts (SRCs) defined over customer sets of size three (Jepsen et al. 2008). These constraints are rank-1 Chvátal-Gomory cuts derived from a subset of constraints (4b). Given a customer triplet $S \subseteq \mathcal{C}, |S| = 3$, the corresponding 3-SRC is defined as

$$\sum_{r \in \mathcal{R}} \left\lfloor \frac{1}{2} \sum_{i \in S} a_i^r \right\rfloor \lambda_r \leq 1, \tag{10}$$

and we denote by $\sigma_S \leq 0$ the associated dual variable. Constraint (10) specifies that, in a feasible integer solution, at most, one route may serve two or more customers in the customer triplet S. In the current form, cut (10) is *non-robust*, meaning that the corresponding dual variable may not be directly incorporated into the modified cost of the arcs, increasing the complexity of the PP.

Every time the CG algorithm terminates, SRCs are separated by enumerating all customer triplets and checking if the corresponding inequality is violated in the current solution. Considering that incorporating many SRCs into the MP results in a more difficult PP, we use three rules proposed by Desaulniers et al. (2008) to separate these cuts. First, at most, $\overline{S} \in \mathbb{N}$ cuts may be added simultaneously, prioritizing the ones with greater violations. Second, a customer may appear in, at most, $\overline{C} \in \mathbb{N}$ of the customer triplets that define these cuts. Third, cuts are only added if the most violated one has a violation greater than or equal to a threshold $\varepsilon \geq 0$. We denote by $\mathcal{S} \subseteq \{S \mid S \subseteq \mathcal{C}, |S| = 3\}$ the subset of customer triplets for which SRCs have been separated and added to the MP.

After incorporating to the MP the rounded capacity inequality (9) and SRCs (10) for the customer triplets in $\mathcal{S}$, the PP finds routes $r \in \mathcal{R}$ such that

$$\sum_{(i,j)^k \in \mathcal{A}_{\mathrm{P}}} r_{(i,j)^k} a^r_{(i,j)^k} - \sum_{S \in \mathcal{S}} \sigma_S \left\lfloor \frac{1}{2} \sum_{i \in S} a^r_i \right\rfloor < \pi, \tag{11}$$

where the second term of this reduced cost (left-hand side) can be interpreted as a *penalty* σ_S that must be paid every second visit to customers in $S \in \mathcal{S}$.

To account for the separated SRCs, we modify the labeling algorithm as in Jepsen et al. (2008), Desaulniers et al. (2008). For every customer triplet $S \in \mathcal{S}$, we add a new (unrestricted) resource $\eta_S(\ell_i) \in \{0,1\}$ that stores the number of times modulo 2 that the label ℓ_i has visited customers in S. When extending a label ℓ_j along arc $(i,j)^k \in \mathcal{A}$ to create a new label ℓ_i (see Algorithm 1), we set $\eta_S(\ell_i) \leftarrow \eta_S(\ell_j)$ if $i \notin S$ and $\eta_S(\ell_i) \leftarrow (\eta_S(\ell_j) + 1) \bmod 2$ otherwise. In the latter case, we subtract σ_S from $r(\ell_i)$ if $\eta_S(\ell_i) = 0$ and $\eta_S(\ell_j) = 1$.

The dominance rule must consider these new resources. To do so, condition (7a) in Definition 1 is replaced by

$$r(\ell_i) - \sum_{S \in \mathcal{S}_{\ell_i, \ell'_i}} \sigma_S \leq r(\ell'_i), \tag{12}$$

where $\mathcal{S}_{\ell_i,\ell'_i} \subseteq \mathcal{S}$ is the set of customer triplets $S \in \mathcal{S}$ for which $\eta_S(\ell_i) > \eta_S(\ell'_i)$. Condition (12) is necessary since, for $S \in \mathcal{S}$ such that $\eta_S(\ell_i) > \eta_S(\ell'_i)$, it may occur that a feasible extension of ℓ_i and ℓ'_i subtracts the dual variable $\sigma_S \leq 0$ from the reduced cost when extending ℓ_i but not when extending ℓ'_i. In other words, dominance can only occur if $r(\ell_i)$ is sufficiently less than $r(\ell'_i)$ to compensate for potential extra penalizations.

4.4. Branching Rules

Since the MP is a linear program, we use branching rules to enforce integrality whenever necessary. Our BPC algorithm branches hierarchically on the number of vehicles used, the arc-flow variables, and the consecutive flow through the depot source vertex.

If an integer number of vehicles is used, we branch on the arc-flow variables. Observe that any decision on an arc-flow variable $x_{(i,j)^k}, (i,j)^k \in \mathcal{A}_{\mathrm{P}}$ (i.e., the two-index formulation commonly used in VRPs) has an equivalent representation in terms of the route variables $\lambda_r, r \in \mathcal{R}$. In this branching strategy, we first branch on the arc-flow variables $x_{(i,j)^k}, (i,j)^k \in \mathcal{A}$ (the arcs between customers or a customer and the depot) by choosing the one with a flow closest to 0.5. Then, two constraints, one in each child node, are (locally) imposed in the PP: $x_{(i,j)^k} = 0$ (meaning that the arc must *not* be used in the solution), and $x_{(i,j)^k} = 1$ (meaning that the arc must be used in the solution). To implement the former, we remove from $\mathcal{R}'$ all routes using arc $(i,j)^k$ and the arc $(i,j)^k$ itself from $\mathcal{A}_{\mathrm{P}}$ to forbid generating a route traversing that arc in the PP. To implement the latter, we remove from $\mathcal{R}'$ all routes using arcs $(i,j)^l, l \neq k$ – those parallel to $(i,j)^k$ – , arcs $(i,u)^l, u \neq j, l = 1, \ldots, |\mathcal{A}_{(i,u)}|$ if $i \in \mathcal{C}$, or arcs $(u,j)^l, u \neq i, l = 1, \ldots, |\mathcal{A}_{(u,j)}|$ if $j \in \mathcal{C}$. We also remove these arcs from $\mathcal{A}_{\mathrm{P}}$, forcing that, in the PP, vertex i is always visited immediately after vertex j using arc $(i,j)^k$.

When the arc-flow variables $x_{(i,j)^k}, (i,j)^k \in \mathcal{A}$ have all integer values, we branch on the arc-flow variable $x_{(i,j)^1}, (i,j)^1 \in \mathcal{A}_1 \cup \mathcal{A}_3$ for which the fractional part of its flow value $\widetilde{B} \notin \mathbb{Z}_+$ is closest to 0.5. The branching is performed by (globally) imposing the following constraints in the MP (one in each child node): $x_{(i,j)^1} \leq \lfloor \widetilde{B} \rfloor$ and $x_{(i,j)^1} \geq \lceil \widetilde{B} \rceil$. Then, the corresponding dual variable is subtracted in the modified cost of the arc $(i,j)^1 \in \mathcal{A}_1 \cup \mathcal{A}_3$.

We remark on two technical details. On the one hand, by flow conservation, when the flow through arcs in $\mathcal{A}, \mathcal{A}_1$, and $\mathcal{A}_3$ is an integer, the flow throughout the entire network is also an integer. For this reason, branching on arcs in $\mathcal{A}_2$ is unnecessary. On the other hand, when branching on arcs in $\mathcal{A}_1$, the dual variable may penalize or reward the labels' reduced cost, depending on their charging start time. Thus, for Definition 2 to remain valid when such branching has been imposed, labels need to be extended even after they have completed their charge.

Despite guaranteeing integer arc flows, the current branching decisions do not necessarily guarantee integer path flows. When none of the above branching decisions can be imposed, then there exist two arcs, $(\underline{0}, j)^k \in \mathcal{A}$ and $(t, \underline{0})^1 \in \mathcal{A}_3$, such that their consecutive flow is fractional. We branch on these arcs as follows. On the first branch, we forbid using arc $(t, \underline{0})^1$ immediately after arc $(\underline{0}, j)^k$. On the second branch, if the arc $(\underline{0}, j)^k$ is used, it must be immediately followed by the arc $(t, \underline{0})^1$ — that is, we forbid the use of all arcs $(t', \underline{0})^1 \in \mathcal{A}_3 \setminus \{(t, \underline{0})^1\}$. These decisions are (locally) imposed in the PP and require modifying the labeling algorithm. For the first decision, a label $\ell_{\underline{0}}$ that used arc $(\underline{0}, j)^k \in \mathcal{A}$ cannot be extended along arc $(t, \underline{0})^1$ and, for the second decision, such a label can only be extended along this arc. For both decisions, $\ell_{\underline{0}}$ can only dominate labels that have also reached vertex $\underline{0}$ through arc $(\underline{0}, j)^k$.

