A Branch-Cut-and-Price Algorithm for the Time-Dependent Electric Vehicle Routing Problem with Time Windows

Gonzalo Lera-Romero[*†] Juan José Miranda-Bront[‡§] Francisco J. Soulignac[†‡]

glerarom ero@dc.uba.ar, jmiranda@utdt.edu, fsoulign@dc.uba.ar

Abstract

The adoption of electric vehicles (EVs) within last-mile deliveries is considered one of the key transformations towards more sustainable logistics. The inclusion of EVs introduces new operational constraints to the models such as a restricted driving range and the possibility to perform recharges in-route. The discharge of the typical batteries is complex and depends on several variables, including the vehicle travel speed, but most of the approaches assume that the energy consumption depends only on the distance traveled. This becomes relevant for distribution problems in large cities, where traffic congestion affects severely the travel speeds. In this paper, we introduce a general version of the Time-Dependent Electric Vehicle Routing Problem with Time Windows (TDEVRPTW), which incorporates the time-dependent nature of the transportation network both in terms of travel times and the battery consumption. We propose a unifying framework to integrate other critical time-dependent times arising during the operations previously studied in the literature, such as the waiting and charging times. We propose a state of the art branch-cut-and-price (BCP) algorithm. Based on extensive computational experiments, we show that the approach is very effective solving instances with up to 100 customers with different time dependent contexts and is able to find 13 new optimal solutions for time-independent instances. From a managerial standpoint, our experiments indicate that neglecting the travel speeds can affect the quality of the solutions obtained, where up to 40 percent of the infeasibilities induced by neglecting the time dependency can be caused by exceeding the battery capacity.

Keywords: electric vehicle routing problem; time-dependent times; branch cut and price; labeling algorithms

1 Introduction

One of the key environmental challenges nowadays relies in the use of cleaner and more sustainable energy resources to reduce emissions and pollution worldwide. Providing access to new and sustainable transportation solutions appears as one of the urgent topics to be addressed from a logistics perspective. As of 2010, it is estimated that 20% of the emission of greenhouse gas (GHG) in that region stemmed from transportation activities.

In the last decade, many companies have been shifting into more environmentally friendly transportation alternatives. One area of significant impact is last-mile logistics, where traditional internal combustion engine vehicles (ICEVs) are being replaced by electric vehicles (EVs) and cargo-bikes, specially in highly congested areas. For instance, UPS aims to have 25% of its vehicles running on alternative fuel by 2020, including

[*]Universidad de Buenos Aires. Facultad de Ciencias Exactas y Naturales. Departamento de Computación. Buenos Aires, Argentina.

[†]CONICET-Universidad de Buenos Aires. Instituto de Investigación en Ciencias de la Computación (ICC). Buenos Aires, Argentina.

[‡]Escuela de Negocios, Universidad Torcuato Di Tella, Buenos Aires, Argentina.

[§]Consejo Nacional de Investigaciones Científicas y Técnicas.

10,000 EVs (UPS, 2019). In addition, governments are introducing incentives to stimulate the acquisition and construction of EVs, while the European Union is also planning to introduce penalties for the emission of CO_2 within routing and logistics (EPEC 2011).

From a technical perspective, EVs offer several benefits compared to the ICEVs, such as less noise contamination, CO_2 emissions and a high energy conversion efficiency. They are usually combined with other methods such as the regenerative braking system, which produces electrical energy from movement. As a counterpart, the restricted battery capacity influences the operations of the EVs by reducing the so-called driving range. Then, routing plans for EVs should incorporate this new operational constraints explicitly, preventing vehicles from running out of battery by making intermediate stops at recharging stations. Recent advances have increased the driving range in the last few years, and initially it may seem sufficient for distribution within cities. However, the battery capacity declines over time and their continuous replacements would represent significant investments. Other aspects such as the use of air conditioning or heating systems also reduce the driving range, which can be up to 30% of the original capacity according to Restrepo et al. (2014).

The Electric Vehicle Routing Problem (EVRP) introduced by Schneider et al. (2014) extends the classical Vehicle Routing Problem (VRP) by incorporating explicitly the battery of the vehicle. It is modelled as an additional resource that is consumed when the vehicle moves along the network, and can be recharged in-route. Most of the literature considers a simplified model where the battery discharge is proportional to the distance traveled, assuming it remains constant during the planning horizon. A similar observation holds for the charging times in the fueling stations. In practice, according to Goeke and Schneider (2015) and as observed in Schneider et al. (2014) the vehicle load, travel speed, and gradient of the terrain are among the most impactful variables on the battery consumption. When used for last-mile logistics in large cities, variations in traffic impact directly on the travel speeds along the day, which in turn affect the battery discharge. As a result, on-route battery levels could be wrongly estimated under the classical linear discharge model, leading to tactical plans that may not be applicable in practice due to unexpected battery depletion. The aim of this paper goes in that direction. We study the time-dependent EVRPTW (TDEVRPTW), which incorporates the effects of congestion into the EVRPTW originally proposed in Schneider et al. (2014), not only in terms of the timing but also on the charge of the battery. Furthermore, we provide a general framework that naturally integrates other relevant operational aspects affecting the applicability of the routing plans, such as variable charging times and waiting times at the recharge stations. We first provide a discussion on the literature related to the EVRPTW and the TDVRP to frame our research, and then we outline our main contributions.

1.1 Literature review

Enviromental aspects within VRPs have been addressed from different perspectives. A stream of research related to *green logistics* considers, among others, the effects of transportation on the environment by minimizing fuel consumption and GHG emissions under different congestion conditions (see, e.g. Heni, 2018 and Toth and Vigo, 2014, Chapter 15). To the best of our knowledge, the first approach including alternative fueling vehicles with a limited driving range is the Green VRP (GVRP) proposed by Erdoğan and Miller-Hooks (2012). Although they do not specifically target the use of EVs, they provide an initial framework for these type of problems. Aligned with this, Davis and Figliozzi (2013) study the impact of replacing ICEVs by EVs in routing plans and Felipe et al. (2014) extend the GVRP by introducing the partial-recharge policy, which allows routes to determine the battery charged at each fueling station, as well as multiple recharging technologies by assigning a different recharging speed to each station.

Recently, several papers consider explicitly distribution problems using a fleet of EVs, where the battery of the vehicle is consumed as the vehicle operates and must be replenished in order to eventually extend the length of the tour. The EVRP with time windows (EVRPTW) is first introduced in Schneider et al. (2014), where the GVRP is extended with limited battery constraints and other classical VRP constraints (e.g. time windows and vehicle capacities), and approached with metaheuristic algorithms. The discharge of the battery is simplified and assumed to be linear with respect to the distance travelled. Desaulniers et al. (2016) develop a Branch-Cut-and-Price (BCP) algorithm for a generalization of the EVRPTW where full and partial recharge are allowed, as well as restricting the number of visits to a recharge station. In this same direction, Roberti and Wen (2016) tackle the single-vehicle version of the problem. Partial recharges are also considered in Keskin and Çatay (2016), while Schiffer and Walther (2017) also incorporates location decisions. We remark that these problems can be formulated as a special case of VRP with intermediate stops, a more general family of VRPs recently surveyed in Schiffer et al. (2019).

Another interesting, complementary stream of research focuses on enriching the EVRPTW and some related variants by incorporating characteristics closer to real-world operations. More complex and realistic energy consumption models for electric batteries, based on tests conducted using real data, are proposed by Goeke and Schneider (2015) and Wu et al. (2015). We remark that Goeke and Schneider (2015) extend the EVRPTW to consider a mixed fleet composed of both EVs and ICEVs. Demir et al. (2014) present an updated review of the different consumption models.

The interaction between the vehicles and the recharging infrastructure introduces new interesting operational constraints, with a direct impact regarding the quality and the feasibility of the solutions. These constraints are motivated by practical contexts, and have been tackled in general independently from each other. Sassi et al. (2014) consider time-dependent charging costs and a mixed fleet of vehicles, modelling the dynamic pricing strategies in smart grid networks. The recharge process is also affected regarding the operations. Montoya et al. (2017) focus on the charging process, observing that the amount of battery charged depends not only on time spent but also on the preliminary battery charge. The charging functions modelling the process are nonlinear, and they show that continuous piecewise linear functions with up to 3 pieces result in good approximations. This problem is also studied by Froger et al. (2019) by developing new algorithms that improve the quality of the solutions. In addition, the limited capacity of a station (in terms of chargers) may be exceeded by a high demand of vehicles during peak hours. This case is considered in Keskin et al. (2019), where the vehicles cannot access to a charger immediately upon arrival to a station, and therefore incur in waiting times that affect the rest of the route. Their experiments suggest that considering waiting times can result in up to a 26% difference in terms of costs, and that the waiting times can be modelled by continuous piecewise linear functions satisfying the First-In-First-Out (FIFO) condition. Overall, we refer the reader to Pelletier et al. (2016) for complete survey of the literature considering both modelling and algorithmic aspects.

Congestion effects on routing decisions have received increasing attention during the last few years, as they represent a critical aspect regarding last mile deliveries in large cities (see e.g. Savelsbergh and Van Woensel, 2016). Time-Dependent VRPs (TDVRPs, see e.g. Gendreau et al., 2015) is the name assigned to a wide family of interesting optimization problems that incorporate explicitly the traffic conditions at a planning level by assuming that the travel time between two customers is variable and does not remain constant during the planning horizon. Regarding congestion, the model proposed by Ichoua et al. (2003) has been widely accepted as a standard within the VRP community. Briefly, variable travel speeds are modelled as a step function over the planning horizon, and the vehicle moves along the network according to these speeds depending on the departure time from a customer. As a result, the travel time between two customers is a continuous piecewise linear function that satisfies the FIFO condition.

From an algorithmic perspective, TDVRPs require more complex models, algorithms and implementations to handle variable travel times. We restrict our review to exact algorithms, although (meta)heuristic approaches have been also proposed recently in the related literature. The Time-Dependent Traveling Salesman Problem (TDTSP) considers a single-vehicle routing problem where the objective is to minimize the makespan of the tour. The TDTSP is tackled by Cordeau et al. (2014); Adamo et al. (2019) using Integer Linear Programming (ILP) and solving an auxiliary time-independent problem to compute lower bounds. A similar approach is followed for the TDTSP with Time Windows (TDTSPTW) by Arigliano et al. (2018, 2019). An alternative objective function allows to delay the departure of the vehicle and minimizes the duration of the route instead of the makespan have been approached by Vu et al. (2020) using Dynamic Discretization Discovery and by Lera-Romero and Miranda-Bront (2019) via classical ILP techniques. Dynamic Programming (DP) algorithms were studied by Sun et al. (2018) for a profitable variant of the TDTSPTW with pickup and deliveries (TDTSPPD), and more recently by Lera-Romero et al. (2020b) to tackle the TDTSP and TDTSPTW.

Exact algorithms were also studied for several variants of the TDVRP. Dabia et al. (2013) propose a Branch-and-Price (BP) algorithm for the TDVRP with time windows (TDVRPTW) where the objective function minimizes the total duration of the routes. The pricing problem is tackled using tailored algorithms that incorporate the time dependency. Recently, Lera-Romero et al. (2020a) approach the TDVRPTW with a BCP algorithm. The TDVRPTW with Pickup and Deliveries (TDVRPTWPD) is studied in detail by Sun et al. (2018) using similar ideas, while a variant of the TDVRPTW with path flexibility is explored by Huang et al. (2017).

The literature connecting the EVRPTW and the TDVRP is somehow scarce, and we highlight some recent research in this direction. Fukasawa et al. (2018) studies a complex optimization problem where, in addition to the routing decisions, the speed of the vehicles in each instant is also a decision variable in order to minimize the costs associated with fuel consumption. Pelletier et al. (2019) consider an optimization problem under the uncertainty of some external variables such as road friction, vehicle speed, and weather by estimating the deviation of the discharge rate in typical scenarios. Shao et al. (2017) enhance the classical EVRPTW by adding time-dependent travel times to obtain more accurate estimations regarding potential violations of the time windows, although the battery discharge model remains simplified and is not affected by the variable speeds. Recently, and simultaneously to our work, Lu et al. (2020) take the first steps towards a model that considers the impact of the time-dependent speeds on the energy consumption. However, they assume that the congestion can be captured with a step function having only three pieces with different travel speeds. Besides the limitations regarding the applicability in practice, the proposed model does not scale for more general contexts. As far as we know, no article in the literature proposed a general model integrating the effects of congestion into the battery discharge model for EVs.

1.2 Our Contributions

We build upon the works by Goeke and Schneider (2015) and Ichoua et al. (2003) to study the TDEVRPTW, a generalization of the EVRPTW that captures the effect of congestion on both the travel times and the battery consumption. However, our conception of time-dependency goes beyond the classical time-dependent travel times, as we propose a unifying framework that naturally integrates further operational constraints such as the variable charging times (Montoya et al., 2017) and the waiting times (Keskin et al., 2019), both of which are modelled with continuous piecewise linear functions. Through our experiments, we provide valuable managerial insights on the impact of congestion in the driving range. For some of the scenarios considered, up to 40% of the routes obtained in a time-independent setup present infeasibilities caused by the driving range when evaluated in a time-dependent one.

