Transportation Research Part C

A Route-Based Algorithm for the Electric Vehicle Routing Problem with Multiple Technologies

--Manuscript Draft--

Manuscript Number:	TRC-22-00048
Article Type:	Research Paper
Keywords:	electric vehicles; routing; branch-and-price; dynamic programming
Corresponding Author:	Alberto Ceselli Università degli Studi di Milano: Universita degli Studi di Milano ITALY
First Author:	Dario Bezzi
Order of Authors:	Dario Bezzi
	Alberto Ceselli
	Giovanni Righini, Prof.
Abstract:	Electric vehicles are seen as a pragmatic way of reducing emissions. In freight transportation, they prove to be appealing also in terms of costs, if proper planning algorithms are designed. We consider a variant of the electric vehicle routing problem: a fleet of identical vehicles of limited capacity needs to visit a set of customers of given demand. Vehicles have limited batteries and limited time: they need to stop en-route to recharge stations. Our variant has two distinguishing features: recharges can be partial and multiple recharge technologies are available at stations, providing energy at different costs and different recharge rates. We propose a novel exact algorithm. It relies on an extended formulation, having one variable for each possible depot-to-depot route of each vehicle, implicitly encoding also recharge plans. We optimize it by branch-and-price. We design ad-hoc pricing algorithms, which exploit a special encoding of recharge plans, allowing for efficient bi-directional dynamic programming techniques. Extensive computational results show our approach to clearly outperform previous ones from the literature.
Suggested Reviewers:	Maurizio Bruglieri maurizio.bruglieri@polimi.it recent publications on the topic of Electric Vehicle Routing
	Matteo Salani matteo.salani@idsia.ch Expert on the methodologies which are used in our paper.
	Gunes Erdogan g.erdogan@bath.ac.uk Expert in various vehicle routing problems and applications.
	Stefan Irnich irnich@uni-mainz.de Expert in vehicle routing algorithms.

Università degli Studi di Milano
Dipartimento di Informatica

Milan, *January 7th 2022*

Dear editors,

we send you the paper "A Route-Based Algorithm for the Electric Vehicle Routing Problem with Multiple Technologies", whose authors are the following:

Dario Bezzi
Dipartimento di Scienze e Metodi dell'Ingegneria,
Università degli Studi di Modena e Reggio Emilia
dario.bezzi@unimore.it

Alberto Ceselli, Giovanni Righini
Dipartimento di Informatica,
Università degli Studi di Milano
alberto.ceselli@unimi.it, giovanni.righini@unimi.it

Please, consider it to undertake a review process on Transportation Research Part C.

We hereby confirm that the paper contains entirely new contributions, and will be submitted to no other journal or conference during the review process. We also state compliance to the ethical and conflict of interest policies of your journal.

Kind regards,

Alberto Ceselli (on behalf of all co-authors)

A Route-Based Algorithm for the Electric Vehicle Routing Problem with Multiple Technologies

Dario Bezzi
Dipartimento di Scienze e Metodi dell'Ingegneria,
Università degli Studi di Modena e Reggio Emilia
dario.bezzi@unimore.it

Alberto Ceselli [*], Giovanni Righini
Dipartimento di Informatica,
Università degli Studi di Milano
alberto.ceselli@unimi.it, giovanni.righini@unimi.it

January 7, 2022

Abstract

Electric vehicles are seen as a pragmatic way of reducing emissions. In freight transportation, they prove to be appealing also in terms of costs, if proper planning algorithms are designed. We consider a variant of the electric vehicle routing problem: a fleet of identical vehicles of limited capacity needs to visit a set of customers of given demand. Vehicles have limited batteries and limited time: they need to stop en-route to recharge stations. Our variant has two distinguishing features: recharges can be partial and multiple recharge technologies are available at stations, providing energy at different costs and different recharge rates.

We propose a novel exact algorithm. It relies on an extended formulation, having one variable for each possible depot-to-depot route of each vehicle, implicitly encoding also recharge plans. We optimize it by branch-and-price. We design ad-hoc pricing algorithms, which exploit a special encoding of recharge plans, allowing for efficient bi-directional dynamic programming techniques.

Extensive computational results show our approach to clearly outperform previous ones from the literature.

Keywords: electric vehicles, routing, branch-and-price, dynamic programming.
Declarations of interest: none.

1 Introduction

Environmental care, especially in terms of emissions control, is currently a major concern. As possible ways out, governments and industries alike are promoting several actions, many requiring technological shifts. It is the case of adoption of electric engines in place of thermal ones. In fact, recent evaluations like [19] show such an approach to be not only environmental friendly, but also cost appealing, also in the field of freight transportation.

[*]corresponding author: Alberto Ceselli, Via Celoria 18, 20133 - Milano (Italy). Tel. +39 02 503 16226.

As common in transportation, in order not to lose such a potential advantage, technological shifts require methodological ones in terms of optimization models. That is why also the literature on the management of electric vehicles has rapidly developed in the last years. A comprehensive survey on the critical issue of optimal routing was published in 2015 by Pelletier, Jabali and Laporte [15].

In this context, the electric vehicle routing problem (EVRP) consists of optimally routing a fleet of electric vehicles to visit a given set of customers. Unlike traditional ones, electric vehicles have limited autonomy, which forces to explicitly plan recharge stops along routes. This makes the combinatorial strucure of EVRPs substantially different to their traditional counterparts. Recently, Keskin, Laporte and Catay [11] classified 49 EVRP papers according to fleet composition, objective function terms, presence of multiple recharge technologies and constraints such as capacities and time windows.

In this paper we focus on exact optimization algorithms for the following variant of the EVRP. The fleet consists of identical electric vehicles with a given capacity and each customer has an associated demand as in the classical capacitated VRP. The vehicles are equipped with a battery of given capacity and each arc implies a given energy consumption. To avoid running out of energy, vehicles can visit recharge stations in given positions.

Our variant has the following two distinguishing features. First, partial recharges are allowed, the cost and time of the each recharge being proportional to the amount of recharged energy. Second, at each station several recharge technologies can be available, with different energy price and recharge speed.

A given fixed cost is also charged for every recharge operation. The vehicles can leave the depot with full charge and the recharge technology at the depot is assumed to be the slowest and cheapest one. A maximum duration is imposed to all routes. The objective is to minimize the overall energy consumption cost.

This problem was first considered by Felipe et al. [8], who developed heuristics. Their results also showed the potential savings achievable owing to the availability of several recharge technologies and partial recharges. Several other authors proposed heuristic algorithms for this variation of the EVRP: among others, Sassi et al. [16] proposed a MILP formulation and a local search heuristic; Li-Ying and Yuan-Bin [13] proposed an adaptive variable neighborhood search; Koç, Jabali and Laporte [12] proposed a matheuristic algorithm, combining adaptive large neighborhood search and mixed integer linear programming.

Exact optimization algorithms for the EVRP were developed for instance by Desaulniers et al. [5], where a single recharge technology was considered.