This last branching rule on the consecutive flow through the depot source vertex ensures that our algorithm is exact. However, it was unnecessary for our computational experiments. In these, branching on the number of vehicles used and arc-flow variables was sufficient to obtain integer path flows. Note that this idea of branching on consecutive flow has been used in other vehicle routing problems, e.g., Desaulniers (2010).

4.5. Outline and Setup

Concisely, the outline of the BPC algorithm is the following:

Step 1. Choose an unprocessed node in the BB tree. If the lower bound is greater than or equal to the (global) upper bound, prune the node.

Step 2. Solve the RMP. If it is infeasible, prune the node and go back to *Step 1.*

Step 3. Solve the PP using the labeling algorithm. If routes with negative reduced cost are found, add them to the RMP and go back to *Step 2.*

Step 4. Separate SRCs and add them to the MP. If at least one cut is separated, go back to *Step 2.*

Step 5. If the MP solution is fractional, mark the node as processed, branch, add the two child nodes to the set of unprocessed nodes in the BB tree, and go back to *Step 1.* If the MP solution is integer, update the upper bound (if possible) and prune the node.

The BB tree is explored using a best-bound search strategy — in Step 1, we choose the unprocessed node with the lowest lower bound. In Step 2, the RMP might be infeasible due to branching decisions. In Step 3, we allow a maximum of 400 routes to be generated. Additionally, we first solve the PP using the heuristic labeling algorithm. We only invoke the exact labeling algorithm if the heuristic labeling fails to find a negative reduced cost route. For the ng-path relaxation, we consider an initial neighborhood size of $\Delta = 7$ and a maximum neighborhood size of $\overline{\Delta} = 12$. These values usually yield a good compromise between the quality of the lower bounds and computational time (Desaulniers et al. 2019). To separate the SRCs in Step 4, we set $\overline{S} = 30$, $\overline{C} = 5$, and $\varepsilon = 0.1$. Finally, in Step 5, we use hierarchical branching rules as explained in §4.4.

5. Computational Results

This section reports the computational results of our BPC algorithm solving the mE-VRSPTW. The algorithm was implemented in Java using the Branch-and-Price framework of the Java OR library (`jORLib`) and the Java graph theory library (`JGraphT`) by Michail et al. (2020). We relied on the ILOG CPLEX solver (version 22.1) for solving the RMPs. All experiments were executed on a standard PC with an Intel Core i7-8665U CPU @ 1.90GHz with 16GB of RAM allocated to the memory heap size and a two-hour computational time limit.

5.1. Test Instances

To perform computational experiments, we construct a set of instances based on the well-known VRPTW benchmark of Solomon (1987). These instances have a 4-parameter naming convention `DTm-n`. Parameter `D` is the geographical distribution of the customers; it can be either R (random), C (clustered), or RC (mix of random and clustered). Parameter `T` is the tightness of the time windows; it can be either 1 or 2, where instances of type 1 have tighter time windows than instances of type 2. Parameter `m` is the (two-digit) instance number, and parameter `n` is the number of customers. The `n`-customer instances are derived from the 100-customer instances by considering only the first `n` customers. For each instance, the load capacity Q of the vehicles, and the load q_i, time window $[\underline{t}_i, \overline{t}_i]$, and service time associated with each vertex $i \in \mathcal{V}$ are given. These instances are well-suited for city logistics problems as customers are distributed over a 100×100 km^2 area, representing a semi-urban operation.

We design a set of benchmark instances for the mE-VRSPTW as follows. We use minutes (min) as the time unit of discretization (i.e., each period in Figure 1 lasts one minute) and kilometers (km) as the distance unit. Furthermore, we assume the EVs travel at a fixed speed of 60 km/h. As a result, one unit of distance (1 km) is traversed in one unit of time (1 min). Despite the traveling speed being slightly higher than most cities' average speed, this assumption guarantees that the time windows of Solomon's instances remain valid.

Similar to Montoya et al. (2017), the EVs in our instances are `Peugeot iOns`. This EV has an energy capacity E of 16 kilowatt-hours (kWh). To capture the trade-off between conflicting resources, we consider two alternative ways to travel between each pair of customers or a customer and the depot: (1) the minimum driving distance (and travel time) alternative $(i,j)^1 \in \mathcal{A}$ and (2) the minimum energy consumption alternative $(i,j)^2 \in \mathcal{A}$.

In the former, the driving distance $c_{(i,j)^1}$ in km is the *euclidean distance* and the energy consumption $e_{(i,j)^1} = c_{(i,j)^1} \times 0.175$ kWh/km given these EVs' average consumption rate (Dataset 2022). In the latter, the driving distance $c_{(i,j)^2}$ in km is equal to the *Manhattan distance* and the energy consumption $e_{(i,j)^2} = c_{(i,j)^2} \times 0.125$ kWh/km. This last consumption rate is the one used by Montoya et al. (2017). Moreover, different consumption rates are possible as these depend primarily on the grade resistance and recurrent acceleration and deceleration caused by heavy traffic, which, in turn, depend on the selected route. With these settings, one alternative may dominate the other. Naturally, we only consider the dominant alternative in such a case. Table 1 presents the average number of alternative ways to travel between two locations for each group of instances.

The EVs are charged with a slow charger (11 kW), often used in practice. Consequently, the three-piece ($P = 3$) piecewise-linear recharging function is defined by $\theta_1 = 10.79$, $\theta_2 = 5.71$, and $\theta_3 = 1.60$ kWh/h and $\phi_1 = 13.6$, $\phi_2 = 15.2$, and $\phi_3 = 16.0$ kWh (Montoya et al. 2017). Recall that EVs may start charging $\underline{T} - 1$ units of time before the depot's opening. We choose the value of $\underline{T}$ so that a full charge can be conducted before the depot's opening time. In addition, Table 1 shows the number of chargers, B, for each instance. We choose B as the minimum number of chargers such that all the instances in the same group (e.g., R1 with 25 customers) remain feasible and solvable within the time limit. As we show later, these values yield difficult charging scheduling problems.

Our solution method assumes that the energy capacity and consumption are integer values. This assumption is needed to precompute the inverse of the recharging function (1) for each energy level $e \in \{0, 1, \ldots, E\}$. Thus, we use decawatt-hours (dWh) as a unit of energy, which is equivalent to working with kWh and considering two decimal points (e.g., $E = 1,600$ dWh rather than $E = 16$ kWh). Also, we compute the driving distance $c_{(i,j)^k}, (i,j)^k \in \mathcal{A}$ as in Kohl et al. (1999); we obtain integer values by truncating the non-negative distances with one decimal point. While our BPC algorithm does not require this truncation, it helps with the results' reproducibility. We compute the travel time $t_{(i,j)^k}, (i,j)^k \in \mathcal{A}$ as the sum of the corresponding (integer) driving distance and the service time at vertex i.

As explained in Appendix C, we set certain energy-related parameters slightly differently in the RC 50-customer instances due to its distinctive customers' locations. Furthermore, we set the arcs' minimum distance, travel time, and energy consumption by running all-pairs shortest paths algorithms to ensure that the triangle inequality holds for these values. Finally, all instances are publicly available at www.vrp-rep.org (Mendoza et al. 2014).

5.2. Performance of the BPC Algorithm

Table 1 summarizes the BPC algorithm's performance by presenting average values for each group of instances. Columns 5 and 6 show the lower bound and number of SRCs separated at the root node. Column 7 shows the number of nodes in the BB tree. Column 8 shows the total computational time in seconds. Columns 9 and 10 show the proportion of time spent on the master and pricing problems. These proportions do not add to 100% since they do not include some steps in the BPC algorithm, such as node selection and branching. Columns 11 and 12 show the number of EVs used and the objective value of the best integer solution found. Columns 13–15 provide information about the charging schedule. The *time in use* is the percentage of the time between the first and last timesteps in which an EV is charging that at least one charger is used. The *chargers in use* is the proportion of chargers used at the depot in that time frame. Finally, the *time at capacity* is the proportion of the time in use that all chargers are used simultaneously. Tables 5 and 6 in Appendix D present the (disaggregated) results for each instance.