We design a BCP algorithm including state-of-the-art components. We develop a labeling algorithm capable of handling piecewise linear functions within each label and including partial dominance rules. The BCP is enhanced with new, tailored preprocessing rules for the TDEVRPTW and a new branching scheme that reduces the number of enumerated nodes in a BP. Extensive computational experiments demonstrate that our approach is effective, solving instances with up to 100 customers and finding 13 new optimal solutions compared to Desaulniers et al. (2016). Furthermore, the proposed algorithm showed to be robust for the time-dependent instances as well, where the results are comparable to the ones obtained when time-dependency is neglected.

The rest of the paper is organized as follows. Section 2 defines the TDEVRPTW in its general fashion and introduces the notation used in the paper. Then, Section 3 reviews the classical preprocessing techniques for the EVRPTW and presents a new procedure specifically designed for the TDEVRPTW. The BCP algorithm is presented in Section 4. Finally, the numerical experiments are presented in Section 5 with conclusions and a discussion of the possible future directions in Section 6.

2 Time-dependent electric vehicle routing problem

Next, we define the TDEVRPTW in its general fashion and provide a brief analysis of its novel characteristics. Then, we review the variants from the literature that are incorporated later on into the algorithms described in Section 4.1. Finally, we introduce the new time-dependent battery consumption model, which integrates the variable speeds from congestion into the battery discharge functions.

2.1 Problem description

The TDEVRPTW is defined on a directed graph $D = (V, A)$ where each vertex $i \in V$ represents a location that can be either the depot, a customer, or a (recharging) station. Thus, $V = \{o, d\} \cup V_c \cup V_s$, where o and d are two vertices representing the depot, V_c is the set of customers and V_s the set of recharging stations. Each arc $(i, j) \in A$ represents a path between locations i and j, where c_{ij} denotes its travel cost. Every route must depart from the start depot $o \in V$ and finish at the end depot $d \in V$.

Classical routing problems incorporate real-life operational constraints such as vehicle capacity, time windows, and service times. We assume that an unlimited fleet of homogeneous EVs is available, where each vehicle has a capacity Q. Each customer $i \in V$ has a positive demand q_i, a time window $[a_i, b_i]$ where the service must start, and a service time s_i to process the customer.

Congestion is modelled following the approach proposed by Ichoua et al. (2003). In this model, operations take place inside a time horizon $[0, T]$, which typically represents the length of a working day. Each arc $(i, j) \in A$ has an associated distance d_{ij} and a *speed function* $v_{ij}(t)$ indicating the speed of any vehicle traversing arc (i, j) at time t. In this model, $v_{ij}(t)$ is a step function of known, fixed average speeds defined over a partition of the planning horizon given as input. Then, a *travel time function* $\tau_{ij}(t)$ is derived for each arc $(i, j) \in A$, indicating the travel time of the trip from i to j if departing at time $t \in [0, T]$. Function $\tau_{ij}(t)$ is continuous, piecewise linear and satisfies the FIFO property, meaning that delaying the departure cannot lead to an earlier arrival time.

Each EV contains a battery with a limited capacity B (expressed in kWh). Although the battery is discharged while the vehicle is in motion, it can be recharged at any station to continue the route. For each arc $(i, j) \in A$ we compute a continuous piecewise linear *battery consumption function* $\beta_{ij}(t)$ that indicates the energy consumed while traveling from location i to j if departing from i at time t. This function is derived from the travel speed functions as explained later in Section 2.4. Besides congestion, time-dependent times can also occur when recharging the battery at a charging station. We incorporate this characteristic to the problem following the ideas proposed by Montoya et al. (2017). For each station $j \in V_s$, we define an invertible concave *recharge function* $\hat{g}_j(t)$ that indicates the energy (kWh) recharged after t units of time in a vehicle with an empty battery. For an EV with an initial battery level of $w_0 > 0$ kWh, the charge obtained after t units of time is given by $g_j(w_0, t) = \min\{B - w_0, \hat{g}_j(\hat{g}_j^{-1}(w_0) + t) - w_0\}$. These definitions generalize the EVRPTW, which can be retrieved by setting the battery-discharge and recharging functions as linear functions and considering constant travel time functions. Section 2.4 provides a deeper analysis regarding these functions, including the necessary algorithms to precompute them.

The cost of a route $r = (o, v_1, \ldots, v_k, d)$ is given by $c_r = \sum_{(i,j)\in r} c_{ij}$. The TDEVRPTW involves finding a set of *feasible* routes visiting each customer exactly once at minimum total cost. To determine whether a route is feasible, it must be ensured that the capacity of the vehicles is not exceeded, that the service at each customer starts within its time window, and that the battery is never depleted during the trip. Note that the last two conditions depend on the timing of the route, as both the travel time and the battery consumption depend on the speed that, in turn, depends on the time of departure from each of the traversed vertices. As we discuss in the next section, the departure of a vehicle from a vertex can be delayed to improve the battery consumption. Thus, the specific details needed to determine if a route is feasible are given after the battery discharge model is discussed. We further remark that while most time-dependent models have an objective function that minimizes the makespan or duration of the routes, we consider a more classical, time-independent objective function. This decision is motivated by several reasons, both regarding the model as well as their algorithmic implications. We defer this discussion to Section 4.3.5, in order to provide further insights and a detailed explanation in this regard.

2.2 Delayed departure

Frequently, tactical plans assume that the vehicles leave the customer immediately after they finish the service. This is the case in most of the classical variants of the VRP, motivated by the fact that travel times obey the FIFO property and, therefore, delaying the departure cannot improve the solution. However, under the presence of some specific type of resources, delaying the departure could result beneficial for the overall solution. This is the case of the battery level, where waiting before departing a customer can lead to a smaller battery consumption, enabling potentially to reach a different (improved) route. Indeed, as we show later, travel speed and battery level are usually inversely correlated, and finding a trade-off is in general mandatory.

Our model incorporates this aspect and allows a vehicle to delay its departure from a customer or station. As a result, the waiting time at the departure becomes a decision variable, which a priori translates into additional complexity (especially for some MIP-based approaches). However, we show in Section 4 that this is not the case for labeling algorithms and, in fact, can be handled efficiently.

In order to avoid solutions including unrealistic waiting times, we limit the deferral of the departure to $b_i + s_i$ when visiting $i \in V$ (eventually, up to T for a station $i \in V_s$). From a managerial perspective, waiting times at the departure enable some interesting interpretations. They can be seen as a buffer regarding potential unexpected delays during the operations. Furthermore, the existence of these waiting times suggest that the speed could be reduced for the previous trip when executing the route. Note that this is not determined at the planning stage as travel speeds are not a decision variable. To illustrate this concept, suppose the optimal solution indicates that a vehicle must depart from a vertex A at a time t_0 to arrive a customer B at time t_1. Then, the vehicle must wait until $t_2 > t_1$ before delivering the goods to depart to a vertex C. Alternatively, the driver can depart A at t_0 but adjusting the travel speed on-route aiming to arrive at B at time t_2, instead of t_1. Thus, in this second scenario the vehicle travels at a slower speed between A and B, thus consuming less battery, and can continue with the remaining plan as expected.

The classical preprocessing rules to tighten the time windows clearly impact possible delays to leave a vertex. Since this limit is somehow arbitrary in our work, we decided to prioritize the algorithmic aspect and apply the preprocessing first, and consider the maximum delay using the preprocessed time windows. Note, however, that the proposed framework remains valid if considering the original time windows becomes a hard constraint.

2.3 Further characteristics for the EVRP

We next review some of the variants that have been explored recently for the EVRPTW. In the next sections, we show how to incorporate them into the time-dependent model and algorithms.

Most of the literature related to EVs handles recharges with either one of two policies: full or partial. Whereas the *full-recharge* policy forces all vehicles to charge batteries up to their full capacity at each station, the *partial-recharge* scenario provides a more granular choice. Partial recharge extends the solution space by indicating not only the path, but also the amount of battery recharged at each station visited within the route. Desaulniers et al. (2016) also limit the number of in-route recharges to one. This policy is called the *single-recharge* policy, while the former is referred to as the *multiple-recharge* policy.

As suggested by Keskin et al. (2019), the assumption that a vehicle can perform a recharge immediately upon arrival to a charging station can sometimes lead to unrealistic solutions. This observation is relevant in the context of city logistics, as congestion is likely to occur at the recharging stations due to an increased demand during peak hours. To model this scenario, they define a time-dependent waiting time function $\omega_s(t)$ for each station $s \in V_s$ indicating the expected waiting time until a terminal is available if a vehicle arrives at station s at time t. We remark that these functions are also continuous, piecewise linear, and satisfy the FIFO property.

Multiple approaches have been proposed to formulate the battery recharging model. Recall that $\hat{g}(t)$ indicates the battery (in kWh) recharged in t units of time in a vehicle with an empty battery. The EVRPTW, in its original setup, considers that $\hat{g}(t)$ is a linear function where the slope stands for the *recharging rate*. Therefore, the time required to recharge w kWh of energy remains the same independently whether the

Notation	Value	Description
g	9.81 meters per square second	Gravitational constant
ρ	1.42 kilograms per cubic meter	Air density
cr	0.006	Coefficient rolling resistance
cd	0.9	Coefficient aerodynamic drag
FS	2.40 square meters	Frontal surface
m_c	900 kilograms	Curb mass

Table 1: Variable definitions and suggested values as proposed by Demir et al. (2014). Vehicle mass and surface from the city delivery truck Alke ATX 320E.

battery is empty or, for instance, half-full charged. According to Montoya et al. (2017), the recharging time is not linear. Indeed, recharging the initial 75% of the battery requires a similar time than the last 25%. They show that considering $\hat{g}(t)$ as a continuous piecewise linear function with three pieces results in a good estimation of the recharge times based on collected data. Moreover, this function is concave and invertible.

Finally, note that some EVs admit multiple charging modes, such as different charging speed or technologies for a given range of the battery capacity. This can be captured by defining a specific charging function for each case. In such a context, each station is associated with a specific *charging mode*, which indicates the recharging function that must be employed. Notice that if a recharging station admits two different modes, then we can simply duplicate the station. A fixed cost for each mode can be incorporated into the inbound arcs to each station as well.

2.4 Battery consumption model

We consider the model proposed in Goeke and Schneider (2015), where the battery consumption depends on variables grouped into three categories: vehicle mass, speed, and conditions of the terrain. Let $P(v, q)$ be the *instantaneous consumption function* of a vehicle traveling at speed v with load q, defined by the following equation.

$$P(v, q) = \left(\frac{1}{2} \cdot cd \cdot \rho \cdot FS \cdot v^2 + (m_c + m_u q) \cdot g \cdot (\sin(\alpha) + cr \cdot \cos(\alpha))\right) \cdot v. \tag{1}$$

The rest of the parameters are defined in Table 1. Note that the mass m_u of each of the q units loaded in the vehicle is not specified. This variable depends on the type of the delivered goods, which is problem-dependent. Similarly, the variable α indicates the gradient of the terrain, which depends on the road conditions.

Since we focus on the effects of the congestion, we adapt this formula to achieve a reasonable compromise between the simplicity and expressiveness of the model. We assume that the mass of each unit is negligible compared to the vehicle mass ($m_u = 0$), and that the gradient is zero ($\alpha = 0$). Thus, we define $h_1 = \frac{1}{2} \cdot cd \cdot \rho \cdot FS$ and $h_2 = m_c \cdot g \cdot cr$, and rewrite $P(v, q)$ in the following fashion:

$$P(v) = h_1 v^3 + h_2 v. \tag{2}$$

Table 1 shows the estimated values for each parameters suggested by Demir et al. (2014). Replacing these values in the former expressions result in $h_1 \approx 1.54$ and $h_2 \approx 52.97$. Thus, we can infer that, under these conditions, $h_2 \approx 35 h_1$. This relation becomes relevant to create synthetic instances: given a single discharge rate h (for instance, as considered in the EVRPTW), an instantaneous consumption function can be derived by setting $h_1 = \frac{h}{35}$ and $h_2 = h - \frac{h}{35}$, assuming $v = 1$ for the EVRPTW. Establishing the analogous connection between the average travel speed and the speed functions will later enable us to evaluate the effects of congestion when compared to the EVRPTW.