The only attempt to develop an exact optimization algorithm for the EVRP with multiple recharge technologies is described in [3]. The algorithm is a branch-and-cut-and-price, built on a formulation where columns correspond to paths between recharge stations. Such an extended formulation lends itself to a highly parallelized implementation of the pricing algorithm. A similar idea was also explored by Bruglieri et al. [2] to develop a two-stage approach based on a MILP formulation.

Main contributions. In this paper we present a new exact algorithm. It is a branch-and-price, based on a formulation where columns correspond to complete routes from the depot to the depot (Section 3) With respect to the path-based extended formulation used in [3], our route-based extended formulation has the advantage of providing tighter lower bounds.

As a drawback, the structure of pricing problems becomes much more involved. We therefore design exact pricing algorithms (Section 4). They rely on a novel encoding of partial recharge plans, besides partial routes, which allows us to devise smart dynamic programming techniques.

In order to assess their effectiveness, and that of a full branch-and-price when specific branching rules are included (Section 5), we report on experiments using datasets from the literature (Section 6). We finally summarize our findings, drawing some conclusions and perspectives (Section 7).

2 Problem definition

The EVRP problem with partial recharges and multiple technologies can be formulated as follows.

Let $G = (\mathcal{N} \cup \mathcal{R}, \mathcal{E})$ be a weighted undirected graph, where $\mathcal{N}$ is a set of n customers and $\mathcal{R}$ is a set of m recharge stations. A distinguished station, $R_0 \in \mathcal{R}$, is located at the depot where all vehicle routes start and end. A set $\mathcal{H} = \{0, \ldots, H\}$ includes $H + 1$ recharge technologies. Each technology $h \in \mathcal{H}$ is characterized by a unit recharge time ρ_h and a unit recharge cost γ_h. Without loss of generality, we assume that faster technologies are more expensive, i.e. $\rho_{h'} < \rho_{h''} \Leftrightarrow \gamma_{h'} > \gamma_{h''}$ $\forall h', h'' \in \mathcal{H}$. We also assume that a single technology is available at each station; this is again without loss of generality, since such a condition can be enforced by simple station node replication. Finally, we assume that technology $h = 0$ is available only at the depot R_0, with $\rho_0 = 0$ and $0 < \gamma_0 \leq \gamma_h \forall h \in \mathcal{H}$.

A demand q_i and a service time s_i are associated with each customer $i \in \mathcal{N}$; all customers must be visited exactly once: split delivery is not allowed.

The available fleet consists of K identical vehicles with given capacity Q and equipped with batteries of given capacity B. Energy consumption and time consumption are assumed to be proportional to the distance traveled: each edge $e \in \mathcal{E}$ requires d_e units of energy and t_e units of time.

While every customer must be visited once, stations can be visited at any time by vehicles, even more than once along the same route. During each recharge operation at station $j \in \mathcal{R}$, a vehicle can recharge any amount δ of energy up to its residual battery capacity. The time consumption for a recharge of an amount δ at a station $j \in \mathcal{R}$ is computed as $s_j + \rho_j \delta$, where s_j is the service time at station j; the recharge cost is given by $\gamma_j \delta$. The duration of each route, which is the sum of its travel and recharge time, is required to be within a given limit T. The objective is to minimize the total recharge cost.

2.1 Properties

Formally, a *route* is an ordered sequence of vertices. The depot R_0 is the initial and final vertex of every route, and it never appears as an intermediate vertex. Customer vertices cannot appear more than once in feasible routes, whereas station vertices can.

For every route r, a corresponding *recharge plan* δ^r must be computed, specifying the total amount of energy δ^r_h recharged with each technology $h \in \mathcal{H}$ along route r, potentially at different recharge stations.

For every route r, let us denote by $E_r \subseteq \mathcal{E}$ the multiset of edges traversed along it. Then we can define the following indicators for each route r:

- total capacity consumption: $\phi_r = \sum_{i \in r \cap \mathcal{N}} q_i$;
- total time consumption: $\tau_r = \sum_{e \in E_r} t_e + \sum_{k \in r} s_k + \sum_{h \in \mathcal{H}} \rho_h \delta^r_h$, where each term t_e occurs as many times as e occurs in E_r;
- total cost $c_r = \sum_{h \in \mathcal{H}} (\gamma_h \delta^r_h)$.

We define a route r with its recharge plan δ^r to be *feasible* if and only if these three properties hold:

- Capacity constraint is respected: $\phi_r \leq Q$;
- Time constraint is respected: $\tau_r \leq T$;
- Battery constraint is respected: the battery charge is kept between 0 and B at any time.

Before stating the mathematical model used for branch-and-price, we outline two useful properties of the EVRP.

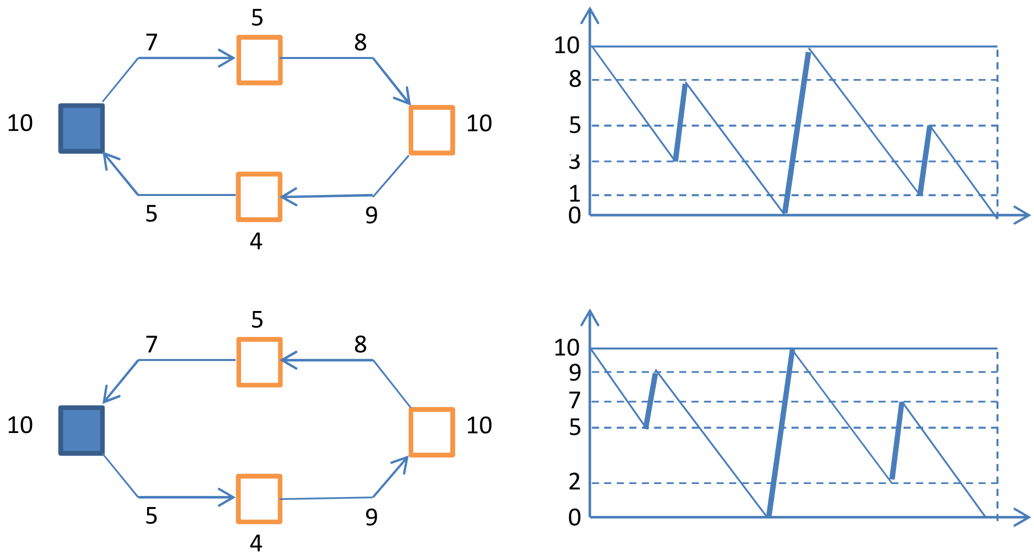

Figure 1: Every feasible route remains feasible in both directions. The battery charge profile of the reversed route is obtained by rotating the charge profile by 180 degrees.

Observation 1. *It is optimal to return to the depot with empty battery. Under the assumption that the recharge cost at the depot is smaller than at the other recharge stations, if the total energy consumption along the route is not smaller than B, then it is optimal to start from the depot with full battery.*

Observation 2. *Being the graph $\mathcal{G}$ symmetric, every feasible route can be traversed in both directions at the same cost using the same recharge plan.*

Proof. In a feasible route the battery charge level is always between 0 and B and the residual battery capacity at any point is given by the difference betweeen B and the battery charge level. At any point along a route, the battery charge level is equivalent to the residual battery capacity when the route is reversed. Therefore, the battery charge level is always between 0 and B along the reversed route, as shown in Figure 1, and hence traveling along a feasible route in the opposite direction is still feasible. $\square$

3 Mathematical formulation

The branch-and-price algorithm illustrated in the remainder is based on an extended formulation where each column represents a feasible route.