Instance	C	Avg. # of altern.	B	LB	Cuts	Nodes	Time			Vehicles used	UB	Time in use (%)	Chargers in use (%)	Time at capacity (%)
							Total (s)	Master (%)	Pricing (%)					
R1		1.64	3	487.4	13.3	3.2	1.73	12.17	71.89	6.0	488.3	94.92	72.96	49.60
C1	25	1.61	1	198.2	10.6	1.7	11.01	3.47	92.30	3.2	198.6	75.27	75.27	75.27
RC1		1.53	3	445.4	59.0	2.5	4.44	11.81	78.34	5.0	446.6	100.00	75.08	47.40
R2		1.66	1	431.6	10.4	6.3	2.87	8.77	63.80	4.8	431.6	74.31	74.31	74.31
C2	25	1.62	1	272.5	43.1	1.5	57.50	2.78	94.21	3.0	272.5	37.93	37.93	37.93
RC2		1.53	1	419.4	39.8	7.3	4.20	16.35	63.42	4.6	419.4	87.19	87.19	87.19
R1		1.63	6	853.2	92.6	21.0	201.26	7.40	87.68	10.0	857.6	100.00	72.87	40.43
C1	50	1.68	2	370.3	21.2	1.4	53.03	3.65	93.53	5.2	371.0	92.08	76.85	61.62
RC1		1.48	4	798.3	108.5	2.3	559.96	2.67	96.42	7.6	800.1	99.59	74.48	54.10
R2		1.65	2	753.9	78.4	15.5	129.21	14.33	76.85	8.1	755.0	86.79	70.75	54.70
C2	50	1.68	1	482.5	80.5	2.0	221.91	6.67	91.08	5.4	482.5	34.21	34.21	34.21
RC2		1.49	2	699.9	33.8	2.5	26.10	8.93	84.14	6.4	699.9	87.72	66.90	46.09

Table 1 Summary of the BPC algorithm's performance.

The lower bounds at the root node are tight, with a maximum integrality gap of 1.86%. This result is mainly due to the large number of separated SRCs, which have demonstrated efficacy in vehicle routing problems. The resulting gaps allow proving optimality by exploring relatively small BB trees. Still, enforcing integrality in the mE-VRSPTW can be harder than in other vehicle routing problems, given that an integer flow value of the arcs between customers does not guarantee an integer solution to this problem. For example, all the branching decisions imposed in instance RC201-25 are over the arcs in $\mathcal{A}_1 \cup \mathcal{A}_3$.

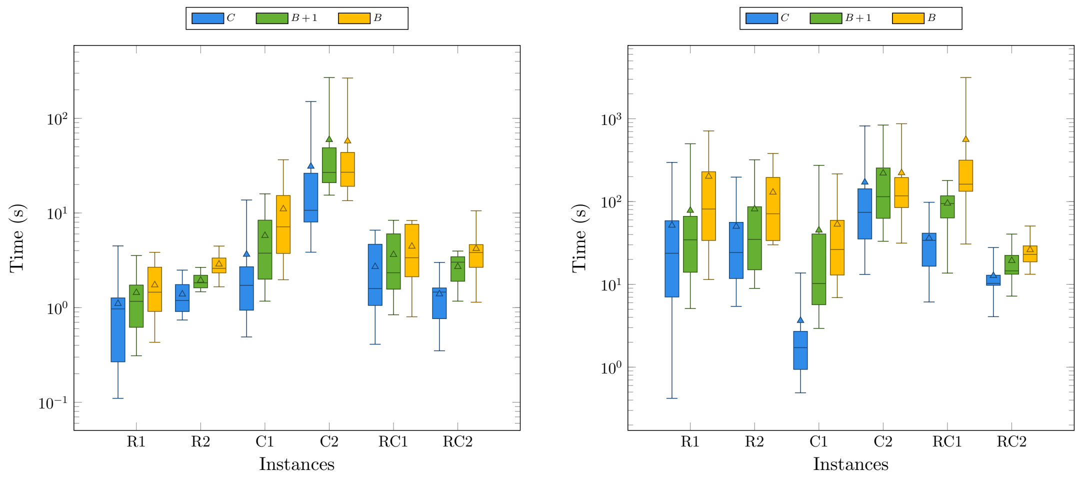
The average running time of the BPC algorithm is 12.15 and 192.46 seconds for 25- and 50-customer instances, respectively. In turn, these values are 129.78 and 72.79 seconds for the type 1 and 2 instances. Unsurprisingly, the running time grows exponentially with the number of customers. While the type 2 instances are usually harder than those of type 1 as the search space of the pricing problem is larger, in this case, solving some type 1 instances can be very time-consuming as the narrow time windows can have a meaningful impact on charging decisions. This can be corroborated by the geometric mean of the BPC's running time for type 1 and 2 instances: 15.72 and 20.18 seconds, respectively. Indeed, the geometric mean avoids being overly sensitive to single, large entries. Overall, the algorithm spends most of the time running the labeling algorithm to find new routes. However, linear programs can have many columns and rows in specific test cases, significantly increasing the proportion of time the algorithm spends on the RMPs.

Regarding the charging schedule, the type 1 instances' average time in use, chargers in use, and time at capacity are 94%, 74%, and 54%, whereas, for the type 2 instances, these average values are 69%, 63%, and 57%. These results corroborate that complying with customers' narrow time windows can complicate the charging operations as the resulting schedule is usually tighter (noting that, as a result of the wide time windows, many type 2 instances have only one charger). On average, 6.36 EVs are used in the type 1 instances, compared to 5.50 in the type 2 instances, resulting in a more congested charging station. In general terms, the average time in use, chargers in use, and time at capacity are 82%, 68%, and 55%, proving that the charging schedules are tight and the station often works at full capacity. These results imply that the instances' setup is consistent with real-life scenarios where planning the charging operations is complex due to the limited number of chargers.

In brief, the BPC algorithm can efficiently solve instances with up to 50 customers. The good performance of the algorithm can be attributed to the fact that it uses some of the state-of-the-art techniques for vehicle routing BPC-based algorithms: the *ng*-path relaxation, subset-row inequalities, and a specialized labeling algorithm.

5.3. Impact of Charging Scheduling on Computational Time

Figure 6 presents the box plots of the computational time when solving the problem with C, $B+1$, and B chargers. The box plots indicate the minimum, maximum, median, and quartiles of the computational time (in seconds) for each group of instances. The triangles in the box plots indicate the mean. Note that the y-axis is on a logarithmic scale.



(a) 25-customer instances. (b) 50-customer instances.

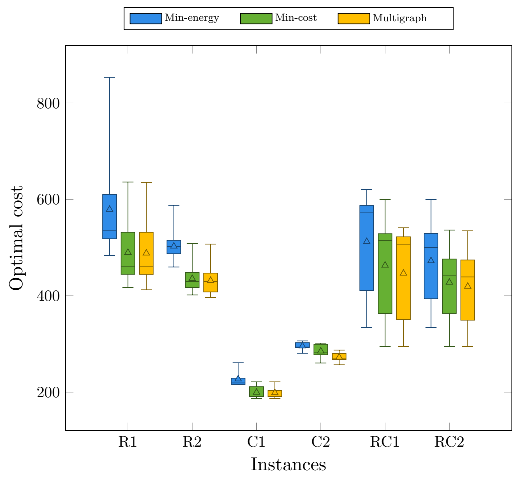
Figure 6 Impact of charging scheduling on computational time.

The computational time increases exponentially when the charging scheduling becomes harder (or the charging schedule tighter). In fact, the problem without charging scheduling (C chargers) is solved almost three times faster than with B chargers. In turn, the problem with $B+1$ chargers is solved over one and a half times faster than when the charging scheduling is harder (B chargers). These results corroborate that the mE-VRSPTW is a very challenging problem from a theoretical and computational perspective due to its combined routing and scheduling structure. They also imply that EV charging scheduling can be difficult at companies' facilities with limited chargers. Thus, it must be addressed when planning EV routing.