Observe that the instantaneous consumption function $P(v)$ indicates the energy consumed in an instant of time by a vehicle traveling at speed v. However, the battery discharge function $\beta_{ij}(t)$ defined in Section 2.1 refers to the total energy consumed during the traversal of arc $(i, j) \in A$ if departing at time t. To compute the battery discharge function $\beta_{ij}(t)$ we must combine the time-dependent travel times with the instantaneous

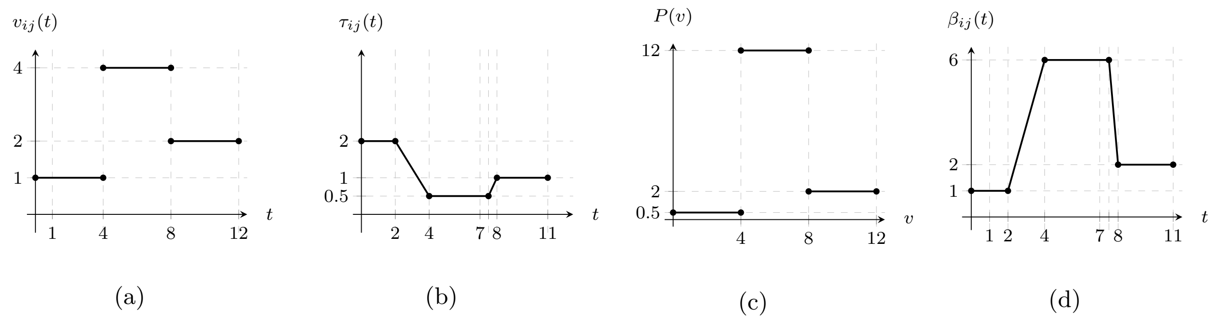


Figure 1: Numerical example illustrating the speed function $v_{ij}(t)$ (1a); travel time function $\tau_{ij}(t)$ (1b) ; instantaneous consumption function $P(v)$ (1c) ; and battery discharge function $\beta_{ij}(t)$ (1d).

consumption functions $P(v)$. If a vehicle departs from vertex i to vertex j at time t, then it arrives j at time $t + \tau_{ij}(t)$. Since the vehicle is traversing arc (i, j) during interval $[t, t + \tau_{ij}(t)]$ then the battery discharge $\beta_{ij}(t)$ can be computed by integrating the instantaneous consumption function over that period of time.

$$\beta_{ij}(t) = \int_{t}^{t+\tau_{ij}(t)} P(v_{ij}(x))dx \tag{3}$$

Recall that the speed function $v_{ij}(t)$ over arc $(i, j) \in A$ is a step function; let $|v_{ij}|$ be its number of pieces. If the k-th piece of v_{ij} ($k = 1, \dots, |v_{ij}|$) is defined on the domain $[T_k, T_{k+1})$ and has a constant value $\bar{v}_{ij}^k$, then Equation (3) can be rewritten as

$$\beta_{ij}(t) = \sum_{k=1}^{|v_{ij}|} P(\bar{v}_{ij}^k) \times |[T_k, T_{k+1}) \cap [t, t + \tau_{ij}(t)]| \tag{4}$$

Similarly to the case of the travel time functions, the battery discharge functions are piecewise linear and continuous, as stated in the following proposition.

Proposition 1. *Given an arc $(i, j) \in A$, the battery discharge function $\beta_{ij}(t)$ is piecewise linear and continuous.*

Proof. The result follows since (4) is a composition of continuous piecewise linear functions. □

Finally, to fully characterize the battery discharge functions, it is necessary to describe their breakpoints. We remark that τ_{ij} is computed in a similar fashion as β_{ij}, and therefore these breakpoints are the same for τ_{ij} and β_{ij} (Ichoua et al., 2003).

Figure 1 presents an example of the different functions involved in the battery discharge model. Consider an arc $(i, j) \in A$ with a distance $d_{ij} = 2$ and a speed function v_{ij} as specified in Figure 1a. Furthermore, the fleet of EVs have batteries with an instantaneous consumption function $P(v) = \frac{1}{6}v^3 + \frac{1}{3}v$ (Figure 1c). Then, the resulting travel time function τ_{ij} and battery discharge function β_{ij} are illustrated in Figures 1b and 1d, respectively.

2.5 Battery consumption and feasible routes.

Having defined the battery consumption model at an edge level, we next proceed to extend these ideas at a route level. This represents a key concept within our model, as the feasibility of a route depends on maintaining a positive battery level during the entire trip, a definition that so far we have eluded. To illustrate the complexity introduced, consider a path $p = (o = v_1, \dots, v_k)$ starting at the depot. Let also t be a potential *ready time* at vertex v_k, that is, a time instant at which the vehicle has visited (processed) all the vertices in p and is ready to depart to the next vertex (or, alternatively, finish the trip if $v_k = d$). Note that both t and the battery level at t depend not only on the sequence of vertices in p and the departure time form the depot, but also on the other interdependent decisions taken along the traversal of p. The amount of battery recharged at an intermediate stop depends on the consumption of the previous visits, which in turn depends on the travel speeds of the previous edges. Furthermore, the delayed departures discussed earlier, which are a decision variable, can also affect the battery level at a ready time t. Indeed, different battery levels could be feasible for a given time instant t resulting from different previous decisions regarding these factors while traversing p.

Note that the battery level, including both the consumption and the recharges, does not affect the cost of a route (and, therefore, the overall solution) as it only restricts its feasibility. Then, given a path p and a time t, we focus on the combinations of decisions that lead to the maximum battery level when p is ready at time t, discarding other dominated solutions. In what follows, we introduce some additional notation and definitions to capture this intuitive idea. We develop the model considering a partial-recharge policy, although it can be easily adapted to other contexts. Similarly to the classical time-independent variants, we assume that the service time s_i for vertex i is already encoded in to the travel time function $\tau_{ij}(t)$. Specific details are provided in Section 3. We first consider a simplified definition for a given arc or vertex, and then we generalize the idea for paths.

Suppose we are given a continuous and piecewise linear function λ such that $\lambda(t)$ indicates the *maximum battery level* of a vehicle that is ready to depart a vertex i at a time t. Moreover, suppose the domain of λ is a closed interval $\mathrm{dom}(\lambda)$. When traveling through an arc $(i, j) \in A$ and departing from i at time t, the battery consumed is given by $\beta_{ij}(t)$ and, therefore, the battery level when arriving at j is by $\lambda(t) - \beta_{ij}(t)$. However, this need not be the maximum battery level achievable to visit j at time $t + \tau_{ij}(t)$ when traveling directly from i. Indeed, the vehicle may have arrived at j at a time t' earlier than t, with more charge in the battery, and waited until time t (which does not affect the battery level). We define the function $TRV^{\lambda}_{ij}(t)$ to capture this behavior. That is, $TRV^{\lambda}_{ij}(t)$ denotes the *maximum battery level* when arriving to j at a time t after traversing the arc $(i, j) \in A$, given that the maximum battery level function at i is λ. For each time t, let $H(t)$ be the set of times t' such that j is reached at time t or earlier when i is departed from at time t'. That is, $H(t) = \{t' \in \mathrm{dom}(\lambda) \mid \max\{a_j, t' + \tau_{ij}(t')\} \leq t\}$. Then, $TRV^{\lambda}_{ij}(t)$ is defined by

$$TRV^{\lambda}_{ij}(t) = \max_{t' \in H(t)} \{\lambda(t') - \beta_{ij}(t')\}. \tag{5}$$

Recall that functions λ, β_{ij} and τ_{ij} are continuous and piecewise linear, and τ_{ij} also satisfies the FIFO property. Thus, $\lambda - \beta_{ij}$ and $t \to \max\{a_j, t + \tau_{ij}(t)\}$ are also continuous and piecewise linear. By definition, $H(t') \subseteq H(t)$ for $t' \leq t$, thus $H(t)$ is a closed interval $[a(t), b(t)]$ because τ_{ij} satisfies the FIFO property. Note that $a(t)$ is a constant function, whereas $b(t)$ is continuous and piecewise linear, hence the domain $\mathrm{dom}(TRV^{\lambda}_{ij})$ of TRV^{λ}_{ij} is a closed interval $[t_1, t_2]$ as well. For the sake of simplicity, we abuse notation and define a restricted domain considering arrival times at j such that the maximum battery level is non-negative, i.e. $\mathrm{dom}(TRV^{\lambda}_{ij}) = \{t \in [t_1, t_2] : TRV^{\lambda}_{ij}(t) \geq 0\}$, that still remains a closed interval. This definition removes the infeasible arrival times at j that cannot be reached because the battery is depleted. Note that $TRV^{\lambda}_{ij}(t)$ implicitly encodes waiting times incurred by eventually delaying the departure from j, as it considers feasible ready times earlier than t.

A similar analysis holds for the stops at the recharging stations. Once again, suppose we are given a continuous and piecewise linear function μ such that $\mu(t)$ indicates the maximum battery level of a vehicle that is ready to start a recharge at time t in a station $i \in V_s$. Again, μ is defined over a closed interval $\mathrm{dom}(\mu)$. Let $CHG^{\mu}_{i}(t)$ be the *maximum battery level* when the recharge is completed at time t, defined as

$$CHG^{\mu}_{i}(t) = \max_{t' \leq t, t' \in \mathrm{dom}(\mu)} \{\mu(t') + g_i(\mu(t'), t - t')\}. \tag{6}$$

Again, as different battery levels are feasible for a given t, $CHG^{\mu}_{i}(t)$ indicates the maximum. Since recharging has no cost, it is always convenient to charge instead of waiting (without recharging). Similarly to the previous case, the function $CHG^{\mu}_{i}(t)$ also captures possible delays incurred by waiting times, while the domain $\mathrm{dom}(CHG^{\mu}_{i})$ of CHG^{μ}_{i} is equal to the closed interval $\mathrm{dom}(\mu)$.

Both TRV_{ij}^{λ} and CHG_i^{μ} are defined in terms of piecewise linear functions, although their structure is slightly more complex than a straightforward composition. Based on the previous approach, the function G in the following result generalize the structure of TRV_{ij}^{λ} and CHG_i^{μ}. The detailed proof can be retrieved from Appendix A.

Proposition 2. *Let f be a continuous and piecewise linear function, and b be a continuous and non-decreasing function. Define a function G on a domain $[x, y]$ as*

$$G(t) = \max_{t' \in [b(x), b(t)]} f(t'), \tag{7}$$

where $[b(x), b(y)] \subseteq dom(f)$. Then, G is continuous, piecewise linear, and non-decreasing.

Proposition 2 is relevant from a practical standpoint, as it enables to compute TRV_{ij}^{λ} and CHG_i^{μ} considering only a finite subset of breakpoints. We summarize this in the following results; note that b is piecewise linear in the definitions of TRV and CHG

Corollary 1. *Let $|G|$ and $|f|$ be the number of pieces of G and f in $[x, y]$ and $[b(x), b(y)]$, respectively. Then, $|G| \leq |f|$.*

Corollary 2. *If b is piecewise linear, then G can be computed in polynomial time. If the pieces of f and b are sorted, then G can be computed in linear time.*

We now extend these ideas to paths, in order to compute the battery level at any given moment of a trip. Let $p = (o = v_1, v_2, \ldots, v_k)$ be a path starting at the depot o, but not necessarily ending at d. Define $\lambda_{v_i}^p(t)$ as the maximum battery level at a ready time t when visiting vertex v_i along path p, $i = 1, \ldots, k$. Recall that t is a ready time when v_i can be departed from at time t, meaning that v_i has already been "processed". Certainly, each $t \in [0, T]$ is a ready time for the depot v_1. Thus, as each vehicle departs from v_1 with a full battery level, it follows that $\lambda_{v_1}^p(t) = B$ for every $t \in [0, T]$. Clearly, $\lambda_{v_1}^p$ is a continuous and piecewise linear function defined over a closed interval. To define $\lambda_{v_i}^p$ we proceed by induction on i, taking advantage of the functions TRV and CHG. Let $\lambda = \lambda_{v_{i-1}}^p$, i.e., $\lambda(t)$ is the maximum battery level when vertex v_{i-1} is ready at time t. If v_i is customer, then each arrival time is a ready time, thus $\lambda_{v_i}^p(t) = TRV_{v_{i-1}v_i}^{\lambda}(t)$. Recall that $\lambda_{v_i}^p$ is a continuous and piecewise linear function defined on a closed interval, as it is required by the induction. If v_i is a recharging station, then v_i can be departed from only after the battery is recharged. Thus, we have to consider the time required to travel from v_{i-1} to v_i plus the time to recharge at v_i. Consequently, $\lambda_{v_i}^p(t) = CHG_{v_i}^{\mu}(t)$ where μ is the function such that $\mu(t) = TRV_{v_{i-1}v_i}^{\lambda}(t)$. Once again, $\lambda_{v_i}^p(t)$ is a continuous and piecewise linear function defined on a closed interval.

The functions $\lambda_{v_i}^p$ are used to encode the battery level within our labeling algorithm, as discussed in Section 4.3. We further provide a numerical example illustrating these definitions in Appendix B.

3 Preprocessing techniques

We present in this section the four preprocessing rules applied to the instance, including a new rule specially designed for the TDEVRPTW.

3.1 Handling service times

Similar to time-independent problems, the service time s_i of a customer $i \in V_c$ can be directly encoded into the travel time and battery discharge functions to simplify the model and definitions. The following equations describe the transformations required to preprocess the service time in our model. Given an arc $(i, j) \in A$, the travel time and the battery discharge functions are updated as follows:

$$\tau_{ij}(t) := s_i + \tau_{ij}(t + s_i) \tag{8}$$

$$\beta_{ij}(t) := \beta_{ij}(t + s_i). \tag{9}$$

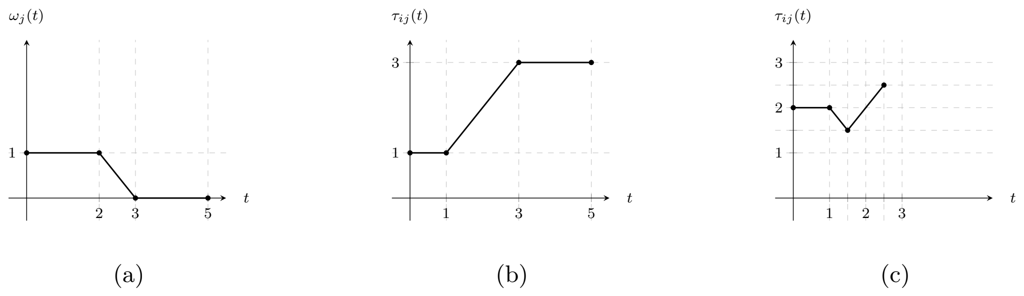


Figure 2: Example for the preprocessed time-dependent waiting times including waiting time function $\omega_j(t)$ of $j \in V_s$ (Figure 2a); $\tau_{ij}(t)$ of arc $(i, j) \in A$ (Figure 2b); $\tau_{ij}(t)$ of arc (i, j) after (10) (Figure 2c)

This preprocessing assumes that the final depot s_d has no service time ($s_d = 0$). Moreover, note that the travel time functions still obey the FIFO property.