3.1 Master problem

Let us define Ω as the set of all feasible routes. A binary variable x_r is associated with each feasible route $r \in \Omega$ and c_r indicates the route cost. Binary coefficients y_{ir} indicate whether each customer $i \in \mathcal{N}$ is visited or not along each route $r \in \Omega$. With this notation the following master problem

(MP) is obtained:

$$\text{minimize} \sum_{r\in\Omega} c_r x_r \tag{1}$$

$$\text{s.t.} \sum_{r\in\Omega} y_{ir} x_r \geq 1 \qquad \forall i \in \mathcal{N} \tag{2}$$

$$-\sum_{r\in\Omega} x_r \geq -K \tag{3}$$

$$x_r \in \{0,1\} \qquad \forall r \in \Omega \tag{4}$$

The objective function (1) asks for the minimization of the total cost of the selected routes. Covering constraints (2) impose that all customers be visited; constraint (3) allows to select at most K columns in the solution.

We indicate by LMP the linear relaxation of the master problem. We indicate by $\beta \geq 0$ the dual variables vector corresponding to the covering constraints (2) and by $\mu \geq 0$ the dual variable corresponding to constraint (3). With this notation, the expression of the reduced cost of a generic column r in the LMP is:

$$\overline{c}_r = c_r - \sum_{i\in\mathcal{N}} \beta_i y_{ir} + \mu.$$

At each node of a branch-and-bound tree the linear relaxation of the master problem is solved by column generation, where a restricted linear master problem ($RLMP$) with a subset $\Omega' \subset \Omega$ of columns is iteratively solved.

To guarantee its feasibility, the $LRMP$ is initially populated with a dummy column corresponding to a route visiting all customers at a very large cost. For this purpose we use a greedy heuristic algorithm to compute a route $\tilde{r}$ visiting all customers with no constraints on duration and capacity. Its cost $\tilde{c}$, its time consumption $\tilde{\tau}$ and its capacity consumption $\tilde{\phi}$ ($= \sum_{i\in\mathcal{N}} q_i$) are guaranteed to be larger than or equal to the cost, time consumption and capacity consumption of any route in an optimal solution. Therefore $K \cdot \tilde{c}$ is used as the cost of the dummy column, while the input data T and Q are updated to $\min\{T, \tilde{\tau}\}$ and $\min\{Q, \tilde{\phi}\}$, respectively.

3.2 Pricing sub-problem

The pricing sub-problem requires to find a minimum cost closed walk from the depot to the depot, not visiting any customer vertex more than once and not consuming more than a given amount of capacity and time. The vehicle can visit recharge stations to comply with battery constraints. A reward $\beta_i \geq 0$ is obtained from visiting each customer $i \in \mathcal{N}$ and a fixed cost $\mu \geq 0$ is paid at the depot.

This problem is a variation of the Resource Constrained Elementary Shortest Path Problem (which is known to be NP-hard, as stated in [6]), in which the elementary path constraints are imposed only on a subset of vertices, the resources are partly discrete (capacity) and partly continuous (time) and one of the resources (energy) is renewable.

4 Pricing algorithm

We solve the pricing sub-problem with a bi-directional dynamic programming (DP) algorithm, where states correspond to paths with an endpoint at the depot. In DP algorithms developed for similar purposes, encoding paths in DP labels is usually enough to fully define pricing solutions. In our case, instead, it is necessary to encode both the path and the recharge plan along it.

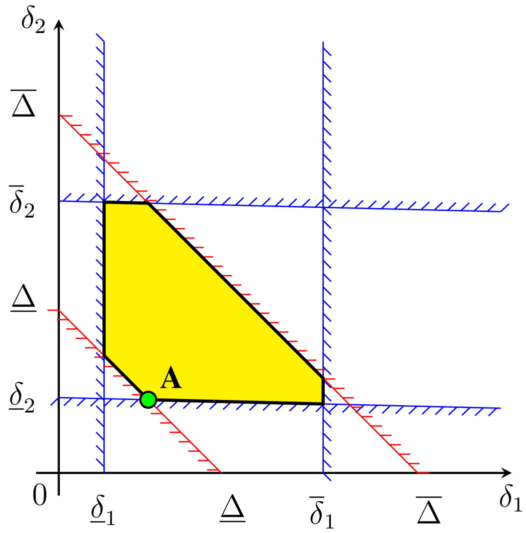

Figure 2: The reference point of a feasible recharge polyhedron $\mathcal{P}$ is the vertex of $\mathcal{P}$ with minimum cost (point **A**, assuming $\gamma_1 < \gamma_2$).

4.1 Solution encoding

As discussed above, for each path infinite feasible recharge plans may exist. They should be recorded in the piece of information associated with each DP label. Accordingly, we introduce the notion of feasible recharge polyhedron.

Definition 1. *A* Feasible Recharge Polyhedron *(FRP)* $\mathcal{P} = (\underline{\Delta}, \overline{\Delta}, \underline{\delta}, \overline{\delta})$ *is defined in as many dimensions as the number of technologies* H *and it has the following special structure:*

- *the two scalars* $\underline{\Delta}$ *and* $\overline{\Delta}$ *represent the minimum and maximum total amount of recharged energy along the path;*
- *the two vectors* $\underline{\delta} = \{\underline{\delta}_0, \dots \underline{\delta}_H\}$ *and* $\overline{\delta} = \{\overline{\delta}_0, \dots \overline{\delta}_H\}$ *represent the minimum and maximum amounts of energy* δ_h *recharged for each* $h \in \mathcal{H}$*.*

The *reference point* of $\mathcal{P}$ is the vertex of $\mathcal{P}$ with minimum cost. We indicate the coordinates of the reference point with a vector $\hat{\delta} = \{\hat{\delta}_0, \dots, \hat{\delta}_{\mathcal{H}}\}$ for any given $\mathcal{P}$. Figure 2 shows an example of a FRP for the case $\mathcal{H} = \{1, 2\}$.

In the DP algorithm, the states corresponding to a generic path from the depot to a vertex u are represented by labels $\mathcal{L} = (u, S, \phi, \mathcal{P}, \tau, \overline{c})$, where

- $u \in \mathcal{N} \cup \mathcal{R}$ is the last vertex reached by the path, that starts from the depot;
- $S \subseteq \mathcal{N}$ is the set of customers visited by the path;
- ϕ is the amount of capacity consumed along the path: $\phi = \sum_{i \in S} q_i$;
- $\mathcal{P}$ is the FRP representing the set of feasible recharge plans associated with the path;
- τ is the time needed to reach the *reference point* of $\mathcal{P}$ along the path;
- $\overline{c}$ is the reduced cost corresponding to the *reference point* of $\mathcal{P}$.