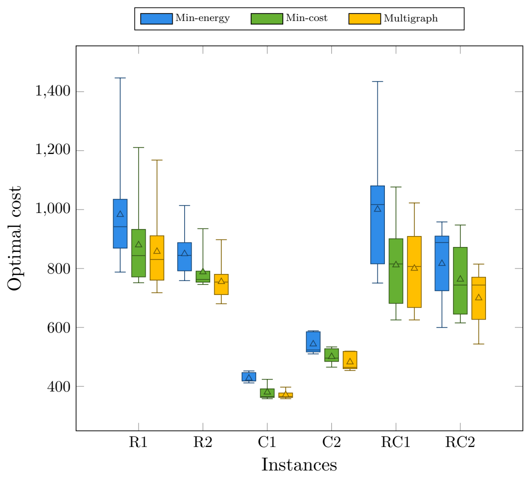
5.4. Impact of Multigraph Representation on Optimal Cost

This section compares the solution obtained on the multigraph to the solution obtained when tackling the problem on a simple graph, either with only the minimum energy consumption alternatives (min-energy graph) or the minimum driving distance alternatives (min-cost graph). Figure 7 shows the box plots of the optimal cost found by the BPC on the min-energy graph, min-cost graph, and multigraph.

Despite all instances being feasible, except for one on the min-cost graph, the long traveling times in the min-energy graph and the high energy consumption rate in the min-cost graph means that more EVs are necessary to service the customers. Interestingly, the multigraph representation improves the solution cost for 112 and 76 (out of 112) instances compared to the solution on the min-energy and min-cost graphs. This improvement (or saving) is, on average, 12.96% for the min-energy graph and 2.87% for the min-cost graph. It reaches up to 28.72%, highlighting the importance of the multigraph representation as routing models can find better solutions by having alternatives to balance conflicting resources. Yet, this improvement in solution quality is at the expense of higher computational times. The problem on the min-energy and min-cost graphs is solved 2.26 and 1.73 times faster than on the multigraph.



(a) 25-customer instances.



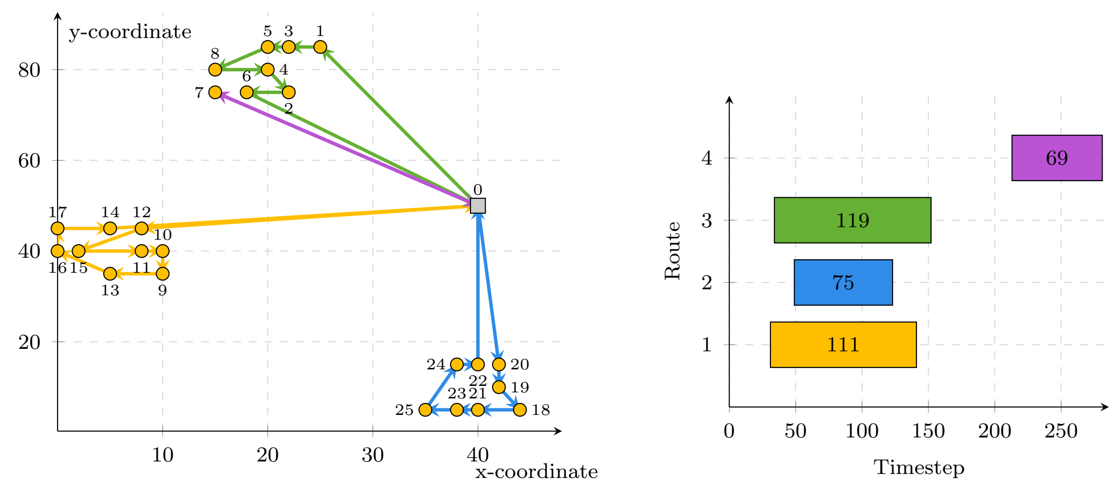
(b) 50-customer instances.

Figure 7 Impact of multigraph representation on optimal cost.

To better understand the savings that the multigraph representation achieves, Figures 8, 9, and 10 show, respectively, the optimal solutions to the RC104-25 instance on the min-energy graph, min-cost graph, and multigraph.

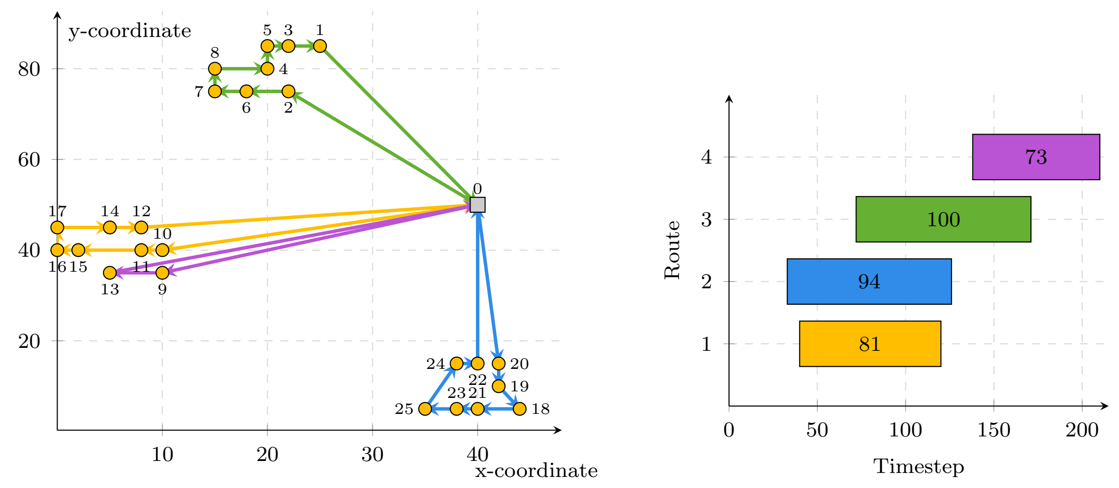
The sequence of customers visited by Route 1 (yellow) is different on every graph. On the min-cost graph, customers 9–17 cannot be served by one EV due to the high energy consumption. While this route's sequence seems odd on the min-energy graph and multigraph, it is due to the setup of the instances. Indeed, when two locations are on the same x-

or y-coordinate, the minimum energy alternative dominates the minimum distance alternative. Moving on, the sequence of customers visited by Route 2 (blue) is the same for every graph. Naturally, this route consumes less energy on the min-energy graph but travels more distance. Finally, the sequence of customers in Route 3 (green) is different on the min-energy graph. The long traveling times on this latter mean it is impossible to traverse the same sequence as in the multigraph or include customer 7 in the current route.



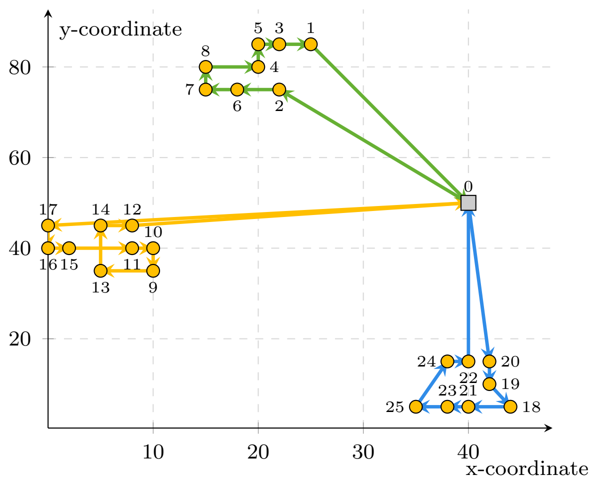
(a) Routing solution. (b) Charging schedule.

Figure 8 RC104-25 optimal solution on the min-energy graph.

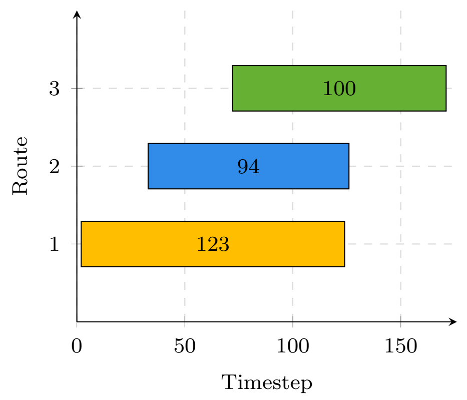


(a) Routing solution. (b) Charging schedule.