3.2 Handling charging waiting times

Recall that each station $j \in V_s$ has a continuous piecewise linear waiting-time function $\omega_j(t)$ indicating the time the driver must wait before a charging terminal is available if arriving at the station at time t, as suggested by Keskin et al. (2019). Time-dependent travel times can also naturally incorporate these waiting times by following a similar procedure than the one described for service times. Given a vertex $j \in V$, then the travel time function for arc $(i, j) \in A$ is redefined as

$$\tau_{ij}(t) := \tau_{ij}(t) + \omega_j(t + \tau_{ij}(t)). \tag{10}$$

Observe that this transformation assumes there is no distinction between travel and waiting times regarding the objective function. Otherwise, the waiting time functions may require to be handled explicitly. A key property behind the time-dependent travel time model lies in the FIFO condition. The following proposition states that this transformation does not modify such condition. The proof is omitted as it is the direct consequence of adding and composing continuous piecewise linear functions that satisfy the FIFO condition.

Proposition 3. *The FIFO property holds for the travel time functions after applying (10).*

However, a negative result is that the breakpoints defining τ_{ij} and β_{ij} may not remain the same after applying this preprocessing, as stated in Section 2.4.

We provide a numerical example to illustrate this transformation in Figure 2. Let $j \in V_s$ be a station having waiting time function ω_j (Figure 2a). Consider an arc $(i, j) \in A$, and let τ_{ij} be its travel time function (Figure 2b). The preprocessed function τ_{ij} is depicted in Figure 2c.

Finally, we remark that a similar analysis can be developed to model time-dependent service times via continuous piecewise linear functions. Although we do not consider explicitly this scenario, they can be incorporated in a similar fashion to the framework proposed. This may become relevant in practice, for instance, to model variable unloading times due to difficulties in finding parking spots during peak hours or similar use cases arising frequently in this context.

3.3 Reducing arcs and time windows

Two interrelated steps are applied that search for infeasible arcs and shrink time windows, respectively. These steps are repeated while there are changes in the instance structure. For this procedure we follow the ideas of Lera-Romero and Miranda-Bront (2019) that extend the classical rules proposed by Desrosiers et al. (1995). Briefly, an arc $(i, j) \in A$ is removed if is either capacity infeasible, i.e. $q_i + q_j > Q$, or time infeasible, i.e. $a_i + \tau_{ij}(a_i) > b_j$. We omit the time window shrinking rules for the sake of brevity, but refer to Desrosiers et al. (1995) for a detailed description.

3.4 Minimum battery required

We next introduce a preprocessing rule specifically designed for the TDEVRPTW. For each vertex $i \in V$ we can compute a lower bound $MBR(i)$ of the *minimum battery required* to reach any recharge station (or eventually the depot) from vertex i without depleting the battery. Let $\underline{\beta}_{ij} = \min\{\beta_{ij}(t) \mid t \in \text{dom}(\beta_{ij})\}$ be a lower bound on the energy consumption for arc $(i,j) \in A$. Then, for $i \in V$, a lower bound $MBR(i)$ can be computed via a standard shortest-path algorithm using $\underline{\beta_{ij}}$ as arc weights. Note that $MBR(i) = 0$ for $i \in V \setminus V_c$. In addition, arcs $(i,j) \in A$ satisfying $B - \underline{\beta}_{ij} < MBR(j)$ indicate that the battery level when reaching j cannot be enough to either reach another station or the depot, and therefore can be safely discarded. These bounds are used further to enhance the feasibility rules of the labeling algorithm to reduce the size of the enumeration tree, as described in Section 4.3.

4 Exact algorithm

This section describes the main components of the exact BCP algorithm. First, we introduce the classical set-partitioning formulation for the VRP. Second, we present the two branching schemes, and propose a new branching rule specifically designed for EVRP and similar problems. Then, we present a mono-directional labeling algorithm, including a brief analysis regarding the impact of time-dependency. Finally, we summarize how to adapt the framework to toggle the different variants described in Section 2.3. We highlight that our algorithm extends and generalizes the one proposed by Desaulniers et al. (2016) and constitutes a unifying framework for several interesting problems related to planning with EVs.

4.1 Set-partitioning formulation

BCP algorithms and extended formulations based on the set-partitioning model stand as one of the most effective approaches to tackle different variants of the VRP. Let Ω be the set of all the feasible routes for the TDEVRPTW. For each route $r \in \Omega$, c_r represents its cost and the constant a_{ir} indicates if route r visits customer $i \in V$. Let y_r be a binary variable indicating whether a route $r \in \Omega$ is selected in the optimal solution. The set-partitioning formulation for the TDEVRPTW is defined as follows.

$$\min \sum_{r \in \Omega} c_r y_r \tag{11}$$

$$\text{s.t.} \sum_{r \in \Omega} a_{ir} y_r = 1, i \in V_c \tag{12}$$

$$y_r \in \{0,1\}, r \in \Omega. \tag{13}$$

The objective function (11) minimizes the total costs of the solution, whereas restrictions (12) enforce that each customer is visited exactly once and (13) impose the domain of the variables.

Considering that Ω has an exponential number of routes, the classical approach is to solve the LP relaxation at each node by using column generation. A restricted master problem (RMP) is initialized with a subset of the routes $\Omega' \subseteq \Omega$. For the TDEVRPTW, the initialization of the RMP is slightly more complicated compared to other variants as the traditional strategy selecting the routes $r = \{o, i, d\}$ to define Ω' may be infeasible due to the battery consumption. To overcome this issue, we initially include in Ω' an artificial (infeasible) route r that visits all customers with cost $c_r = \infty$ to guarantee the feasibility of the LP relaxation of the RMP.

After the initialization step, new columns are added iteratively until the algorithm converges to the optimal (fractional) solution. At each iteration, the LP relaxation of the RMP is computed to obtain the dual variables π_i associated with constraints (12) for $i \in V_c$. Using this information, if a feasible route r with negative reduced cost $\bar{c}_r = c_r - \sum_{i \in r} \pi_i$ exists, then it is added to the RMP and the procedure is repeated. Otherwise, the current fractional solution is optimal. For the TDEVRPTW, the pricing problem becomes

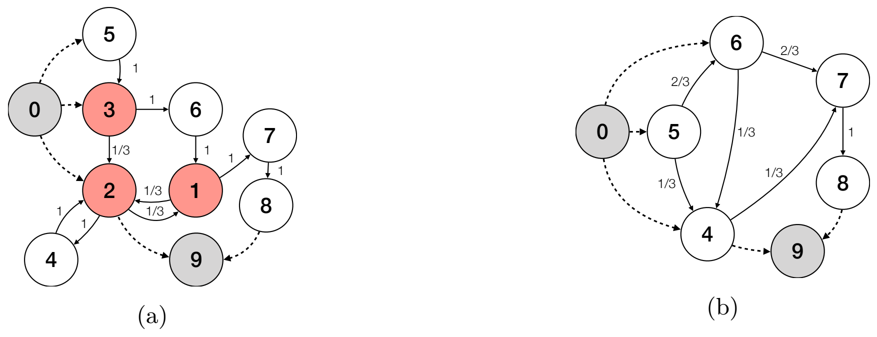


Figure 3: Example for arc variables x_{ij} (ABR, 3a) and customer variables (CBR, 3b) for a solutions with routes r_1, r_2, r_3 and r_4.

a Time-dependent Electric Elementary Shortest Path Problem with Resource Constraints (TDEESPPRC). The most effective exact approaches to tackle similar problems rely on dynamic programming and labeling algorithms (Desaulniers et al. (2016)). Heuristics are in general used to speedup the performance of the pricing step, and exact algorithms are executed when no negative reduced cost routes are found to guarantee optimality (see Section 4.3.4).

4.2 Branching scheme

In this section we review the standard ideas considered for the EVRPTW, and propose a tailored branching rule to manage the intermediate stops at the recharging stations. To the best of our knowledge, this branching rule is new and has not been proposed previously in the literature.

4.2.1 Arc branching rule

Classical BCP algorithms consider branching on the so-called *arc variables*. Given an optimal solution y^* of the LP relaxation for (11)–(13), define for each arc $(i, j) \in A$ a binary variable $x^*_{ij} = \sum_{r \in \Omega, (i,j) \in r} y^*_r$. When no intermediate stops between customers are possible, such as in the VRP, it can be easily shown that y^* is fractional iff x^* is fractional as well. Therefore, branching consists in finding a variable x_{ij} with non-integral value and opening two branches setting its value to 0 or 1. From an implementation standpoint, adopting this branching rule is beneficial as it is *robust* and does not change the structure of the pricing problem after branching. Branches $x_{ij} = 0$ and $x_{ij} = 1$ can be enforced by simply removing edges from the graph G conveniently (see, e.g. Costa et al. (2019)). We refer to this rule as the Arc Branching Rule (ABR).

Under the presence of recharging stations, variables x_{ij} must be redefined to consider integer values for the recharging stations $j \in V_s$ that can be visited more than once. In this case, even if all the variables x^*_{ij} corresponding to arcs incident to a customer vertex are binary, the solution y^* of the RMP may still remain fractional. An arc $(i, j) \in A$ with fractional cost x^*_{ij} must be selected and two nodes must be created with additional constraints on the LP of the form $x_{ij} \geq \lceil x^*_{ij} \rceil$ and $x_{ij} \leq \lfloor x^*_{ij} \rfloor$. Adding these constraints introduce new dual variables that can be robustly incorporated into the arc costs to maintain the same pricing problem structure. However, as opposed to binary variables, arc (i, j) can not be removed from the graph, which may impact negatively on the efficiency.

We illustrate the above situation in Figure 3a on an instance with depots 0 and 9, recharging stations $\{1, 2, 3\}$ and customers $\{4, \ldots, 8\}$ where a feasible fractional solution composed of four routes $r_1 = (0, 5, 3, 6, 1, 7, 8, 9)$, $r_2 = (0, 5, 3, 2, 4, 2, 9)$, $r_3 = (0, 3, 6, 1, 2, 4, 2, 9)$, $r_4 = (0, 2, 4, 2, 1, 7, 8, 9)$, with variable values $y_1 = \frac{2}{3}$ and $y_2 = y_3 = y_4 = \frac{1}{3}$ is shown. As observed, all arcs incident to a customer have integer values yet the solution is fractional.

4.2.2 Customer branching rule

Formulation (11)–(13) encodes the feasibility of a route, including the visits to the recharging stations, as part of the definition of Ω. Note that constraints (12) are enforced only for customer vertices $i \in V_c$, and indeed Ω can be restricted to consider at most one minimum-cost route for each set of customer vertices $S \subseteq V_c$, since all other routes visiting the same customers become dominated. This can be easily handled in the implementation, and in fact represents a key characteristic for our branching. From now on, we assume Ω satisfies this property.

Based on this observation, we propose an alternative branching rule exploiting the sequence of customers visited by a route, which we name Customer Branching Rule (CBR). Let $r = (0 = v_1, \ldots, v_k = d) \in \Omega$ be a feasible route. We define the *customer sequence of* r, $CR(r)$, as the sequence of customer vertices in r, removing those vertices $v_i \in V_s$. Given $i, j \in V_c$ and $r \in \Omega$, let b_{ijr} be a constant indicating whether j is visited immediately after i in $CR(r)$. Intuitively, $b_{ijr} = 1$ if there are no customers between i and j in route r. Let further define binary variables z_{ij} indicating if vertex j is visited immediately after i in any route $r \in \Omega$ disregarding recharge stations, that can be retrieved from the primal solution y as follows

$$z_{ij} = \sum_{r \in \Omega} b_{ijr} y_r. \tag{14}$$

We propose the Customer Branching Rule (CBR), based on variables z_{ij}. Given a fractional variable z_{ij} we create two descending branches, one for $z_{ij} = 0$ and the other one for $z_{ij} = 1$. The following proposition proves that CBR is a valid branching rule. The detailed proof can be found in Appendix C.

Proposition 4. *Let $(\bar{y}, \bar{z})$ be a solution of formulation (11)–(13) extended with variables z from (14). Then, $\bar{y}$ is a feasible solution for the TDEVRPTW if and only if $\bar{z}_{ij} \in \{0, 1\}$ for all $i, j \in V_c$.*

From an implementation standpoint, the CBR can be handled in a similar fashion as the ABR. Constraint $z_{ij} = 1$ can be replaced by the union of constraints $z_{ik} = 0$ for $k \in V_c \setminus \{j\}$, and $z_{kj} = 0$ for $k \in V_c \setminus \{i\}$. This is particularly helpful when adapting the pricing problem to handle this branching strategy as explained in Section 4.3.2. Finally, Figure 3b depicts the variables z_{ij} associated to the solution presented for Figure 3a.