Every DP label represents an infinite set of states and each state corresponds to a point of the FRP.

The initial state is $\mathcal{L}_0 = (R_0, \emptyset, 0, \mathcal{P}^0, 0, 0)$, corresponding to a degenerate path including only the depot R_0. The initial polyhedron $\mathcal{P}^0$ is defined as follows:

$$\mathcal{P}^0 = \begin{cases} \underline{\Delta} = 0 \\ \overline{\Delta} = B \\ \underline{\delta_h} = 0 \; \forall h \in \mathcal{H} \\ \overline{\delta_h} = 0 \; \forall h \in \mathcal{H} \backslash \{0\} \\ \overline{\delta_0} = B \end{cases}$$

4.2 Label extension

When a label $\mathcal{L}' = (i, S', \phi', \mathcal{P}', \tau', \overline{c}')$ of vertex i is tentatively extended along an edge $e = [i, j]$ to generate a new label for vertex j, the following conditions are tested:

- Elementarity: $j \notin S'$;
- Capacity: $\phi' + q_i/2 + q_j/2 \leq Q$ (assuming $q_j = 0 \; \forall j \in \mathcal{R}$);
- Duration: $\tau' + s_i/2 + t_e + s_j/2 \leq T$;
- Energy: $\overline{\Delta}' - \underline{\Delta}' \geq d_e$.

A valid but stronger version of the last two feasibility conditions is

- Duration: $\tau' + s_i/2 + t_e + s_j/2 + t_{[0,j]} \leq T$,
- Energy: $\overline{\Delta}' - \underline{\Delta}' \geq d_e + m_j$,

where $t_{[0,j]}$ is the travel time along the edge from j to the depot, while m_j is the amount of energy needed to reach the nearest recharge station from vertex j (it can be pre-computed in linear time for each vertex: $m_j = \min_{k \in \mathcal{R}} d_{[j,k]}$).

The next observation is used to avoid infinite loops and some useless extensions.

Observation 3. *If $j \in \mathcal{R}$ and no customer vertex has been visited after the last visit to j, then the extension toward j is forbidden. Keeping a boolean flag for each station is enough for this test.*

When the extension of a label $\mathcal{L}' = (i, S', \phi', \mathcal{P}', \tau', \overline{c}')$ to a vertex j along edge $e = [i, j]$ is feasible, a new label $\mathcal{L}'' = (j, S'', \phi'', \mathcal{P}'', \tau'', \overline{c}'')$ is generated according to the following rules.

First, the new FRP $\mathcal{P}''$ and its reference point $\hat{\delta}''$ are computed as indicated in Algorithm 1. Traveling along an edge e of length d_e increases $\underline{\Delta}$ by d_e.

Figures 3 and 4 show the effect of the FRP update: when the vehicle consumes energy traveling along an edge, $\underline{\Delta}$ (and possibly $\underline{\delta_h}$ for some h) increases, restricting the FRP (Figure 3). When the vehicle visits a station equipped with technology h, then $\overline{\Delta}$ and $\overline{\delta}_h$ increase, enlarging the FRP (Figure 4).

Finally, the following extension rules are applied to generate the new label:

- $S'' = \begin{cases} S' \cup \{j\} & \text{if } j \in \mathcal{N} \\ S' & \text{if } j \in \mathcal{R} \end{cases}$,
- $\phi'' = \phi' + q_i/2 + q_j/2$,
- $\tau'' = \tau' + s_i/2 + t_e + s_j/2 + \sum_{h \in \mathcal{H}} \rho_h(\hat{\delta}''_h - \hat{\delta}'_h)$,
- $\overline{c}'' = \overline{c}' - \beta_i/2 - \beta_j/2 + \sum_{h \in \mathcal{H}} \gamma_h(\hat{\delta}''_h - \hat{\delta}'_h)$.

Algorithm 1 Extension from $\mathcal{P}'$ to $\mathcal{P}''$ along edge e toward vertex j

// Bounds update //
$\overline{\Delta}'' \leftarrow \overline{\Delta}'$
$\underline{\Delta}'' \leftarrow \underline{\Delta}' + d_e$
for $h \in \mathcal{H}$ **do**
  $\overline{\delta}''_h \leftarrow \overline{\delta}'_h$
  $\underline{\delta}''_h \leftarrow \max\{\underline{\delta}'_h, \underline{\Delta}'' - \sum_{k\in\mathcal{H}:k\neq h} \overline{\delta}''_k\}$
// Reference point update //
$Z = d_e$
for $h \in \mathcal{H}$ **do**
  $\epsilon \leftarrow \min\{Z, \overline{\delta}''_h - \hat{\delta}'_h\}$
  $\hat{\delta}''_h \leftarrow \hat{\delta}'_h + \epsilon$
  $Z \leftarrow Z - \epsilon$
// Recharge (if j is a station vertex) //
if $j \in \mathcal{R}_h$ **then**
  $\overline{\Delta}'' \leftarrow \underline{\Delta}'' + B$
  $\overline{\delta}''_h \leftarrow \overline{\delta}'_h + (\overline{\Delta}'' - \overline{\Delta}')$

4.3 Dominance

The exponential number of labels to be generated is a primary source of inefficiency of the DP algorithm. For this reason, a dominance test is performed to fathom labels that cannot lead to an optimal solution, thus limiting the combinatorial explosion in the number of labels.

Efficient dominance rules for the RCESPP have been described in [10]. In our case, a label $\mathcal{L}' = (i, S', \phi', \mathcal{P}', \tau', \overline{c}')$ dominates a label $\mathcal{L}'' = (i, S'', \phi'', \mathcal{P}'', \tau'', \overline{c}'')$ if and only if the following inequalities hold and at least one of them is strict:

$$\begin{cases} S' \subseteq S'' \\ \phi' \leq \phi'' \\ \tau' \leq \tau'' \\ \overline{c}' \leq \overline{c}'' \\ \overline{\Delta}' - \underline{\Delta}' \geq \overline{\Delta}'' - \underline{\Delta}'' \\ \overline{\delta}'_h - \hat{\delta}'_h \geq \overline{\delta}''_h - \hat{\delta}''_h \ \ \forall h \in \mathcal{H} \\ \hat{\delta}'_h - \underline{\delta}'_h \geq \hat{\delta}''_h - \underline{\delta}''_h \ \ \forall h \in \mathcal{H} \end{cases}$$

To be met, the last three conditions require that $\mathcal{P}' \supseteq \mathcal{P}''$ when the reference points of the two polyhedra are made coincident.

Observation 4. *The test $S' \subseteq S''$ takes $O(|\mathcal{N}|)$ time. However, memory word parallelism can be triggered very efficiently, if implemented by bitwise operations.*

It therefore exhibits constant time in any practical run, implying that every dominance check exhibits $O(H)$ time complexity as a whole.

Labels are kept in buckets, one for each vertex of the graph. This allows to test for dominance only labels associated with the same vertex.