Figure 9 RC104-25 optimal solution on the min-cost graph.



(a) Routing solution.



(b) Charging schedule.

Figure 10 RC104-25 optimal solution on the multigraph.

6. Conclusion

Electric vehicles (EVs) are a realistic alternative to reduce the carbon footprint of city logistics operations. However, their limited autonomy and recharging hassles must be addressed before EVs can be massively integrated into urban logistics systems. To facilitate this integration, we introduced the mE-VRSPTW, in which a fleet of EVs must be routed to serve a set of customers and their overnight charging operations scheduled so that the total traveled distance is minimized. In addition to the usual load and time window constraints, routes must comply with energy requirements. Furthermore, the EVs are recharged prior to performing their routes, using a limited number of chargers located at the depot. The recharging process of each EV occurs non-preemptively in a single charger and according to a piecewise-linear recharging function. The problem is defined on a multigraph, meaning that there are alternative arcs (i.e., paths on the road network) to travel between two locations. This representation captures the trade-off between the consumed energy and the traveled distance.

We formulated the mE-VRSPTW as a route-based set-partitioning model. This formulation is solved using a BPC algorithm that implements state-of-the-art techniques. Specifically, the master problem handles the capacity constraints of the chargers at the depot and is reinforced with rounded capacity and subset-row inequalities. The pricing problem works on a network whose structure accounts for the dual variables from the chargers' capacity constraints and ensures that the charge of the EVs is conducted without preemption. To solve the pricing problem, we used a specialized labeling algorithm that employs a dominance rule accounting for energy consumption and recharging time. We also devised a heuristic labeling algorithm that works on a reduced network and uses a simpler dominance criterion. Finally, to find integer solutions, we derived specific-purpose branching rules.

Through extensive computational experiments, we demonstrated that the BPC algorithm could efficiently solve instances with up to 50 customers. The good performance of the algorithm can be attributed to the fact that it uses some of the state-of-the-art techniques for vehicle routing BPC-based algorithms. Furthermore, we analyzed the impact of charging scheduling on computational time. The problem without charging scheduling is solved about three times faster, corroborating that the mE-VRSPTW is a very challenging problem from a theoretical and computational perspective due to its combined routing and scheduling structure. In addition, EV charging scheduling at companies' facilities with limited chargers can be difficult. Thus, it must be addressed when planning EV routing. Finally, we studied the impact of multigraph representation on optimal cost. Using this representation, routing models can find better solutions by having alternatives to balance conflicting resources. Yet, this improvement in solution quality is at the expense of higher computational times.

The central role of EVs in the urgent task of replacing fossil fuels with renewable energy sources is transforming the nature of routing and scheduling problems, exposing several fascinating research opportunities. There are many research streams that can spring from this work. For instance, one could incorporate real-world EVs' energy consumption characteristics into the problem and modeling. This integration is particularly exciting in a multigraph representation as it would rigorously capture the trade-off between conflicting resources. Tackling the mE-VRSPTW in a multi-day planning horizon or accounting for charging-related costs is also an interesting avenue. Overall, the combined routing and scheduling structure of the mE-VRSPTW makes the problem a well-suited test-bed for developments on hybrid large-scale optimization frameworks, such as branch-and-check or data-driven methods that integrate mathematical programming and machine learning. Finally, the BPC algorithm could be used as an analytical tool in real-world goods distribution schemes.

Appendix A: Notation Table for the Problem Statement

Notation	Description
	General Definitions (§3.1)
$\mathcal{C}$	Set of $C \in \mathbb{Z}_+$ customers.
$\mathcal{G} = (\mathcal{V}, \mathcal{A})$	Directed multigraph.
$\mathcal{V}$	Set of vertices.
$\mathcal{A}$	Multiset of arcs.
$\mathcal{A}_{(i,j)}$	Set of arcs from $i \in \mathcal{V}$ to $j \in \mathcal{V}$.
$Q \in \mathbb{Z}_+$	Load capacity of each EV.
$E \in \mathbb{Z}_+$	Maximum amount of energy that an EV can store.
$\underline{T}, \overline{T} \in \mathbb{Z}_+$	Depot's time window opening and closing.
$q_i \in \{0, 1 \dots, Q\}$	Load associated with vertex $i \in \mathcal{V}$.
$\underline{t}_i, \overline{t}_i \in [\underline{T}, \overline{T}]$	Time window opening and closing of vertex $i \in \mathcal{V}$.
$c_{(i,j)^k} \geq 0$	Driving distance along arc $(i,j)^k \in \mathcal{A}$.
$t_{(i,j)^k} \geq 0$	Travel time along arc $(i,j)^k \in \mathcal{A}$.
$e_{(i,j)^k} \in \{0, 1, \dots, E\}$	Energy consumption along arc $(i,j)^k \in \mathcal{A}$.
$\underline{c}_{(i,j)}, \underline{t}_{(i,j)}, \underline{e}_{(i,j)}$	Minimum distance, travel time, and energy consumption from $i \in \mathcal{V}$ to $j \in \mathcal{V}$.
$B \in \mathbb{Z}_+$	Number of chargers at the depot.
$\mathcal{T}$	Set of timesteps at which vehicles can charge.
$\widetilde{T} \in \mathbb{Z}_+$	Latest timestep at which a vehicle can depart from the depot.
f	Piecewise-linear recharging function.
$\mathcal{P}$	Set of $P \in \mathbb{Z}_+$ pieces of the recharging function.
$\phi_{p-1}, \phi_p \geq 0$	Start and end of the energy level associated with piece $p \in \mathcal{P}$.
$\theta_p > 0$	Recharging rate of piece $p \in \mathcal{P}$.
f^{-1}	Inverse recharging function — refer to equation (1).
	Set-Partitioning Formulation (§3.3)
$\mathcal{R}$	Set of all feasible routes.
$a_i^r \in \mathbb{Z}_+$	Number of times that route $r \in \mathcal{R}$ visits customer $i \in \mathcal{C}$.
$a_{(i,j)^k}^r \in \mathbb{Z}_+$	Number of times that route $r \in \mathcal{R}$ traverses arc $(i,j)^k \in \mathcal{A}$.
$b_t^r \in \{0, 1\}$	Constant indicating whether route $r \in \mathcal{R}$ charges during timestep $t \in \mathcal{T}$.
$c_r \in \mathbb{Z}_+$	Cost of route $r \in \mathcal{R}$ — refer to equation (2).
$\lambda_r \in \{0, 1\}$	Variable indicating whether route $r \in \mathcal{R}$ is selected.