4.2.3 Other branching alternatives

Generally, the ABR is combined with some additional rules to further reduce the enumeration tree. For the EVRPTW, Desaulniers et al. (2016) propose a four-level branching strategy. Firstly, they branch on the number of vehicles $\sum_{r \in \Omega} y_r$. Then, branching is performed on the total number of recharges $\sum_{r \in \Omega} \rho_r y_r$, where ρ_r indicates the number of recharges performed in the traversal of route r. The next decision is based on the number of recharges at a given station $\sum_{r \in \Omega} \rho_{rj} y_r$, where ρ_{rj} indicates the number of recharges of route r at station $j \in V_f$. Finally, ABR is considered by variables x_{ij} if none of the above are fractional. We remark that these rules can be combined with the CBR as well.

4.3 The pricing problem

This section describes the labeling algorithm developed to tackle the pricing problem for Section 4.1, including the adaptations needed to handle the different variants from Section 2.3 as well as the CBR branching rule from Section 4.3.2. Additionally, we provide a brief discussion regarding the impact of considering alternative objective functions.

4.3.1 The labeling algorithm

We develop a forward labeling algorithm that explores all feasible paths. These algorithms generate implicitly an enumeration tree where each node is called a label and represents a partial path starting from the initial depot o. Aiming to overcome the exponential growth, pruning rules are in general incorporated to reduce the number of labels enumerated. In a time-independent context, each label is usually either discarded or processed. However, time-dependent problems require further considerations since labels encode some of their resource values as functions over a given domain. Thus, partial pruning rules eventually discard some specific intervals in the domain (see, e.g. Lera-Romero et al. (2020a)).

Next we present the forward labeling algorithm for the TDEVRPTW with a partial-recharge policy and unlimited recharges. Each path $p = (o, \dots, v)$ is represented by a label $L = (prev, v, S, q, c, \lambda^p_v)$ where $prev$ is a pointer to the parent label, v is the last vertex, S is a set of unreachable customers, q is the total demand, c is the cost, and λ^p_v is the maximum battery level function defined in Section 2.5, i.e., $\lambda^p_v(t)$ is the maximum battery level at the ready time t after traversing p. For simplicity, let $prev(L)$, $v(L)$, $S(L)$, $q(L)$, $c(L)$, and λ_L denote the components $prev$, v, S, q, c, and λ^v_p of label L, respectively. In addition, let $p(L)$ be the path represented by L and let $\text{dom}(L) = \text{dom}(\lambda_L) = [a(L), b(L)]$ denote the feasible ready times at $v(L)$ after traversing p, with $a(L) = \min(\text{dom}(L))$ being the earliest arrival time to vertex $v(L)$. When $p(L)$ is a route we call L a *route label*.

Let $\bar{c}_{ij} = c_{ij} - \pi_i$ be the adjusted arc costs including the dual variables from the RMP. Since vehicles depart from the depot with a full battery, the initial label is represented by tuple $(\perp, o, \{o\}, 0, 0, 0, \lambda_o)$, where $\lambda_o(t) = B$ for $t \in [a_o, b_o]$. Given a label L, the extension through an arc $(v(L), w)$ is feasible when $w \notin S(L)$, $q(L) + q_w \leq Q$ and $a(L) + \tau_{vw}(a(L)) \leq b_w$, resulting in a new label $L_w = (L, w, S(L) \cup \{w\}, q(L) + q_w, c(L) + \bar{c}_{vw}, \lambda_w)$. The function λ_{L_w} is computed from λ_L as discussed in Section 2.5.

Battery requirements enable to introduce additional tailored feasibility rules. Recall from Section 3 that given a vertex $v \in V$, $MBR(v)$ indicates the minimum battery required to depart from a vertex v and reach a station or, alternatively, the end depot without depleting the battery. The following rule exploits this information to discard some intervals from $\text{dom}(L)$.

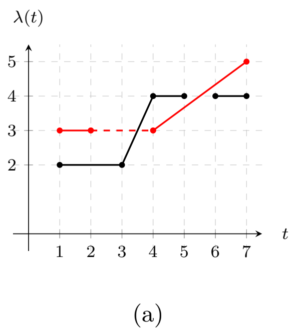
Rule 1 (Feasibility)**.** *Given a label L, any time $t \in dom(L)$ such that $\lambda_L(t) < MBR(v(L))$ can be safely discarded from λ_L. If $dom(\lambda_L) = \emptyset$ then $p(L)$ is* infeasible *and L can be discarded.*

Since function λ_L is non-decreasing by definition, any interval discarded from $\text{dom}(\lambda_L)$ must be a prefix. This fact can reduce the time required to compute this rule for many labels.

Another action to reduce the number of labels enumerated relies in applying dominance rules. Briefly, given two labels L, M, if every feasible extension p' of $p(L)$ is also a feasible extension of $p(M)$ but $p(M) + p'$ always has a lower cost, then L can be discarded. In practice, to reduce the computational burden, sufficient conditions are tested to ensure that a label can be discarded. In a time-dependent context, partial dominance enables to discard some specific dominated time instants from the domain $\text{dom}(L)$, even if the label is not entirely dominated. As a result, the domain of a label may become a collection of intervals instead of a single one. However, this can be mitigated by assuming the vehicle simply waits without consuming battery in this dominated interval, which is modelled by considering a constant function extending the function from the previous non-dominated piece (we refer to Lera-Romero et al. (2020a) for details). To avoid introducing further definitions, we abuse notation and assume the battery level functions λ_L and λ_M, as well as their corresponding domains, are adjusted in this fashion in the following rule.

Rule 2 (Partial dominance)**.** *Let L and M be two labels satisfying (i) $v(L) = v(M)$, (ii) $q(M) \leq q(L)$, (iii) $c(M) \leq c(L)$, and (iv) $S(M) \subseteq S(L)$. Then, any time $t \in dom(\lambda_L) \cap dom(\lambda_M)$ such that (v) $\lambda_L(t) \leq \lambda_M(t)$ can be discarded from $dom(L)$. If $dom(\lambda_L) = \emptyset$ then we say L is* fully dominated *and can be safely discarded.*

From an implementation standpoint, to accelerate the dominance tests we consider the classical bucket structure where a dynamic programming table stores labels indexed by their resources v, q. Within each bucket labels are sorted by cost. Observe that dominance tests include a comparison of the battery level functions between two labels, which requires linear time with respect to the total number of pieces in both functions. For the sake of brevity, we skip the implementation details. Figure 4 shows an example of the partial dominance algorithm when label L is being verified for dominance against label M, where the dashed line represents the constant extension over a dominated interval in the domain of label M. Figure 4b shows the remaining pieces from L after the dominance.



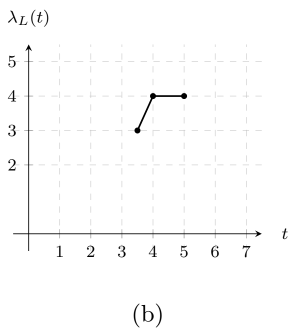


Figure 4: Example of the partial dominance applied to L (black) and M (red) (4a) and the remaining pieces (4b)

4.3.2 Incorporating CBR to the pricing algorithm

The CBR branching rule imposed over variables z_{ij} can be managed within the pricing algorithm in a similar fashion as the branching over the classical arc variables x_{ij} by preventing such routes from being generated instead of adding explicitly the constraints $z_{ij} = 0$.

At each node of the branch and bound tree, define $F_i = \{j \in V_c \cup \{d\} \mid z_{ij} = 0\}$ as the set of *forbidden successors* for each customer $i \in V_c \cup \{o, d\}$ (and let $F_i = \emptyset$ for each $i \in V_s$). Intuitively, F_i encodes the set of successors of i that cannot be visited due to branching decisions taken earlier in the tree. We include in the representation of a label L an additional resource u indicating the last vertex from $V_c \cup \{o, d\}$ visited in $p(L)$, i.e. either a customer or the depot. The extension of L through an arc $(v(L), w) \in A$ must additionally verify that $w \notin F_{u(L)}$. Consequently, Rule 2 must be modified accordingly to remain valid, where $F_{u(M)} \subseteq F_{u(L)}$ must also hold in order to conclude that label M dominates label L at time t. This new condition ensures that every feasible extension of L does not involve $w \in F_{u(M)}$ as the next vertex, and therefore is also a feasible extension of M. We further strengthen this condition by considering the set of already visited customers $S(L)$ as follows

$$F_{u(M)} \subseteq F_{u(L)} \cup S(L). \tag{15}$$

Incorporating condition (15) to Rule 2 affects the efficiency of the labeling algorithm. Similarly to the ABR for VRPs without intermediate stops between customers, label extensions are discarded because they may become infeasible. On the contrary, dominance rules become weaker due to the introduction of a new condition. Clearly, this implies that more labels are enumerated, although we remark that the effect of condition (15) is limited to only those labels L that end in a recharging station, otherwise $v(L) = v(M) = u(L) = u(M)$ and the inclusion is trivially satisfied.

4.3.3 Alternative recharge policies

So far, the battery consumption model and the labeling algorithm have been presented under a partial-recharge policy. Fortunately, adapting the model to consider a full-recharge policy is straightforward. Figure 6e illustrates the battery level function after applying the charging step. Observe that there is a prefix of the domain $P = [\min(\text{dom}(\lambda_j^p)), t_2] \subseteq \text{dom}(\lambda_j^p)$ where $\lambda_j^p(t) < B = 10$ for all $t \in P$. Under a full-recharge policy, it is infeasible to depart from a recharge station $j \in V_s$ with that battery level.

Formally, let p be a path ending at a station $j \in V_s$, and let λ_j^p be its battery level function at vertex j. Then, all time instants $t \in \text{dom}(\lambda_j^p)$ such that $\lambda_j^p(t) < B$ must be discarded. Observe that a full-recharge policy can be enforced by setting $MBR(j) = B$ for all recharging stations $j \in V_s$ since Rule 1 removes any time from the domain under this value.

Another variant introduced by Desaulniers et al. (2016) is the possibility to restrict the number of in-route recharges. Given a maximum number of recharges per route R_{max}, extend the definition of a label L of to consider a new resource $r(L)$ indicating the number of recharges performed, and forbid an extension to a recharging station if $r(L) \geq R_{max}$. Additionally, the dominance rule 2 must be modified by including condition $r(M) \leq r(L)$, which enforces that all extensions of L are feasible for M in terms of number of recharges.

4.3.4 Pricing heuristics

The execution of the column generation is usually accelerated by resorting to heuristics in order to identify columns with a negative reduced cost, while the exact labeling algorithm is executed if no such column is identified. One type of heuristic adaptation frequently used for labeling algorithms is to reduce the size of the graph. At the beginning of each pricing iteration, the graph is reduced to consider only the k outbound arcs $(i, j) \in A$ from each vertex $i \in V$ with the smallest reduced cost $\bar{c}_{ij}$. We refer to this approach as the k-shrink heuristic.

We further consider relaxing the dominance Rule 2 within the pricing algorithm, maintaining the feasibility of the solutions but eventually discarding routes with a negative reduced cost due to a weaker dominance criterion. We refer to the heuristic where condition (iv) is relaxed from Rule 2 as *relax-S*, while the heuristic relaxing condition (v) (i.e., the battery level) is referred as *relax-B*.

4.3.5 Alternative objective functions

Although the TDEVRPTW minimizes the total cost of the routes, we include a brief discussion regarding the feasibility and the impact of considering further alternative objective functions.

In some time-dependent problems, such as the TDTSP (see, e.g. Cordeau et al. (2014); Arigliano et al. (2018); Adamo et al. (2019)), the objective is to minimize the makespan of the route for a vehicle that departs at the beginning of the planning horizon, i.e., at time $t_0 = 0$. Our approach can be easily adapted to consider such objective function. In this case, the reduced cost of a label L at time $t \in \text{dom}(L)$ is defined as $t - \sum_{i \in p(L)} \pi_i$, where π are the values of the dual variables. Within the pricing algorithm, setting the cost $c_{ij} = 0$ for all arcs $(i, j) \in A$ redefines $c(L)$ as the sum of the dual prices of path $p(L)$ for a label L. Note that the objective function does not affect the extension rules, that remain unchanged. However, dominance rules depend on the reduced cost of the route. For a label L, when reaching $v(L)$ at some time t the reduced cost is $t - c(L)$, and thus condition (iii) from Rule 2 effectively compares the reduced cost of two labels L and M at some fixed time t. Note that even when partial dominance can discard some specific intervals, the optimal route preserves its makespan in the domain.