Following [7], unreachable customers are included in the subset S for each given path. These are customers that cannot be visited in any feasible extension of the path, owing to capacity or time constraints.

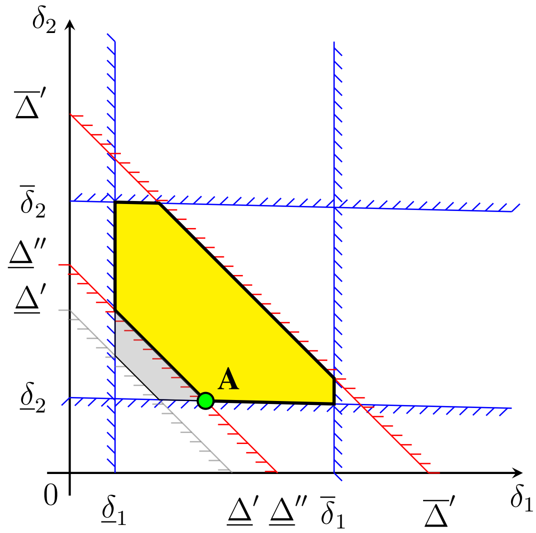

Figure 3: The FRP is restricted when an edge is traversed.

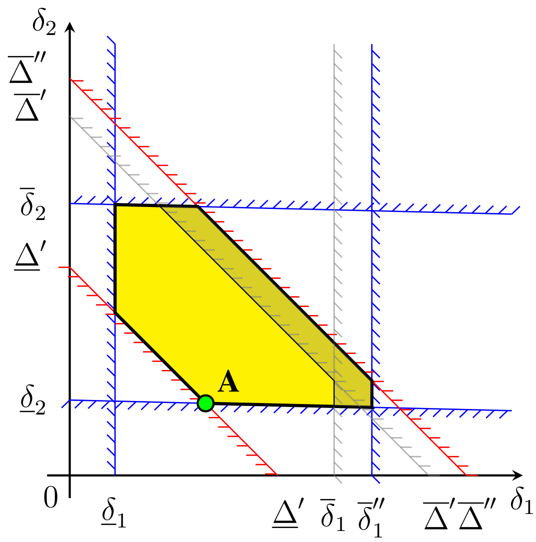

Figure 4: The FRP is enlarged when a station is visited.

4.4 Bi-directional DP

In general, bi-directional DP requires backward extension rules for labels. However, here we consider symmetric graphs and hence the same paths can be traversed in either direction at the same (reduced) cost in the same time. In bi-directional DP, labels are never extended to the depot.

The extension stop criterion must guarantee that every feasible route can be generated by a pair of paths. For this purpose extensions are stopped as soon as half of the available amount of a suitably chosen *critical resource* has been consumed. The selection of the critical resource can heavily affect the overall algorithm performance.

In the EVRP, vehicle capacity and time can be chosen as critical resources, whereas energy cannot because it can be recharged. If capacity is selected as critical, then the stop condition $\phi' + q_i/2 + q_j/2 \leq Q/2$ replaces the feasibility test $\phi' + q_i/2 + q_j \leq Q$; if time is slected as critical, then the stop condition $\tau' + s_i/2 + t_e + s_j/2 \leq T/2$ replaces the feasibility test $\tau' + s_i/2 + t_e + s_j \leq T$. For the run-time selection of the critical resource we devised and tested some strategies, described hereafter.

4.4.1 Critical resource selection

Since we have precomputed a dummy route $\tilde{r}$ with capacity consumption $\tilde{\phi}$ and time consumption $\tilde{\tau}$, as described in Subsection 3.1, the critical resource can be guessed from it. Three different criteria were considered:

- Criterion A is simply based on total resource consumptions: time is selected as critical when it is tight on $\tilde{r}$ (i.e. $\tilde{\tau} \geq T$), capacity otherwise.
- Criterion B compares the relative consumption of resources defined as follows:

$$\overline{Q} = Q/\tilde{\phi}$$

$$\overline{T} = T/\tilde{\tau}$$

  Time is selected as critical if $(\tilde{\phi} < Q) \vee (\overline{Q} > \overline{T})$, capacity otherwise.

- Criterion C is similar to B, but the definition of $\overline{T}$ is

$$\overline{T} = KT/\tilde{\tau}.$$

In Section 6.2, some computational results on the effectiveness of these criteria are presented.

4.4.2 Joining paths

When two paths with labels $\mathcal{L}' = (i, S', \phi', \mathcal{P}', \tau', \overline{c}')$ and $\mathcal{L}'' = (j, S'', \phi'', \mathcal{P}'', \tau'', \overline{c}'')$ are joined along the edge $e = [i, j]$, they produce a route whose energy consumption is given by $\underline{\Delta}' + \underline{\Delta}'' + d_e$. The energy needed to traverse edge e, implies an increase in time and in (reduced) cost. These amounts can be quickly lower bounded by $d_e\rho^*$ (time) and $d_e\gamma^*$ (reduced cost), where ρ^* is the unit recharge time of the fastest technology along the route, i.e. $\rho^* = \min_{j\in\mathcal{R}:\hat{\delta}_j<\bar{\delta}_j} \rho_j$ and γ^* is the unit recharge cost of the cheapest technology along the route, i.e. $\gamma^* = \min_{j\in\mathcal{R}:\hat{\delta}_j<\bar{\delta}_j} \gamma_j$.

Additionally, we must take into account that one of the two joined paths must be traversed *to* the depot, instead of *from* the depot. The energy consumption along a reversed path is the same, but the recharge technology cannot be the same, because the initial cheap and instantaneous recharge at the depot is not possible when traveling *to* the depot. Therefore some units of energy recharged with technology 0 (available at the depot only) must be replaced using a different technology. In the worst case (routes whose overall consumption is larger or equal to B) this implies the replacement of B units of energy. We call this amount *join energy*:

$$E^{join} = \max(0, \hat{\delta}'_0 + \hat{\delta}''_0 - B).$$

The additional time and reduced cost due to the join energy can be lower bounded by $E^{join}\rho^*$ (time) and $E^{join}(\gamma^* - \gamma_0)$ (cost), as before.

This lower bounding technique can be used to possibly discard a route immediately. Consider two labels $\mathcal{L}' = (i, S', \phi', \mathcal{P}', \tau', \overline{c}')$ and $\mathcal{L}'' = (j, S'', \phi'', \mathcal{P}'', \tau'', \overline{c}'')$ and the edge $e = [i, j]$. The following (necessary but not sufficient) conditions are tested to check whether the two corresponding paths can be joined to produce a feasible route

- Elementarity: $S' \cap S'' = \emptyset$;
- Capacity: $\phi' + \phi'' \leq Q - (q_i/2 + q_j/2)$;
- Energy: $(\overline{\Delta}' - \underline{\Delta}') + (\overline{\Delta}'' - \underline{\Delta}'') \geq d_e$;
- Duration: $\tau' + (E^{join} + d_e)\rho^* + \tau'' + s_i/2 + t_e + s_j/2 \leq T$;

Algorithm 2 Computing the FRP of a route. IN: $\mathcal{P}'$, $\mathcal{P}''$. OUT: $\mathcal{P}^*$.