Table 2 **Notation table for the problem statement.**

Appendix B: Notation Table for the BPC Algorithm

Notation	Description
	The Master Problem (MP, §4.1)
$\mathcal{R}' \subseteq \mathcal{R}$	Relatively small subset of routes.
$\alpha_i \in \mathbb{R}$	Dual variable associated with constraint (4b) indexed by $i \in \mathcal{C}$.
$\beta_t \leq 0$	Dual variable associated with constraint (4c) indexed by $t \in \mathcal{T}$.
	The Pricing Problem (PP, §4.2)
$\mathcal{G}_{\mathrm{P}} = (\mathcal{V}_{\mathrm{P}}, \mathcal{A}_{\mathrm{P}})$	Directed multigraph, where $\mathcal{V}_{\mathrm{P}} = \{s\} \cup \mathcal{T} \cup \{\underline{0}\} \cup \mathcal{C} \cup \{\overline{0}\}$ and $\mathcal{A}_{\mathrm{P}} = \mathcal{A} \cup \mathcal{A}_1 \cup \mathcal{A}_2 \cup \mathcal{A}_3$.
s	Dummy source vertex.
$\underline{0}$, $\overline{0}$	Source and sink depot vertices.
$\mathcal{A}$	Multiset of arcs connecting vertices of $\mathcal{V} = \{\underline{0}\} \cup \mathcal{C} \cup \{\overline{0}\}$.
$\mathcal{A}_1$	Set of arcs connecting the dummy source with charging time vertices.
$\mathcal{A}_2$	Set of arcs connecting successive charging time vertices.
$\mathcal{A}_3$	Set of arcs connecting charging time vertices with the source depot vertex.
$r_{(i,j)^k} \in \mathbb{R}$	Modified cost of arc $(i,j)^k \in \mathcal{A}_{\mathrm{P}}$ — refer to equation (6).
$\mathcal{N}_i \subseteq \mathcal{C}$	Neighborhood containing the $\Delta \in \mathbb{Z}_+$ closest customers to $i \in \mathcal{C}$ and customer i itself.
$\overline{\Delta} \in \mathbb{Z}_+$	Maximum neighborhood size.
ℓ_i	Label associated with a subpath starting at vertex $i \in \mathcal{V}_{\mathrm{P}}$.
$r(\ell_i) \in \mathbb{R}$	The cumulative reduced cost of label ℓ_i.
$q(\ell_i) \in \mathbb{Z}_+$	The remaining load capacity of label ℓ_i.
$t(\ell_i) \in [\underline{T}, \overline{T}]$	The latest time for starting service of label ℓ_i.
$e(\ell_i) \in \mathbb{Z}_+$	The remaining energy capacity of label ℓ_i.
$b(\ell_i) \in \mathbb{Z}_+$	The number of timesteps required to charge of label ℓ_i.
$\mathbb{I}_c^u(\ell_i) \in \{0,1\}$	Resource indicating whether customer $c \in \mathcal{C}$ is unreachable for label ℓ_i.
$\mathbb{I}_c^{ng}(\ell_i) \in \{0,1\}$	Resource indicating whether label ℓ_i visiting customer $c \in \mathcal{C}$ violates the cycling restrictions.
	Valid Inequalities (§4.3)
$\pi \geq 0$	Dual variable associated with constraint (9).
$\mathcal{S}$	Set of customer triplets for which SRCs have been separated.
$\sigma_S \leq 0$	Dual variable associated with constraint (10) indexed by $S \in \mathcal{S}$.
$\overline{S} \in \mathbb{N}$	Maximum number of cuts that can be added simultaneously.
$\overline{C} \in \mathbb{N}$	Maximum number of times that a customer may appear in a customer triplet.
$\epsilon \geq 0$	Violation threshold to separate cuts.
$\eta_S(\ell_i) \in \{0,1\}$	Number of times modulo 2 that ℓ_i has visited customers in S.
$\mathcal{S}_{\ell_i,\ell_i'} \subseteq \mathcal{S}$	Set of customer triplets $S \in \mathcal{S}$ for which $\eta_S(\ell_i) > \eta_S(\ell_i')$.
	Branching Rules (§4.4)
$x_{(i,j)^k}, (i,j)^k \in \mathcal{A}_{\mathrm{P}}$	Arc-flow variable.
$\widetilde{B} \notin \mathbb{Z}_+$	A fractional flow value.

Table 3 Notation table for the BPC algorithm.

Appendix C: 50-customer RC Test Instances

Table 4 presents RC-50 instances' minimum distance and energy consumption from the depot to the customers 26–34, using the setup described in §5.1. With an energy capacity of $E = 1,600$ dWh, arriving at most of these customers demands more than half of the EVs' driving range. As a result, some customers are unreachable or are serviced exclusively by one EV, which is impractical in real-life scenarios.

To overcome this issue, we perform two changes in this group of instances. First, we set the EVs' energy capacity to $E = 20$ kWh, which is consistent with real-life practices as a long-range EV would be used for these long-distance trips. Second, we define the piecewise-linear recharging function by $\theta_1 = 13.49$, $\theta_2 = 7.14$, and $\theta_3 = 2.00$ kWh/h and $\phi_1 = 17.0$, $\phi_2 = 19.0$, and $\phi_3 = 20.0$ kWh. This latter means that the energy per timestep delivered by the chargers increases, but the time for a full recharge remains unchanged.

Customer $j \in \mathcal{C}$	Min. distance $\underline{c}_{(0,j)}$ (km)	Min. energy consumption $\underline{e}_{(0,j)}$ (dWh)
26	58.5	933
27	57.0	873
28	55.7	898
29	52.2	811
30	52.0	848
31	50.2	786
32	51.0	836
33	51.4	875
34	47.4	749

Table 4 RC-50 instances' minimum distance and energy consumption from the depot to customers 26–34.

Appendix D: Computational Results

References

Alvo M, Angulo G, Klapp MA (2021) An exact solution approach for an electric bus dispatch problem. *Transportation Research Part E: Logistics and Transportation Review* 156:102528.

Asamer J, Graser A, Heilmann B, Ruthmair M (2016) Sensitivity analysis for energy demand estimation of electric vehicles. *Transportation Research Part D: Transport and Environment* 46:182–199.

Baldacci R, Mingozzi A, Roberti R (2011) New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research* 59(5):1269–1283.

Bektaş T, Ehmke JF, Psaraftis HN, Puchinger J (2019) The role of operational research in green freight transportation. *European Journal of Operational Research* 274(3):807–823.

Ben Ticha H, Absi N, Feillet D, Quilliot A (2017) Empirical analysis for the vrptw with a multigraph representation for the road network. *Computers & Operations Research* 88:103–116.

Ben Ticha H, Absi N, Feillet D, Quilliot A (2018) Vehicle routing problems with road-network information: State of the art. *Networks* 72(3):393–406.

Bigazzi AY, Clifton KJ (2015) Modeling the effects of congestion on fuel economy for advanced power train vehicles. *Transportation Planning and Technology* 38(2):149–161.

Bruglieri M, Mancini S, Pisacane O (2021) A more efficient cutting planes approach for the green vehicle routing problem with capacitated alternative fuel stations. *Optimization Letters* 15(8):2813–2829.

Contardo C, Desaulniers G, Lessard F (2015) Reaching the elementary lower bound in the vehicle routing problem with time windows. *Networks* 65(1):88–99.

Costa L, Contardo C, Desaulniers G (2019) Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science* 53(4):946–985.

Dataset EV (2022) Electric vehicle dataset. URL `https://ev-database.org/car/1095/Peugeot-iOn`, accessed: June 29, 2022.

Davis BA, Figliozzi MA (2013) A methodology to evaluate the competitiveness of electric delivery trucks. *Transportation Research Part E: Logistics and Transportation Review* 49(1):8–23.