A more complex objective function is considered for the TDVRPTW (see, e.g. Dabia et al. (2013); Lera-Romero et al. (2020a)) where vehicles are allowed to delay their departure from the depot in order to minimize the duration of the route. Unfortunately, handling this objective function within the BCP algorithm proposed for the TDEVRPTW introduces significant complexity compared to the TDVRPTW. The reason is that a new decision variable is added, which is the initial departing time is t_0. Given a path p, the battery level depends both on the completion time t and on the initial departing time t_0, and becomes a two dimensional function due to the delays both in the departure from the depot as well as from the departure from a vertex, as discussed in Section 2.2. Although the dominance procedure can be adapted, it would become computationally too expensive to store this type of functions within each label and perform the corresponding comparisons in the dominance step. Similarly, considering energy-dependent costs for the path during the trip would suffer from the same drawback. Note, however, that a fixed recharge cost c_j can be easily included for each recharge made at station $j \in V_s$ by defining $c_{ij} := c_{ij} + c_j$ for $(i, j) \in A$.

4.4 Cutting planes

We incorporate the Subset Row Inequalities (SRCs) proposed by Jepsen et al. (2008) as cutting planes to tighten the LP relaxation, aiming to reduce the size of the branching tree. These valid inequalities have become a standard within VRP exact algorithms based on the set-partitioning formulation (see, e.g. Costa et al. (2019)) and have also been used for the EVRPTW by Desaulniers et al. (2016). Formally, a SRC $s = (S, k)$ is defined by a set of customers $S \subseteq V_c$, and a coefficient $k \in \mathbb{N}$. Given a route $r \in \Omega$ and

$s = (S, k)$, let $v_{sr} = \left\lfloor \frac{|S \cap r|}{k} \right\rfloor$ be the number subsets in S having k vertices that are visited by r. Then, the SRC reads

$$\sum_{r \in \Omega} y_r v_{sr} \leq \left\lfloor \frac{|S|}{k} \right\rfloor. \tag{16}$$

Since the separation problem is NP-hard for the general case, a common practice is to maintain the size of S and k manageable for an exhaustive search. We consider the so-called *3-SRC*, i.e., having $|S| = 3$ and $k = 2$. Another important observation is that the SRCs are non-robust cuts, meaning that the pricing problem must be modified accordingly every time a new cut is added to the RMP by defining a new resource. Since these modifications are standard, we refer the reader to Costa et al. (2019) and omit the details. However, it is worth noting that there is a compromise between the number of cuts considered and the efficiency of the pricing algorithms.

Our cutting plane algorithm is the following. After solving the LP relaxation via column generation, a maximum of n_{cuts} violated inequalities are iteratively added to the formulation following a maximum violation criterion, reoptimizing the RMP at each step without solving the pricing problem. Preliminary experiments show that this approach tends to reduce the number of cuts added to the RMP while preserving their impact in the quality of the LP relaxation.

5 Computational experiments

In this section we evaluate the time-dependent model proposed in Section 2 as well as the BCP algorithm developed in Section 4. For this purpose, we introduce a set of time-dependent benchmark instances, based on the ones proposed by Desaulniers et al. (2016), as described in Section 5.1. The experiments are designed to provide managerial insights regarding the importance of incorporating time-dependent travel times, and show a strong evidence about the effectiveness of our approach through extensive experiments with and without variable travel speeds. The experiments are conducted on an Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz with 32GB of RAM, and the algorithms are coded in C++ and use CPLEX 12.9 as an LP solver.

5.1 Experimental setup

In order to evaluate the different components of the model, we extend the instances proposed in Desaulniers et al. (2016) for the EVRPTW by incorporating time-dependent travel speeds, non-linear recharging times, and time-dependent waiting times as follows.

Regarding time-dependent travel speeds, we follow a similar strategy as Dabia et al. (2013). The planning horizon $[0, T = b_d]$ is partitioned into five speed zones $T_1 = [0, 0.2T], T_2 = [0.2T, 0.4T], T_3 = [0.4T, 0.6T], T_4 = [0.6T, 0.8T], T_5 = [0.8T, 1.0T]$. Each arc $(i, j) \in A$ is randomly assigned to one of three speed profiles representing different traffic congestion levels: *high*, *normal*, and *low*. Figure 5a shows the speed function v_p assigned to each speed profile. To enable a meaningful qualitative comparisons, each speed function v_p has an average speed of 1.0 over the planning horizon, which is the travel speed considered in the time-independent instances. The parameter r required to define the instantaneous consumption function $P(v)$ is obtained from the original instances.

All instances are enhanced with piecewise linear recharging functions following the pattern proposed by Montoya et al. (2017). Recall that the original EVRPTW instances from Desaulniers et al. (2016) assume linear charging, where the full charge of the battery with capacity B takes t_g units of time, recharging B/t_g kWh per unit. We define the piecewise linear recharging functions where a full recharge of a battery with capacity B also requires t_g units of time as well, although the recharge per unit of time is not uniform along the interval (Figure 5b). In addition, each station belongs to one mode: *fast*, *medium*, and *slow*. The medium mode is defined according to B and t_g, while the slow and fast modes are a scaled version of the medium mode, $2t_g$ and $0.5t_g$, respectively.

Regarding the time-dependent waiting times at stations, Keskin et al. (2019) consider multiple scenarios that capture various real-life situations. As we do not focus exclusively on evaluating waiting times, instances

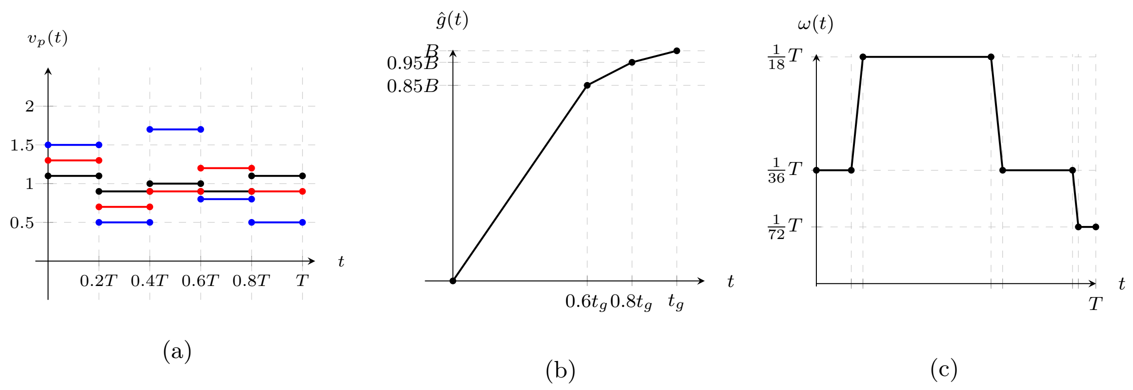


Figure 5: Description of the dataset, including the speed function $v_p(t)$ for each profile p high (blue), normal (red), low (black) (Figure 5a); a generic recharging function $\hat{g}$ from Montoya et al. (2017) (Figure 5b); and waiting time function $\omega(t)$ from Keskin et al. (2019) (Figure 5c)

are extended with the *TD-Smooth-Long* pattern, as it represents an intermediate scenario where stations are neither too congested nor empty. Figure 5c shows the function considered, which is scaled for each instance according to the planning horizon.

Note that the benchmark instances proposed by Desaulniers et al. (2016) can be retrieved by setting all the speed profiles to $v_p(t) = 1$, defining the recharging functions with a unique piece requiring the same total time as the medium charger to charge the battery completely, removing the waiting times by setting $\omega_s(t) = 0$ for every station $s \in V_s$. For the experiments, we define the following configurations for the instances:

- Basic: instances from Desaulniers et al. (2016) extended with time-dependent travel and discharge functions.
- NC: Basic extended with non-linear charging functions.
- WT: Basic extended with time-dependent waiting time functions.
- All: Basic extended with non-linear charging and time-dependent waiting time functions.

To evaluate the impact of the time-dependent travel speeds, we further consider the time-independent versions of the above benchmarks as indicated previously.

The BCP algorithm relies on several parameters, some of them dependent on the type of instances considered. Regarding the heuristic pricing, we execute a sequence of heuristics relax-S, relax-B and k-shrink described in Section 4.3.4 with different combinations of parameters. Based on limited preliminary experiments, the configuration is as a follows: relax-S first combined with k-shrink for $k = 3, 7$ and 12; relax-B; relax-S; and finally 7-shrink. Once a configuration fails, is not attempted again in future iterations. If all heuristics fail to find a column with a negative reduced cost, then the exact algorithm is executed. As for the cutting planes, we set $n_{max} = 200$.

The configuration of the algorithm is also affected by the recharge policy considered, whether if single (S) or multiple (M) recharges are allowed, and they are restricted to be full (F) or partial (P). Under SF and MF recharge policies, we set $MBR(j) = B$ for all $j \in V_s$. For SF and SP, we set $R_{max} = 1$ to limit the number of recharges in route to 1.

5.2 Impact of the TDEVRPTW

The time-dependent model presented in Section 2 includes more information and, therefore, is expected to produce better quality solutions. In this section, we evaluate and compare the solutions produced when considering time dependency to their time-independent counterparts.

			time independent				time dependent			
dataset	$\lvert V_c \rvert$	common	cost	makespan	#rech	#routes	cost (%)	makespan (%)	#rech (%)	#routes (%)
Basic	25	56	458.50	2879.88	1.09	4.11	-0.20	-0.05	6.82	-3.04
	50	43	702.85	4441.24	0.97	6.63	0.33	0.02	24.57	-0.70
	100	14	1251.98	5449.47	1.28	12.93	0.38	2.16	4.53	2.21
NC	25	56	457.03	2846.69	1.10	4.02	0.01	0.72	11.92	-2.22
	50	45	699.23	4256.58	1.08	6.51	0.55	0.84	14.13	0.00
	100	14	1188.15	6643.29	1.21	12.14	0.61	0.50	4.07	2.94
WT	25	56	462.79	2948.02	0.93	4.21	0.06	-0.01	5.31	-0.85
	50	43	723.78	4582.24	0.89	6.88	0.46	0.91	9.68	2.36
	100	13	1224.89	7364.81	0.90	13.54	0.65	0.89	7.74	-0.57
ALL	25	56	461.17	2906.51	0.94	4.11	0.15	0.09	5.23	-0.87
	50	44	717.97	4480.35	0.88	6.91	0.69	0.58	13.98	-0.99
	100	12	1238.16	6825.26	0.90	13.17	0.48	1.75	11.35	3.16

Table 2: Comparison of the quality of optimal solutions: time-independent vs time-dependent.

To evaluate the time-dependent model, we conduct experiments over the four datasets described in the previous section with and without time-dependent travel speeds. For the experiments, we consider the policy MP, with multiple stops and partial recharge, as it stands as the more complex and rich scenario. We evaluate the quality of the solution in each case over four metrics: total cost; sum of makespan of the routes; average number of recharges per route (#rech); and number of routes (#routes). Table 2 reports the results, disaggregated by dataset and averaged over each instance size only over the instances that are solved to optimality in both cases. Column $|V_c|$ indicates the number of customers, column *common* shows the number of instances solved to optimality. For the time-independent case, averages for each metric are reported in absolute terms and used as a baseline. Let z_{ti} and z_{td} be the value of a given metric for the time-independent and for the time-dependent case, respectively. For the time-dependent setup the values are reported as the relative difference with respect to the time-independent scenario, i.e. $(z_{ti} - z_{td})/z_{ti}$.

The main message of Table 2 is that incorporating the congestion at the planning level induces differences between the optimal solutions with respect to all metrics. For instance, the number of recharges per route grows more than a 10% on average and up to 24% for the time-dependent case, suggesting that simplifying travel speeds by averaging them impacts directly on the battery discharge model. Notice, however, that including time-dependency does not necessarily lead to higher cost solutions, as is the case of the 25 customer instances for dataset NC. In fact, considering congestion conditions can some times lead to faster routes, which are still feasible. We further highlight the differences in the number of routes, that may become meaningful from a managerial perspective.

By construction, the time-independent instances can be interpreted as the result of averaging the travel speeds from the time-dependent ones. To highlight the congestion effects, we propose a follow-up analysis by evaluating the solutions obtained for the time-independent instances under the time-dependent context. In this sense, we assume that the time-dependent instances represent a better approximation of the real-world operations, and the infeasiblities detected are the result of neglecting the time-dependency. We focus on two sources of infeasibilities. We say a route r is *time-infeasible* if it is infeasible even when setting the battery capacity to $B = \infty$. Intuitively, a time-infeasible route is infeasible either due to the violation of a time window or because its makespan exceeds the planning horizon. Similarly, we say a route r is *battery-infeasible* if all deadlines from time windows are removed by letting $b_i = T$ for all $i \in V$, and r is still infeasible due to the battery level. These infeasibilities can be further measured and quantified. We define the *time-violation* as the worst violation of a customer's time window divided by the horizon size T, and the *battery-violation* as the extra energy required to make the route feasible divided by the battery capacity B. Finally, we indicate count solution as infeasible if at least one of its routes is infeasible.