// Sum of the energy consumptions along the paths //
$\underline{\Delta}^* = \underline{\Delta}' + \underline{\Delta}''$
$\overline{\Delta}^* = \overline{\Delta}' + \overline{\Delta}''$
$\overline{\delta}^* = \overline{\delta}' + \overline{\delta}''$
$\underline{\delta}^* = \underline{\delta}' + \underline{\delta}''$
$\hat{\delta}^* = \hat{\delta}' + \hat{\delta}''$
$\tau^* \leftarrow \tau' + s_i/2 + t_e + s_j/2 + \tau''$
$\overline{c}^* \leftarrow \overline{c}' + \beta_i/2 + \beta_j/2 + \overline{c}''$
// Energy and time consumption along edge e //
$\underline{\Delta}^* \leftarrow \underline{\Delta}^* + d_e$
for $h \in \mathcal{H}$ **do**
  $\underline{\delta}^*_h \leftarrow \max\{\underline{\delta}^*_h, \underline{\Delta}^* - \sum_{k \in \mathcal{H}\setminus\{h\}} \overline{\delta}^*_k\}$
// Reference point update //
$Z = d_e$
for $h \in \mathcal{H}$ **do**
  $\epsilon \leftarrow \min\{Z, \overline{\delta}^*_h - \hat{\delta}^*_h\}$
  $\hat{\delta}^*_h \leftarrow \hat{\delta}^*_h + \epsilon$
  $Z \leftarrow Z - \epsilon$
  $\tau^* \leftarrow \tau^* + \rho_h \epsilon$
  $\overline{c}^* \leftarrow \overline{c}^* + \gamma_h \epsilon$
// Join energy replacement //
$h \leftarrow 0$
while $(\hat{\delta}^*_0 > B) and (\overline{c}^* < 0)$ **do**
  $h \leftarrow h + 1$
  $\epsilon \leftarrow \min\{\hat{\delta}^*_0 - B, \overline{\delta}^*_h - \hat{\delta}^*_h\}$
  $\hat{\delta}^*_h \leftarrow \hat{\delta}^*_h + \epsilon$
  $\hat{\delta}^*_0 \leftarrow \hat{\delta}^*_0 - \epsilon$
  $\tau^* \leftarrow \tau^* + \rho_h \epsilon$
  $\overline{c}^* \leftarrow \overline{c}^* + (\gamma_h - \gamma_0)\epsilon$

- Cost: $\overline{c}' + d_e\gamma^* + E^{join}(\gamma^* - \gamma_0) + \overline{c}'' - \beta_i/2 - \beta_j/2 < 0$.

When the route is not discarded by these tests based on lower bounds on duration and cost, its corresponding duration and cost must be computed exactly. For this purpose, we need to examine the FRP $\mathcal{P}^* = (\underline{\Delta}^*, \overline{\Delta}^*, \underline{\delta}^*, \overline{\delta}^*)$ produced when two paths with FRP $\mathcal{P}' = (\underline{\Delta}', \overline{\Delta}', \underline{\delta}', \overline{\delta}')$ and $\mathcal{P}'' = (\underline{\Delta}'', \overline{\Delta}'', \underline{\delta}'', \overline{\delta}'')$ are joined. It represents the set of all feasible recharge plans of the resulting route. Its computation is illustrated in Algorithm 2. In the first step the energy consumptions along the two paths are summed up. Then, the FRP is updated according to the additional consumption of energy d_e along the edge between the two path endpoints, as in Algorithm 1; the reference point coordinates are updated accordingly. Finally, the join energy is taken into account: the amount of energy coming from technology 0, i.e. $\overline{\delta}^*_0$, is decreased to B if needed. For this purpose the technologies are scanned from the cheapest to the costliest and the maximum amount of replacement is done and the duration τ^* and the reduced cost $\overline{c}^*$ of the route are increased accordingly.

If the algorithm exits the loop on line 23 with $\hat{\delta}^*_0 \leq B$ and $\overline{c}^* < 0$, then a feasible recharge plan with negative reduced cost may exist; on the contrary, if the algorithm exits the loop with $\overline{c}^* \geq 0$, then no feasible recharge plan has a negative reduced cost.

Algorithm 3 Technology replacement to make the route duration feasible

$D \leftarrow \{h \in \mathcal{H} : \hat{\delta}^*_h > \underline{\delta}^*_h\}$
$I \leftarrow \{h \in \mathcal{H} : \hat{\delta}^*_h < \overline{\delta}^*_h\}$
while $(\tau^* > T)$ and $(\overline{c}^* < 0)$ **do**
  $(h, k) \leftarrow \arg\min_{h \in D k \in I}\{\frac{\gamma_k - \gamma_h}{\rho_h - \rho_k}\}$
  $\epsilon \leftarrow \min\{\hat{\delta}^*_h - \underline{\delta}^*_h, \overline{\delta}^*_k - \hat{\delta}^*_k, \frac{\tau^* - T}{\rho_h - \rho_k}\}$
  $\hat{\delta}^*_h \leftarrow \hat{\delta}^*_h - \epsilon$
  $\hat{\delta}^*_k \leftarrow \hat{\delta}^*_k + \epsilon$
  $\tau^* \leftarrow \tau^* - (\rho_h - \rho_k)\epsilon$
  $\overline{c}^* \leftarrow \overline{c}^* + (\gamma_k - \gamma_h)\epsilon$
  $D \leftarrow D \cup \{k\}$
  if $\epsilon = \hat{\delta}^*_h - \underline{\delta}^*_h$ **then**
    $D \leftarrow D \backslash \{h\}$
  $I \leftarrow I \cup \{h\}$
  if $\epsilon = \overline{\delta}^*_k - \hat{\delta}^*_k$ **then**
    $I \leftarrow I \backslash \{k\}$

4.4.3 Repairing infeasible recharge plans

It may happen that the updated reference point of the FRP $\mathcal{P}^*$ is not feasible, because the duration of the route exceeds the prescribed limit T. In this case another point in the FRP must be found, trading time for cost, i.e. replacing slower and cheaper recharges with faster and more expensive ones.

Finding the minimum cost point in the FRP $\mathcal{P}^*$ satisfying the time constraint is a special linear programming problem that can be solved efficiently through Algorithm 3. Two subsets D and I contain the technologies h for which the amounts $\hat{\delta}^*_h$ can be decreased or increased respectively. The most profitable replacement between two tecnologies $h \in D$ and $k \in I$ is selected according to the minimum value of the ratio $\frac{\gamma_k - \gamma_h}{\rho_h - \rho_k}$, i.e. the ratio between the cost increase and the duration decrease. The values of τ^* and c^* are updated accordingly.

When the algorithm terminates with $\tau^* \geq T$ and $\overline{c}^* < 0$, a feasible recharge plan with negative reduced cost has been found; on the contrary, when it terminates with $\overline{c}^* \geq 0$, no feasible recharge plan with negative reduced cost exists.