Instance	Avg. # of altern.	LB	Cuts	Nodes	Time			Vehicles used	UB	Time in use (%)	Chargers in use (%)	Time at capacity (%)
					Total (s)	Master (%)	Pricing (%)					
R101-25	1.60	633.6	0	3	0.99	16.16	40.40	9	634.7	100.00	89.08	76.99
R102-25	1.64	564.9	9	1	0.43	20.93	74.42	8	564.9	100.00	75.50	44.78
R103-25	1.66	493.3	22	9	2.65	11.70	68.68	6	493.9	100.00	74.39	41.81
R104-25	1.65	437.7	0	3	2.73	6.59	76.92	5	437.7	100.00	76.64	51.33
R105-25	1.60	550.7	0	1	0.68	20.59	66.18	7	550.7	100.00	90.54	79.60
R106-25	1.63	515.7	37	15	3.84	11.46	50.52	7	525.5	97.89	69.05	49.36
R107-25	1.66	454.4	4	1	1.10	10.00	88.18	5	454.4	100.00	68.20	38.88
R108-25	1.65	418.6	10	1	1.58	6.33	86.71	5	418.6	68.29	48.31	23.69
R109-25	1.62	446.8	0	1	0.68	10.29	75.00	5	446.8	100.00	82.24	61.53
R110-25	1.62	465.8	34	1	1.97	11.68	81.22	5	465.8	100.00	82.76	60.00
R111-25	1.66	454.4	18	1	1.33	14.29	66.17	5	454.4	100.00	71.52	43.20
R112-25	1.65	412.4	26	1	2.82	6.03	88.30	5	412.4	72.88	47.34	24.06
C101-25	1.63	220.3	40	3	36.49	3.40	93.94	4	221.4	64.92	64.92	64.92
C102-25	1.59	190.3	0	1	3.68	1.90	94.57	3	190.3	61.35	61.35	61.35
C103-25	1.60	190.3	0	1	7.99	1.75	97.00	3	190.3	70.42	70.42	70.42
C104-25	1.61	186.9	0	1	16.47	1.52	97.81	3	186.9	64.49	64.49	64.49
C105-25	1.61	200.3	9	3	15.27	3.54	88.47	3	203.3	100.00	100.00	100.00
C106-25	1.62	221.4	46	1	7.14	11.90	85.01	4	221.4	64.72	64.72	64.72
C107-25	1.61	191.3	0	1	1.97	2.03	94.42	3	191.3	93.00	93.00	93.00
C108-25	1.59	191.3	0	3	3.74	3.48	85.56	3	191.3	91.62	91.62	91.62
C109-25	1.60	191.3	0	1	6.37	1.73	93.88	3	191.3	66.90	66.90	66.90
RC101-25	1.53	541.1	51	1	2.22	15.77	78.38	6	541.1	100.00	92.79	81.62
RC102-25	1.53	516.9	96	9	8.20	18.78	67.20	6	521.4	100.00	80.32	57.56
RC103-25	1.51	498.8	111	3	8.33	9.96	80.31	6	503.4	100.00	76.37	41.74
RC104-25	1.52	312.8	0	1	1.82	3.85	91.21	3	312.8	100.00	62.15	31.17
RC105-25	1.55	524.9	69	1	3.13	15.34	78.59	6	524.9	100.00	79.22	49.75
RC106-25	1.54	510.7	103	1	7.38	13.14	80.62	6	510.7	100.00	73.93	37.27
RC107-25	1.52	363.1	42	3	3.60	13.89	71.67	4	363.7	100.00	68.54	43.07
RC108-25	1.52	294.5	0	1	0.80	3.75	78.75	3	294.5	100.00	67.31	37.01
R201-25	1.65	507.2	25	9	4.47	17.90	41.39	6	507.2	94.13	94.13	94.13
R202-25	1.66	451.3	10	1	1.66	8.43	83.13	5	451.3	57.95	57.95	57.95
R203-25	1.67	442.7	10	3	2.34	5.98	64.96	5	442.7	77.27	77.27	77.27
R204-25	1.66	396.9	0	3	3.07	3.58	53.75	4	396.9	75.08	75.08	75.08
R205-25	1.66	435.4	19	19	3.61	13.57	52.91	5	435.4	71.89	71.89	71.89
R206-25	1.66	419.0	10	5	2.96	7.09	67.91	5	419.0	70.69	70.69	70.69
R207-25	1.66	419.0	10	17	4.24	10.61	54.72	5	419.0	70.69	70.69	70.69
R208-25	1.66	396.5	0	3	2.32	4.74	67.24	4	396.5	71.00	71.00	71.00
R209-25	1.65	429.4	21	1	2.53	10.67	81.03	5	429.4	91.57	91.57	91.57
R210-25	1.67	453.4	9	1	1.82	6.59	73.63	5	453.4	57.41	57.41	57.41
R211-25	1.65	396.5	0	7	2.60	7.31	61.15	4	396.5	79.77	79.77	79.77
C201-25	1.63	287.3	30	1	13.49	4.74	92.44	3	287.3	81.98	81.98	81.98
C202-25	1.63	282.0	34	1	19.94	3.11	95.29	3	282.0	28.87	28.87	28.87
C203-25	1.64	267.9	51	1	46.71	2.33	96.57	3	267.9	21.05	21.05	21.05
C204-25	1.64	267.6	77	1	266.76	1.08	98.35	3	267.6	32.14	32.14	32.14
C205-25	1.62	279.9	25	3	16.73	3.71	88.94	3	279.9	89.25	89.25	89.25
C206-25	1.61	270.3	42	1	27.74	3.46	95.24	3	270.3	15.45	15.45	15.45
C207-25	1.61	256.8	26	1	26.12	1.49	97.28	3	256.8	14.05	14.05	14.05
C208-25	1.61	267.9	60	3	42.53	2.28	89.58	3	267.9	20.67	20.67	20.67
RC201-25	1.55	534.8	58	37	10.55	41.04	24.08	6	534.8	87.93	87.93	87.93
RC202-25	1.54	459.0	49	3	3.58	15.36	65.92	5	459.0	91.52	91.52	91.52
RC203-25	1.52	437.1	63	3	5.25	14.48	66.29	5	437.1	75.00	75.00	75.00
RC204-25	1.52	304.4	0	1	1.57	3.82	83.44	3	304.4	95.22	95.22	95.22
RC205-25	1.54	520.0	75	1	4.05	19.75	67.16	6	520.0	84.86	84.86	84.86
RC206-25	1.55	441.0	44	3	4.42	17.65	60.86	5	441.0	82.80	82.80	82.80
RC207-25	1.53	364.4	29	9	3.03	15.18	52.81	4	364.4	91.30	91.30	91.30
RC208-25	1.52	294.5	0	1	1.14	3.51	86.84	3	294.5	88.85	88.85	88.85

Table 5 Performance of the BPC algorithm on 25-customer instances.

Desaulniers G (2010) Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations Research* 58(1):179–192.