Table 3 reports the results of the infeasibility tests aggregated over the instance sizes for each dataset. Column #inst the number of instances solved to optimality in the time-independent context. The rest of the columns indicate the results over the metrics described above. Column #inf-s indicates the number of

dataset	$\vert V_c \vert$	#inst	#inf-s	%inf	%inf-t	%inf-b	%Δ-t	%Δ-b
Basic	25	56	47	50.22	30.49	23.32	1.47	0.76
	50	48	41	60.07	40.28	26.15	1.61	0.54
	100	17	17	82.16	58.92	34.05	3.61	0.88
NC	25	56	47	50.45	33.64	23.64	1.49	0.61
	50	48	42	57.34	37.20	25.94	1.76	0.60
	100	16	16	74.29	48.57	32.00	3.71	0.76
WT	25	56	46	50.43	28.21	25.21	1.57	0.67
	50	48	42	62.05	39.93	28.38	1.68	0.49
	100	18	18	87.43	54.86	41.14	3.60	1.12
ALL	25	56	48	50.44	27.19	25.88	1.54	0.61
	50	48	42	57.14	36.88	27.57	1.66	0.57
	100	15	15	74.23	47.24	36.81	3.46	1.05

Table 3: Feasibility analysis of the time-independent solutions evaluated in a time-dependent context.

infeasible solutions, and columns %inf, %inf-t and %inf-b represent the number of infeasible, time-infeasible and battery-infeasible routes, respectively, over the total number of routes, while %Δ-t and %Δ-b indicate the average violation of the time and battery-infeasible routes.

As observed, a large number of solutions from the time-independent model become infeasible when incorporating the time-dependent information. By inspecting the solutions, we note that on average more than 60% of routes are infeasible. As one may expect, a considerable percentage of these infeasibilities are due to a violation of the time windows because of the variations in the travel times, usually between 30 and 55% of the routes. However, a very interesting result lies that there is also a significant number of infeasibilities due to the EV running out of battery during the execution of the route. Such percentages range between 20 and 40%, depending on the scenario considered. From a managerial standpoint, these type of infeasibilities may result considerably expensive. Finally, we note that the average violations are rather small both for the time windows as well as for the battery level. This may be related to the parameters considered for the instances. Speed profiles with a higher variability as well as a different relation between h_1 and h_2 could lead to larger values.

5.3 The CBR branching rule

In this section, we evaluate the CBR within the framework. To reduce the potential impact of external factors, we conduct the experiments over the original time-independent instances from Desaulniers et al. (2016). Moreover, both CBR and ABR are tested on a pure BP (without the SRCs) in order better appreciate the impact regarding the branching strategy by enumerating a larger number of nodes in the branch and bound tree. An important observation is that the ABR can be implemented by removing of arcs from the support digraph since, for these instances, no arcs between two recharging stations exist. We consider such an implementation in this experiment, although we remark that this may result beneficial for the ABR computationally compared to the general case, as no additional constraints must included in the RMP and no edges are removed from the underlying graph.

The detailed results are reported in Table 5 in Appendix D. The main message is that using CBR in this context is beneficial, as BP-CBR is able to close 15 more instances than BP-ABR. The key behind this lies in the number of nodes enumerated, where CBR obtains reductions of about 50% percent on average. This effect is more evident on larger instances. However, we highlight that the average execution time does not decrease at this same rate. Recall that the pricing problem needs to be modified within the CBR, and some pricing iterations may become computationally more expensive due to this modification, while the ABR is being implemented by removing arcs.

			time independent				time dependent			
dataset	$\lvert V_c \rvert$	inst	opt	time	nodes	%rg	opt	time	nodes	%rg
Basic	25	56	56	7.05	1.04	0.00	56	25.13	1.14	0.01
	50	56	48	193.27	2.88	0.06	44	197.67	2.00	0.09
	100	56	17	775.94	11.82	0.27	17	822.30	10.88	0.34
NC	25	56	56	8.72	1.25	0.02	56	43.16	1.11	0.02
	50	56	48	350.40	2.83	0.06	47	291.80	1.51	0.05
	100	56	16	536.48	9.63	0.22	15	446.46	7.67	0.19
WT	25	56	56	5.64	1.21	0.02	56	21.84	1.75	0.05
	50	56	48	219.54	6.75	0.10	44	165.87	3.45	0.12
	100	56	18	907.41	20.00	0.43	14	727.61	16.86	0.70
ALL	25	56	56	6.77	1.21	0.01	56	41.76	1.04	0.00
	50	56	48	259.72	3.75	0.06	47	263.20	3.21	0.08
	100	56	15	530.57	14.87	0.76	13	489.25	14.54	0.54

Table 4: Performance metrics for the BCP across all variants, with and without time-dependent travel speeds and under a MP recharge policy.

5.4 Performance of the BCP

In this section, we report extensive computational experiments on the performance of the BCP algorithm to evaluate its effectiveness under different contexts and configurations. We begin by evaluating the BCP on the four configurations (Basic, NC, WT and All) for both time-dependent and time-independent versions for the MP recharge policy to maintain the number of experiments manageable. Again, we select this policy since it stands as the most challenging scenario. Table 4 reports the number of instances solved to optimality (opt), the average execution time (time), and the average number of nodes enumerated (nodes) and the average % gap at the root node (%rg) aggregated over each dataset and $|V_c|$. The columns time and nodes are averaged over the instances solved to optimality for each metric, where the column %rg is averaged over the instances where the root node was closed.

A first observation from the results in Table 4 is that the inclusion of the different variants has a limited impact on the performance of our algorithm. This is the expected behavior for our time-dependent framework, since the time-dependent waiting times directly encoded into the travel time functions and the non-linear charging functions are handled in the extension step of the labeling algorithm. Although this increases the difficulty, conceptually it does not introduce a new dimension to the problem. We remark, however, that the time-dependent instances appear to be slightly harder to solve than their time-independent counterparts, leaving all other variables fixed. The pricing problem becomes more difficult to solve in a time-dependent context, which is reasonable considering that the battery level function within each label has potentially more breakpoints. In fact, we observe that the BCP tends to enumerate less nodes in a time-dependent setup compared to its time-independent counterpart for comparable execution times. Finally, we note that the BCP produces remarkable gaps at the root node, below 1% on average, as a result of combining elementary routes with SRCs.

An additional experiment is conducted to provide further evidence regarding the performance of our framework. We establish a direct comparison between our BCP and the results reported by Desaulniers et al. (2016) over the original EVRPTW instances with all four recharge policies (SF, SP, MF, MP). Besides the TDEVRPTW, this is an interesting experiment itself. Although both algorithms consider a BCP framework, they incorporate different components. Briefly, Desaulniers et al. (2016) implements a bidirectional labeling algorithm, *ng*-route relaxations and a different cutting plane algorithm. We remark, however, that our framework includes a considerable overhead as it is more general, capable of handling piecewise linear functions.

The results are shown in Table 6 in Appendix D, where the key is the same as in the previous experiments. In this case, averages are computed over all instances solved to optimality by the corresponding method. A first analysis indicates that our BCP is competitive with the bidirectional algorithm from Desaulniers et al.

(2016). Indeed, we are able to obtain 13 new optimal solutions, although we remark that the computer we use is approximately a 20% faster. Besides the computation times, the remaining metrics suggest that some new components of the algorithm have a positive impact. The number of nodes enumerated remains moderated, particularly for instances with $n = 50, 100$. The root gaps share the same behaviour as they are smaller than %0.20 on average. We believe this is due to a combination of factors, namely considering elementary routes that lead to tighter lower bounds, the CBR branching and the enhanced preprocessing rules.

6 Conclusions

In this paper, we propose a general model for the TDEVRPTW that accounts for the effect of congestion on the discharge of the battery. Moreover, we develop a general framework that unifies and integrates several realistic characteristics considering some of the most relevant operational constraints for the EVRP from the literature, in general addressed independently. From a managerial standpoint, we show that neglecting the time-dependent travel speeds at the planning level can affect the quality of the routing plans, potentially incurring in violations of the time windows and exceeding the driving range. Our experiments suggest that up to 40% of the infeasibilities of the distribution plan can be caused by exceeding the battery capacity of the EVs. From an algorithmic perspective, we present a new BCP algorithm that generalizes existing methods for the EVRPTW that is capable of handling time-dependent waiting times, non-linear charging functions, time-dependent travel times, and battery discharge functions. We introduce a new branching rule derived to efficiently manage the intermediate stops between customers. Overall, the proposed algorithm shows to be very effective on a large set of instances from a wide spectrum of scenarios, solving instances with up to 100 customers to optimality and improving previous results from the literature for the EVRPTW.

As future research, we believe that the evaluation of the model on real data for the battery discharge coefficients and travel speed information is one of the most interesting directions. This would bring the model and the experiments closer to real-world operations. As for the modelling, there are still several open research lines that are worth considering. A similar approach could be followed to evaluate a load-dependent discharge model for the battery, which appears as another relevant parameter affecting the driving range. An integrated model considering both the load and the speed would be challenging algorithmically, but could lead to meaningful managerial insights.

Acknowledgements

This research has been founded by FONCyT grants PICT-2016-2677 and PICT-2018-2961 from the Ministry of Science, Argentina, and by the Google Latin America Research Award 2019.

References

Tommaso Adamo, Gianpaolo Ghiani, and Emanuela Guerriero. An enhanced lower bound for the time-dependent travelling salesman problem. *Comput. Oper. Res.*, 113:104795, 2019.

Anna Arigliano, Tobia Calogiuri, Gianpaolo Ghiani, and Emanuela Guerriero. A branch-and-bound algorithm for the time-dependent travelling salesman problem. *Networks*, 72(3):382–392, 2018.

Anna Arigliano, Gianpaolo Ghiani, Antonio Grieco, Emanuela Guerriero, and Isaac Plana. Time-dependent asymmetric traveling salesman problem with time windows: Properties and an exact algorithm. *Discrete Appl. Math.*, 261:28–39, 2019.

Jean-François Cordeau, Gianpaolo Ghiani, and Emanuela Guerriero. Analysis and branch-and-cut algorithm for the time-dependent travelling salesman problem. *Transport. Sci.*, 48(1):46–58, 2014.

Luciano Costa, Claudio Contardo, and Guy Desaulniers. Exact branch-price-and-cut algorithms for vehicle routing. *Transport. Sci.*, 53(4):946–985, 2019.

Said Dabia, Stefan Ropke, Tom van Woensel, and Ton De Kok. Branch and price for the time-dependent vehicle routing problem with time windows. *Transport. Sci.*, 47(3):380–396, 2013.

Brian A. Davis and Miguel A. Figliozzi. A methodology to evaluate the competitiveness of electric delivery trucks. *Transport. Res. E*, 49(1):8–23, 2013.

Emrah Demir, Tolga Bektaş, and Gilbert Laporte. A review of recent research on green road freight transportation. *Eur. J. Oper. Res.*, 237(3):775–793, 2014.

Guy Desaulniers, Fausto Errico, Stefan Irnich, and Michael Schneider. Exact algorithms for electric vehicle-routing problems with time windows. *Oper. Res.*, 64(6):1388–1405, 2016.

Jacques Desrosiers, Yvan Dumas, Marius M. Solomon, and François Soumis. Time constrained routing and scheduling. In *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, pages 35–139. Elsevier, 1995.

Sevgi Erdoğan and Elise Miller-Hooks. A green vehicle routing problem. *Transport. Res. E*, 109:100–114, 01 2012.

Angel Felipe, M.Teresa Ortuño, Giovanni Righini, and Gregorio Tirado. A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transport. Res. E*, 71:111–128, 11 2014.

Aurélien Froger, Jorge E. Mendoza, Ola Jabali, and Gilbert Laporte. Improved formulations and algorithmic components for the electric vehicle routing problem with nonlinear charging functions. *Comput. Oper. Res.*, 104:256–294, 2019.

Ricardo Fukasawa, Qie He, Fernando Santos, and Yongjia Song. A joint vehicle routing and speed optimization problem. *INFORMS J. Comput.*, 30(4):694–709, 2018.

Michel Gendreau, Gianpaolo Ghiani, and Emanuela Guerriero. Time-dependent routing problems: a review. *Comput. Oper. Res.*, 64:189–197, 2015.

Dominik Goeke and Michael Schneider. Routing a mixed fleet of electric and conventional vehicles. *Eur. J. Oper. Res.*, 245(1):81–99, 2015.

H. Heni. *Optimization of time-dependent routing problems considering dynamic paths and fuel consumption.* PhD thesis, Université Laval, 2018.

Yixiao Huang, Lei Zhao, Tom Van Woensel, and Jean-Philippe Gross. Time-dependent vehicle routing problem with path flexibility. *Transport. Res. B*, 95:169–195, 2017.

Soumia Ichoua, Michel Gendreau, and Jean-Yves Potvin. Vehicle dispatching with time-dependent travel times. *Eur. J. Oper. Res.*, 144(2):379 – 396, 2003.

Mads Jepsen, Bjørn Petersen, Simon Spoorendonk, and David Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Oper. Res.*, 56:497–511, 04 2008.

Merve Keskin, Gilbert Laporte, and Bülent Çatay. Electric vehicle routing problem with time-dependent waiting times at recharging stations. *Comput. Oper. Res.*, 107:77–94, 2019.

Merve Keskin and Bülent Çatay. Partial recharge strategies for the electric routing problem with time windows. *Transport. Res. C*, 65:111–127, 03 2016.

Gonzalo Lera-Romero and Juan José Miranda-Bront. A branch and cut algorithm for the time-dependent profitable tour problem with resource constraints. *Eur. J. Oper. Res.*, (Icc):1–24, 2019.