Remark. Although this last step is needed to ensure the correctness of the pricing algorithm, in practice it is very unlikely to happen. In our computational tests we could never observe any occurrence.

4.5 Implementation and speed-up techniques

Pricing is the most time consuming step of the whole branch-and-price algorithm and in particular the join operation turns out to be the bottleneck. Hence, we devised and tested some ideas to avoid as many tentative join operations as possible.

Duplicates avoidance. In general each route can be generated by joining several different pairs of paths. To avoid useless duplicates, we introduced an additional test, that depends on the selected critical resource: the unbalance in the resource consumption between the two paths must be minimum along the resulting route.

For instance, assume capacity has been selected as critical and let ϕ' and ϕ'' be the capacity consumptions in vertices i and j, respectively. Then $\mathcal{L}'$ and $\mathcal{L}''$ are joined through edge $e = [i, j]$ only if

$$|\phi' - \phi''| \leq (q_i + q_j)/2.$$

Alternatively, assume time has been selected as critical and let τ' and τ'' be the time consumptions in vertices i and j, respectively. Then $\mathcal{L}'$ and $\mathcal{L}''$ are joined through edge $e = [i, j]$ only if

$$|\tau' - \tau''| \leq s_i/2 + t_e + \rho_{\mathcal{H}} d_e + s_j/2,$$

where $\rho_{\mathcal{H}}$ is the unit recharge time of the fastest technology.

Label extension in stages. Instead of generating all labels first and then trying to combine them in pairs, it is possible and profitable to have these two phases interleaved. A threshold l_{max} is set for the number of labels to be generated at each stage. When the number of labels reaches l_{max}, the label extension phase freezes and the algorithm starts tentatively joining the labels obtained so far. If no route are found in this way, the value of l_{max} is raised and another stage is executed.

In our implementation we heuristically set $l_{max} = 200k^3$ for the k^{th} stage.

Maximum number of routes. Another threshold s_{max} has been set for the number of routes to be returned by the pricer. When the number of generated feasible routes reaches s_{max}, the pricer stops This is especially effective if the most promising path pairs are examined first, as explained below.

In our implementation we heuristically set $s_{max} = 1000$.

Labels sorting. The labels of each vertex are sorted by non-decreasing reduced cost. The most promising path pairs correspond to nearby vertices and to the first labels of the vertices. Hence, tentative join operations are done according to the following order: first, the edge set is sorted and edges are examined by non-decreasing distance. Then, for each edge $e = [i, j]$ the label lists of i and j are scanned according to their order.

5 Branching

When the optimal solution of the LMP is fractional, two branching policies are used.

Branching on the number of routes. The total number of routes in the solution is upper bounded by m in one branch and lower bounded by $m + 1$ in the other, where m is the integer part of the number of routes occurring in the fractional solution.

This branching rule is applied first and, according to our computational experience, it is applicable only at the root node, because at non-root nodes the number of routes in the optimal fractional solution of the LMP is always integer.

Branching on customer pairs. Following the commonly used binary branching technique introduced by Ryan and Foster [4], a pair of customer vertices $u, v \in \mathcal{N}$ is selected such that $\sum_{r \in \Omega: y_{ur} = y_{vr} = 1} x_r$ is closest to $1/2$. Then, u and v are forced to be visited in the same route in one branch and they are forced to be visited by different routes in the other.

It is worth noting that such a rule is not in general *robust* with respect to the structure of the pricing problem. That is the pricing problem changes after branching decisions are taken, which then requires to either map branching decisions in weak form as master constraints, or to consider more complex pricing problems.

We decided for the second option, adapting the DP algorithm to get bounds which are as strong as possible. In particular, branching decisions imposing u and v to be visited in different routes are easy to embed in our setting: when u (resp. v) is visited, v (resp. u) is marked as unreachable. No additional change is required. Instead, when u and v are imposed to be visited in the same route, the DP labels need to be enriched by additional resources. In details, we keep in each label a set of *open* nodes which are forced to be visited before completing the route; when u (resp. v) is visited, v (resp. u) is inserted in such a set. Dominance can occur only if the set of open nodes of the dominanting label is a subset (possibly equal) of those of the dominated.

Finally, branching decisions are checked during join: those pairs of semi-routes violating them are simply discarded.

The Ryan-Foster rule is applied only to customer vertices: station vertices do not appear in the constraints of the LMP and they do not play any role concerning the integrality of the solution.

A well-known property of the Ryan-Foster rule guarantees that it is always applicable whenever the optimal solution of the LMP is fractional.

In our computational tests, the number of branchings needed to reach optimality never reached 10.

6 Computational results

6.1 Dataset

We did our experiments on three datasets. The full repository is available at [1] in xml format, compliant with the VRP-rep [14] specifications.

Remark. Input files include also two constants π and v, since energy and time consumption are assumed to be proportional to the distance along each edge through coefficients π and $1/v$, respectively. In other words, being l_e the lenght of an edge $e \in \mathcal{E}$, in our experiments we have $d_e = \pi l_e$ and $t_e = l_e/v$, although our algorithms do not rely on such a regularity.

Dataset A. This dataset was derived in [18] from the Solomon dataset, by relaxing the time windows constraints: instances have up to 15 customers (the last part of the name indicates the size of each instance) and 5 stations with a single technology. Some of these instances are very small and not challenging: we solved them mainly to use them as benchmarks and for the sake of coomparison between instances with single technology and multiple technologies.

Instances in dataset A are split in three classes: C (*clustered*), R (*random*) and RC (*random-clustered*), according to the distances between customers. This dataset is described in Table 1.

For some instances in this dataset we modified the number of vehicles with respect to the original value used in [18]. In one case this was done to make the instance feasible, because the original one was not [17]. In some other cases we decreased the number of vehicles to the minimum value for which the instance was known to be feasible [8].

Dataset B. Instances in this dataset have 10 customers, up to 5 vehicles, up to 9 stations and 3 technologies. Full details are given in [1].

Dataset C. This dataset was directly adapted from the Solomon dataset (clustered instances): all instances have 30 customers, 7 vehicles, 5 stations and 3 technologies. Full details are given in [1].

6.2 Computational tests

We implemented the branch-and-price algorithm in C++, using the SCIP framework [9] version 7.0.1 (linking CPLEX 20.1 for the LP subproblems). Upper bounds for the master problem were generated by general-purpose rounding heuristics embedded in SCIP. The algorithm was executed on a PC equipped with an AMD Ryzen 1950X 16-Core processor and 32 GB of RAM, running Linux Ubuntu 18.

Selection of the critical resource. The three criteria outlined in Subsection 4.4.1 were compared.

Table 1 contains our results on Dataset A. The table reports for each instance, in turn, the instance name, the number of customer and station nodes, the number of fleet vehicles, the vehicle capacity, the time duration limit and the battery limit.