Instance	Avg. # of altern.	LB	Cuts	Nodes	Time			Vehicles used	UB	Time in use (%)	Chargers in use (%)	Time at capacity (%)
					Total (s)	Master (%)	Pricing (%)					
R101-50	1.59	1,158.7	20	109	56.74	20.55	63.52	15	1,168.0	100.00	83.55	66.07
R102-50	1.64	971.5	12	11	11.86	6.91	84.74	12	972.1	100.00	79.04	55.33
R103-50	1.64	859.1	71	31	162.17	5.72	91.38	11	863.0	100.00	73.62	46.60
R104-50	1.65	721.9	131	11	672.72	1.02	98.54	8	727.4	100.00	69.80	29.56
R105-50	1.60	997.0	67	11	11.44	20.72	64.86	12	1,001.5	100.00	82.58	50.50
R106-50	1.64	887.3	82	7	18.88	8.90	85.49	10	890.7	100.00	70.73	43.66
R107-50	1.65	791.3	106	1	38.96	2.67	96.61	9	791.3	100.00	63.94	20.56
R108-50	1.65	716.7	85	3	136.99	1.79	96.92	7	721.9	100.00	67.77	53.84
R109-50	1.62	862.0	139	23	59.38	11.87	81.69	10	868.0	100.00	78.11	34.87
R110-50	1.62	765.1	90	11	103.19	4.25	94.18	9	771.6	100.00	74.48	31.95
R111-50	1.65	795.0	143	23	429.57	2.89	96.16	9	798.3	100.00	76.58	44.10
R112-50	1.65	712.2	165	11	713.26	1.52	98.06	8	717.8	100.00	54.28	8.16
C101-50	1.69	397.3	30	1	12.98	9.63	87.37	6	397.3	98.28	78.96	59.65
C102-50	1.68	366.1	0	1	23.93	1.42	96.03	5	366.1	75.58	64.53	53.48
C103-50	1.68	365.2	19	1	68.48	1.20	97.49	5	365.3	93.13	77.68	62.23
C104-50	1.68	358.0	0	1	216.04	0.39	99.48	5	358.0	83.38	59.46	35.54
C105-50	1.68	375.5	95	3	59.29	8.30	88.33	5	377.4	100.00	94.57	89.14
C106-50	1.69	383.8	47	3	53.76	4.74	92.80	6	388.0	96.91	77.31	57.70
C107-50	1.68	362.4	0	1	6.92	3.47	88.87	5	362.4	100.00	90.14	80.29
C108-50	1.66	362.4	0	1	9.46	2.75	93.13	5	362.4	100.00	80.97	61.94
C109-50	1.67	362.4	0	1	26.38	0.95	98.29	5	362.4	81.41	68.02	54.64
RC101-50	1.49	1,022.6	60	1	30.72	6.51	90.40	10	1,022.6	100.00	84.90	68.01
RC102-50	1.48	897.3	114	3	275.69	2.26	96.90	9	908.1	100.00	88.75	76.00
RC103-50	1.47	814.7	209	3	3,147.23	0.41	99.52	8	815.4	100.00	74.88	54.28
RC104-50	1.48	643.5	73	1	144.86	0.86	98.90	6	643.5	100.00	55.35	26.78
RC105-50	1.48	910.5	104	5	179.45	7.78	90.64	9	911.6	100.00	90.24	81.81
RC106-50	1.48	796.3	168	3	436.36	2.11	97.35	7	798.5	100.00	86.87	63.53
RC107-50	1.48	675.9	90	1	133.68	0.66	98.95	6	675.9	100.00	64.33	40.28
RC108-50	1.48	625.5	50	1	131.70	0.75	98.72	6	625.5	96.72	50.51	22.13
R201-50	1.64	897.8	43	1	30.66	40.48	53.82	10	897.8	95.40	80.32	65.24
R202-50	1.65	815.7	43	1	30.63	14.10	79.56	9	815.7	100.00	83.54	67.09
R203-50	1.66	732.7	51	3	36.97	8.55	81.85	8	732.7	99.80	81.26	62.71
R204-50	1.65	705.3	90	5	82.48	4.89	90.75	7	705.3	60.62	40.52	20.41
R205-50	1.65	791.1	17	75	380.71	32.53	29.40	9	792.3	100.00	92.85	85.71
R206-50	1.65	753.9	30	19	30.03	18.95	68.43	8	753.9	100.00	91.00	82.01
R207-50	1.65	712.1	196	3	323.40	4.16	94.33	8	717.8	51.73	41.53	31.32
R208-50	1.65	680.4	78	3	71.23	3.33	93.46	7	680.4	63.57	42.23	20.89
R209-50	1.65	755.1	143	41	284.14	14.63	79.69	8	760.5	100.00	78.71	57.43
R210-50	1.66	767.8	71	5	44.32	6.84	88.56	8	768.1	87.05	73.27	59.49
R211-50	1.65	681.0	100	15	106.76	9.16	85.46	7	681.0	96.56	72.99	49.42
C201-50	1.69	519.5	139	1	77.75	20.50	78.37	6	519.5	28.25	28.25	28.25
C202-50	1.69	519.5	192	1	320.50	11.75	85.75	6	519.5	31.46	31.46	31.46
C203-50	1.69	460.6	55	5	147.18	1.96	92.40	5	460.6	44.40	44.40	44.40
C204-50	1.69	459.3	80	3	871.08	0.81	98.16	5	459.3	29.43	29.43	29.43
C205-50	1.68	518.1	148	1	152.86	11.96	85.82	6	518.1	39.48	39.48	39.48
C206-50	1.68	465.3	0	1	31.48	2.57	96.28	5	465.3	28.49	28.49	28.49
C207-50	1.67	453.8	0	1	87.40	1.24	97.56	5	453.8	40.96	40.96	40.96
C208-50	1.68	463.6	30	3	87.06	2.60	94.30	5	463.6	31.19	31.19	31.19
RC201-50	1.50	814.9	44	1	13.24	14.35	82.85	7	814.9	100.00	86.43	72.86
RC202-50	1.48	753.3	25	3	25.35	8.28	77.12	7	753.3	93.75	80.07	66.40
RC203-50	1.48	734.3	33	3	20.68	4.35	89.89	7	734.3	98.69	79.58	60.47
RC204-50	1.48	548.9	0	1	19.25	1.71	97.04	5	548.9	70.50	36.33	2.15
RC205-50	1.49	784.8	53	1	27.10	24.61	70.15	7	784.8	100.00	92.95	85.91
RC206-50	1.50	765.5	91	7	50.66	13.46	70.49	7	765.5	99.75	75.79	51.84
RC207-50	1.49	653.6	24	3	17.15	3.73	87.29	6	653.6	76.71	50.19	23.66
RC208-50	1.48	543.7	0	1	35.37	0.90	98.25	5	543.7	62.33	33.86	5.40

Table 6 Performance of the BPC algorithm on 50-customer instances.

Desaulniers G, Desrosiers J, Spoorendonk S (2011) Cutting planes for branch-and-price algorithms. *Networks* 58(4):301–310.

Desaulniers G, Errico F, Irnich S, Schneider M (2016) Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research* 64(6):1388–1405.

Desaulniers G, Gschwind T, Irnich S (2020) Variable fixing for two-arc sequences in branch-price-and-cut algorithms on path-based models. *Transportation Science* 54(5):1170–1188.

Desaulniers G, Lessard F, Hadjar A (2008) Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. *Transportation Science* 42(3):387–404.

Desaulniers G, Pecin D, Contardo C (2019) Selective pricing in branch-price-and-cut algorithms for vehicle routing. *EURO Journal on Transportation and Logistics* 8(2):147–168.

Desrochers M, Desrosiers J, Solomon M (1992) A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research* 40(2):342–354.

Dror M (1994) Note on the complexity of the shortest path models for column generation in vrptw. *Operations Research* 42(5):977–978.

Erdoğan S, Miller-Hooks E (2012) A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation review* 48(1):100–114.

Feillet D, Dejax P, Gendreau M, Gueguen C (2004) An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks* 44(3):216–229.

Florio AM, Absi N, Feillet D (2021) Routing electric vehicles on congested street networks. *Transportation Science* 55(1):238–256.

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

Froger A, Mendoza JE, 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.

Irnich S, Desaulniers G (2005) Shortest path problems with resource constraints. *Column Generation*, 33–65 (Springer).

Jepsen M, Petersen B, Spoorendonk S, Pisinger D (2008) Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research* 56(2):497–511.

Kohl N, Desrosiers J, Madsen OB, Solomon MM, Soumis F (1999) 2-path cuts for the vehicle routing problem with time windows. *Transportation Science* 33(1):101–116.

Kullman ND, Goodson JC, Mendoza JE (2021) Electric vehicle routing with public charging stations. *Transportation Science* 55(3):637–659.

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

Lebeau P, Macharis C, Van Mierlo J (2016) Exploring the choice of battery electric vehicles in city logistics: A conjoint-based choice analysis. *Transportation Research Part E: Logistics and Transportation Review* 91:245–258.

Marra F, Yang GY, Træholt C, Larsen E, Rasmussen CN, You S (2012) Demand profile study of battery electric vehicle under different charging options. *2012 IEEE Power and Energy Society General Meeting*, 1–7 (IEEE).

Mendoza JE, Guéret C, Hoskins M, Lobit H, Pillac V, Vidal T, Vigo D (2014) VRP-REP: a vehicle routing community repository, in 'third meeting of the euro working group on vehicle routing and logistics optimization (VeRoLog). *Oslo (Norway)* 56:77.

Michail D, Kinable J, Naveh B, Sichi JV (2020) Jgrapht—a java library for graph data structures and algorithms. *ACM Transactions on Mathematical Software* 46(2):1–29.

Montoya A, Guéret C, Mendoza JE, Villegas JG (2016) A multi-space sampling heuristic for the green vehicle routing problem. *Transportation Research Part C: Emerging Technologies* 70:113–128.

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

Morganti E, Browne M (2018) Technical and operational obstacles to the adoption of electric vans in france and the uk: An operator perspective. *Transport Policy* 63:90–97.

Pelletier S, Jabali O, Laporte G (2016) 50th anniversary invited article—goods distribution with electric vehicles: review and research perspectives. *Transportation Science* 50(1):3–22.

Pelletier S, Jabali O, Laporte G (2018) Charge scheduling for electric freight vehicles. *Transportation Research Part B: Methodological* 115:246–269.

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

Perumal SS, Lusby RM, Larsen J (2021) Electric bus planning & scheduling: A review of related problems and methodologies. *European Journal of Operational Research* .

Sassi O, Oulamara A (2014) Joint scheduling and optimal charging of electric vehicles problem. *International Conference on Computational Science and its Applications*, 76–91 (Springer).

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

Schneider M, Stenger A, Hof J (2015) An adaptive VNS algorithm for vehicle routing problems with intermediate stops. *Or Spectrum* 37(2):353–387.

Solomon MM (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research* 35(2):254–265.

Toth P, Vigo D (2014) *Vehicle routing: problems, methods, and applications* (SIAM).

Zhou Y, Meng Q, Ong GP (2022) Electric bus charging scheduling for a single public transport route considering nonlinear charging profile and battery degradation effect. *Transportation Research Part B: Methodological* 159:49–75.