Gonzalo Lera-Romero, Juan J. Miranda Bront, and Francisco J. Soulignac. Linear edge costs and labeling algorithms: The case of the time-dependent vehicle routing problem with time windows. *Networks*, 76(1): 24–53, 2020a.

Gonzalo Lera-Romero, Juan José Miranda-Bront, and Francisco Juan Soulignac. Dynamic programming for the time-dependent traveling salesman problem with time windows. Technical report, Optimization Online, 2020b.

Ji Lu, Yuning Chen, Jin-Kao Hao, and Renjie He. The time-dependent electric vehicle routing problem: Model and solution. *Expert Syst. Appl.*, page 113593, 2020.

Alejandro Montoya, Christelle Guéret, Jorge E. Mendoza, and Juan G. Villegas. The electric vehicle routing problem with nonlinear charging function. *Transport. Res. B*, 103:87–110, 2017.

Samuel Pelletier, Ola Jabali, and Gilbert Laporte. Goods distribution with electric vehicles: Review and research perspectives. *Transport. Sci.*, 50(1):3–22, 2016.

Samuel Pelletier, Ola Jabali, and Gilbert Laporte. The electric vehicle routing problem with energy consumption uncertainty. *Transport. Res. B*, 126:225–255, 2019.

J. Restrepo, J. Rosero, and S. Tellez. Performance testing of electric vehicles on operating conditions in Bogotá DC, Colombia. In *2014 IEEE PES Transmission Distribution Conference and Exposition – Latin America*, pages 1–8, 2014.

R. Roberti and M. Wen. The electric traveling salesman problem with time windows. *Transport. Res. E*, 89: 32–52, 2016.

Ons Sassi, Wahiba Ramdane Cherif, and Ammar Oulamara. Vehicle routing problem with mixed fleet of conventional and heterogenous electric vehicles and time dependent charging costs. Technical report, 2014. URL https://hal.archives-ouvertes.fr/hal-01083966.

Martin W. P. Savelsbergh and Tom Van Woensel. 50th anniversary invited article—city logistics: Challenges and opportunities. *Transport. Sci.*, 50(2):579–590, 2016.

Maximilian Schiffer and Grit Walther. The electric location routing problem with time windows and partial recharging. *Eur. J. Oper. Res.*, 260, 01 2017.

Maximilian Schiffer, Michael Schneider, Grit Walther, and Gilbert Laporte. Vehicle routing and location routing with intermediate stops: A review. *Transport. Sci.*, 53(2):319–343, 2019.

Michael Schneider, Andreas Stenger, and Dominik Goeke. The electric vehicle-routing problem with time windows and recharging stations. *Transport. Sci.*, 48(4):500–520, 2014.

Sai Shao, Wei Guan, Bin Ran, Zhengbing He, and Jun Bi. Electric vehicle routing problem with charging time and variable travel time. *Math. Probl. Eng.*, 2017, 2017.

Peng Sun, Lucas P. Veelenturf, Said Dabia, and Tom Van Woensel. The time-dependent capacitated profitable tour problem with time windows and precedence constraints. *Eur. J. Oper. Res.*, 264(3):1058–1073, 2018.

Paolo Toth and Daniele Vigo, editors. *Vehicle Routing: Problems, Methods, and Applications.* 2nd ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2014.

UPS. Accelerating sustainable solutions, UPS 2019 sustainability progress report, 2019.

Duc Minh Vu, Mike Hewitt, Natashia Boland, and Martin Savelsbergh. Dynamic discretization discovery for solving the time-dependent traveling salesman problem with time windows. *Transport. Sci.*, 54(3): 703–720, 2020.

Xinkai Wu, David Freese, Alfredo Cabrera, and William A. Kitch. Electric vehicles' energy consumption measurement and estimation. *Transport. Res. D*, 34:52–67, 2015.

A Proof of Proposition 2

Proof. Suppose f has $|f|$ pieces $f_1, \ldots, f_{|f|}$ in the domain $[b(x), b(y)]$, and let $b(x) = u_0, \ldots, u_{|f|} = b(y)$ be its breakpoints. Since b is continuous and non-decreasing, it follows that there exist $x = t_0 < \ldots < t_{|f|} = y$ such that $b(t_i) = u_i$ for every $1 \leq i \leq |f|$. By induction on $i = 0, \ldots, |f|$ we shall prove that G is continuous, piecewise linear, and non-decreasing in the domain $[t_0, t_i]$. The base case $i = 0$ is trivial. For the inductive step $i + 1$, let $z = \max\{f(t') \mid t' \in [u_i, u_{i+1}]\}$. If $z \leq G(t_i)$, then $G(t) = G(t_i) = \max\{f(t') \mid t' \in [u_0, u_i]\}$ for every $t \in [t_i, t_{i+1}]$; clearly G is continuous, piecewise linear, and non-decreasing in $[t_0, t_{i+1}]$. Otherwise, since f is continuous and f_i is linear, it follows that f_i is increasing and there exists some $u^* \in [u_i, u_{i+1}]$ such that $f_i(u^*) = G(t_i) \geq f(u_i)$. Note, also, that $u^* = b(t^*)$ for some $t^* \in [t_i, t_{i+1}]$ because b is continuous. Therefore, $G(t) = G(t_i)$ for every $t \in [t_i, t^*]$ and, since b is non-decreasing, $G(t) = f_i(b(t)) = \max\{f(t') \mid t' \in [u_0, b(t)]\}$ for every $t \in [t^*, t_{i+1}]$. Since f_i is linear, we conclude that G is continuous, piecewise linear, and non-decreasing in $[t_0, t_{i+1}]$. □

B Example for the battery level function

Figure 6 illustrates the idea of the battery level function and its different steps. Suppose there is a two-vertex path $p = (i, j)$ traveled by an EV for $i, j \in V$. For simplification purposes suppose there are no time windows ($a_i = a_j = 0$, $b_i = b_j = T$), vertex $i = o$ is the depot, and the travel time is illustrated in Figure 6b, the battery capacity is $B = 10$ and the discharge function β_{ij} is as described in Figure 6a. Finally, let $\hat{g}(t)$ be the non-linear recharging function as defined in Figure 6c.

The single vertex path $p_0 = (i) = (o)$ has a constant battery level function $\lambda_{p_0}(t) = B$ for all $t \in [0, T]$, which represents a full-charge when departing from the depot. Then, the battery level function λ_p results from extending λ_{p_0} over the arc (i, j). In this case, we have two possible scenarios: (i) $j \in V \setminus V_s$ or (ii) $j \in V_s$.

In the first scenario, the battery level function λ_p after traversing arc (i, j) is depicted in Figure 6d. Here, the red segment shows the times when the battery level was achieved by means of waiting. For example, the battery level if arriving to j **exactly** at time $t = 10$ is 4. This occurs since vertex i must be departed from at time $t' = 9$, and then the battery consumption must be 6. Observe however that $\lambda_p(10) = 8$, since the EV can arrive to j at time $t' = 6 < 10$ with battery level 8 and then wait 4 units of time.

The second scenario shows the battery level function λ_p when $j \in V_s$ is a recharging station. In this case, function CHG_j is applied and the result is illustrated in Figure 6e. The red portion of the domain includes those times where the battery level was achieved after performing a recharge.

C Proof of Proposition 4

Proof. Clearly, having $\bar{y}_r \in \{0, 1\}$ for $r \in \Omega$ guarantees $\bar{z}_{ij} \in \{0, 1\}$ by definition since each arc (i, j) can be present in at most one route having $\bar{y}_r > 0$. Conversely, recall that for each customer set $S \subseteq V_c$ there is at most one route that visits exactly S in Ω. Additionally, constraints (12), (14) and hypothesis that $\bar{z}_{ij} \in \{0, 1\}$ enforce that for each customer $k \in V_c$ there is exactly one active variable $\bar{z}_{kj}$ and $\bar{z}_{ik}$ for some $i, j \in V_c \cup \{o, d\}$. If two active routes $r_1, r_2 \in \Omega$ such that $\bar{y}_{r_1}, \bar{y}_{r_2} > 0$ share a customer $k \in V_c$, then the customer vertex following k in $CR(r_1)$ and $CR(r_2)$ must be the same (or the previous vertex is the depot).

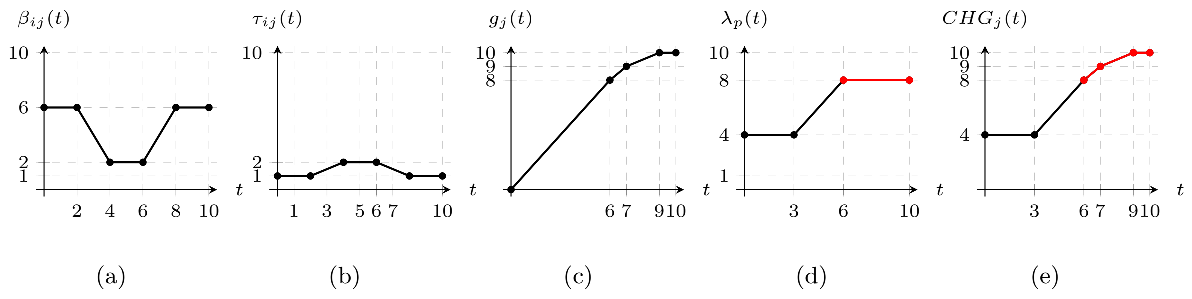


Figure 6: Example of the battery level function $\lambda_p(t)$, including the discharge function β_{ij} (6a), travel time τ_{ij} (6b), charge function $g_j(t)$ (6c), $TRV_{ij}, j \in V \setminus V_s$ (6d), and $CHG_j, j \in V_s$ (6e)

An analogous argument also holds for the previous customer vertex. As a result, it is simple to verify that $CR(r_1) = CR(r_2)$, and then $r_1 = r_2$ since there a unique route in Ω for each $S \subseteq V_c$. □

D Intances and Detailed Results

Table 5 shows the results for the two branching rules, CBR and ABR. We report the number of instances solved to optimality (opt), the total execution time (time), and the number of nodes enumerated (nodes) aggregated by each recharge policy and instance customer count $|V_c|$ and averaged over the instances solved by both methods.

			time independent				time dependent			
dataset	$\|V_c\|$	inst	opt	time	nodes	%rg	opt	time	nodes	%rg
Basic	25	56	56	7.05	1.04	0.00	56	25.13	1.14	0.01
	50	56	48	193.27	2.88	0.06	44	197.67	2.00	0.09
	100	56	17	775.94	11.82	0.27	17	822.30	10.88	0.34
NC	25	56	56	8.72	1.25	0.02	56	43.16	1.11	0.02
	50	56	48	350.40	2.83	0.06	47	291.80	1.51	0.05
	100	56	16	536.48	9.63	0.22	15	446.46	7.67	0.19
WT	25	56	56	5.64	1.21	0.02	56	21.84	1.75	0.05
	50	56	48	219.54	6.75	0.10	44	165.87	3.45	0.12
	100	56	18	907.41	20.00	0.43	14	727.61	16.86	0.70
ALL	25	56	56	6.77	1.21	0.01	56	41.76	1.04	0.00
	50	56	48	259.72	3.75	0.06	47	263.20	3.21	0.08
	100	56	15	530.57	14.87	0.76	13	489.25	14.54	0.54

Table 5: Comparison of the different branching strategies from Section 4.2.

The results are shown in Table 6 in Appendix D, where the key is the same as in the previous experiments. In this case, averages are computed over all instances solved to optimality by the corresponding method. To enable a comparison, the values of %rg for the algorithms developed Desaulniers et al. (2016) are computed over all instances based on the results they report.

			Desaulniers et al. (2016)								BCP			
			Monodirectional				Bidirectional							
Var.	$\|V_c\|$	inst	opt	time	nodes	%rg	opt	time	nodes	%rg	opt	time	nodes	%rg
	25	56	55	13.90	1.25	0.08	56	30.60	1.25	0.08	56	2.23	1.36	0.03
SF	50	56	46	265.97	7.74	0.14	47	250.87	7.47	0.16	47	92.28	2.74	0.09
	100	56	13	575.68	33.92	0.12	14	615.30	36.43	0.13	19	552.28	18.16	0.23
	25	56	55	112.98	1.22	0.04	56	16.03	1.21	0.04	56	1.78	1.11	0.01
SP	50	56	44	255.35	4.68	0.09	50	297.38	7.16	0.12	49	162.39	3.12	0.08
	100	56	18	578.65	6.44	0.06	22	628.40	15.36	0.08	22	572.31	12.64	0.12
	25	56	53	93.22	1.19	0.03	56	93.43	1.21	0.03	56	11.08	1.50	0.02
MF	50	56	44	357.85	27.09	0.05	46	258.94	31.17	0.05	46	138.58	1.74	0.04
	100	56	10	599.27	17.20	0.10	13	780.16	77.15	0.14	18	625.86	12.89	0.19
	25	56	52	151.33	1.08	0.01	54	23.48	1.11	0.01	56	7.05	1.04	0.00
MP	50	56	37	374.72	85.43	0.12	45	494.80	82.78	0.12	48	193.27	2.88	0.06
	100	56	15	855.50	5.40	0.10	18	576.32	13.11	0.11	17	775.94	11.82	0.13

Table 6: Comparison with numerical results from Desaulniers et al. (2016) for the time-independent scenario.