The last block of four columns details the selection of the critical resource during pricing. Column c.r. reports the "correct answer", i.e. the critical resource that provides the best results. That was obtained by performing two independent runs of the algorithm, using either time or capacity as critical. The last three columns report instead the critical resource selected according to the three criteria described in subsection 4.4.1; in these columns 'T' stands for time, 'Q' stands for capacity. We do not report results for datasets B and C, because all algorithms always (correctly) predicted time as critical. Accuracy, Precision and Recall metrics are reported in Table 2. We denote by CT (resp. WT) the case in which T was the correct choice, and the criterion correctly predicted T (resp. predicted T but was instead Q). Symmetrically, we denote by CQ (resp. WQ) the case in which Q was the correct choice, and the criterion correctly predicted Q (resp. predicted Q but was instead T). Accuracy is measured as (CT + CQ) / (CT + CQ + WT + WQ); precision on T is measured as CT / (CT+WT); recall on T is measured as CT / (CT+WQ). Precision and recall on Q are measured in the same way, replacing T with Q. Criterion C proved best in all metrics, therefore we decided to use it in all our tests.

6.3 Results

To better appreciate the pros and cons of the branch-and-price algorithm described so far, we present its results in comparison with those obtained by another branch-and-price algorithm devised for the same problem in [3], where columns do not correspond to routes but to paths between recharge stations.

The route-based formulation presented here is characterized by a complex mixed-integer pricing sub-problem and a very simple master problem, while the path-based formulation of [3] has a pure integer pricing sub-problem and a much more involved mixed-integer master problem. The formulations are equivalent in the discrete domain, but their linear relaxations are not, because the linear relaxation of the path-based formulation allows for convex combinations of paths that are not routes.

We remark that the problem addressed in [3] is slightly different from the one considered here, because the number of vehicles to be used was fixed, rather than upper bounded. However, this minor difference does not hamper the conclusions that come from the comparison between the two algorithms.

In Tables 3, 4 and 5 we report the results obtained by the two algorithms with a timeout of two hours. Table columns indicate:

- the number of customer vertices, $|\mathcal{N}|$;
- the number of station vertices, $|\mathcal{R}|$;
- the number of avalilable vehicles, K;
- the number of calls to the pricing algorithm;
- the number of nodes in the branch-and-bound tree;
- the total computing time in seconds ($*$ indicates a two hours timeout).

Tables 6, 7 and 8 report the primal bound, the dual bound and the gap (computed as primal - dual, divided by dual) obtained when solving the problem to proven optimality with 2 hours timeout. The results of [3] are obtained by imposing an additional cutoff when optimality gap reaches 0.1%.

The results show that optimal solution is usually found by the route-based branch-and-price algorithm rather early, when only few nodes have been analysed. Indeed, for the smallest instances the optimal solution was found at the root node, without branching at all. This is due to the relative strength of the lower bound (compared with the path-based formulation), that in turn requires few branching steps to achieve integrality.

It is interesting to note that the most time-consuming instances for the route-based formulation are not the largest ones in datasets B and C, but some 15 customers instances in dataset A (namely, A-C208-15, A-R209-15 and A-RC204-15). This is due to the loose bound on time and capacity. Indeed, on two of these three instances (A-R209-15 and A-RC204-15) the algorithm reached the time-out at the root node: they have redundant duration and capacity constraints, so that their optimal solution requires a single vehicle. These are the only instances for which the route-based algorithm does not outperform the path-based one.

For some instances in dataset 7 and 8, such as C-3-N030, it looks inconsistent to have a primal bound (from the route-based formulation) smaller than a lower bound (from the path-based formulation). This is due to the mandatory usage of all vehicles in the path-based formulation: when the number of routes in the optimal solution is smaller than K, forcing the use of an extra vehicle worsens the optimal value.

7 Conclusions

In this paper we have described a branch-and-price algorithm for the EVRP problem with partial recharges and multiple technologies, employing bi-directional labeling algorithms for generating feasible routes.

In order to provide detailed computational insights we have considered data sets from the literature. The advantage of our method with respect to previous attempts emerges clearly: we have been able to solve all instances but two, with a computing effort which is a fraction of that of [3]. Still, the size of instances which can be solved to proven optimality (up to 30 customers) keeps appearing small compared to those solvable for other vehicle routing problems. Interestingly, the three instances left open by our algorithm are not among the largest ones. This confirms that the presence of continuous variables and multiple recharge technologies changes the structure of the problem, making it different in nature (and much more difficult) than the classical VRP, as already reported in the literature. The specific nature of such a difficulty, however, has still to be fully clarified.

Our computational tests gave other interesting insights. For instance, it appears that a path-based approach like that developed in [3] could be promising to solve single-vehicle EVRP variants, i.e. the "electric" equivalent of the TSP, whereas route-based approaches like ours are more powerful when fleets of vehicles need to be optimized.

Many extensions are possible both to make the model more realistic and to develop better algorithms. On the model side, besides incorporating constraints that also appear in classical VRP variants (e.g. time windows, heterogeneous fleets, pick-up and delivery, loading constraints etc.), it is worth to mention some extensions that are specific of the Electric VRP:

- synchronization constraints on simultaneous use of capacitated recharge stations by different vehicles;
- asymmetric graphs and negative cost arcs (corresponding to downhill roads, where energy can be accumulated instead of being consumed);

- variable (e.g. speed-dependent, time-dependent, load-dependent) energy consumption along the edges;
- time-dependent price of energy.

On the side of possible improvements to the algorithms, we mention the development of a parallel implementation of the join procedure in the pricing algorithm: once the bi-directional extension phase has been performed, each pair of vertices could be examined independently.

Acknowledgments The project has been partially funded by Regione Lombardia, grant agreement n. E97F17000000009, Project AD-COM.

References

[1] D. Bezzi. Electric Vehicle Routing Problem with Multiple Recharge Technologies (EVRP-MRT) Dataset. Technical report, UNIMI Davaverse, doi:10.13130/RD_UNIMI/JEA5XX, 2021.

[2] M. Bruglieri, S. Mancini, F.Pezzella, and O.Pisacane. A path-based solution approach for the green vehicle routing problem. *Computers and Operations Research*, 103:109–122, 2019.

[3] Alberto Ceselli, Angel Felipe, M. Teresa Ortuno, Giovanni Righini, and Gregorio Tirado. A branch-and-cut-and-price algorithm for the electric vehicle routing problem with multiple technologies. *Operations Research Forum*, 8, 2021.

[4] Ryan David and Brian A. Foster. An integer programming approach to scheduling. *Computer scheduling of public transport: Urban passenger vehicle and crew scheduling*, pages 269–280, 1981.

[5] Guy Desaulniers, Fausto Errico, Stefan Irnich, and Michael Schneider. Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research*, 64:1388–1405, 2016.

[6] Moshe Dror and Pierre Trudeau. Split delivery routing. *Naval Research Logistics*, 37:383–402, 1990.

[7] Dominique Feillet, Pierre Dejax, Michel Gendreau, and Cyrille Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44:216–229, 2004.

[8] Angel Felipe, M. Teresa Ortuno, Giovanni Righini, and Gregorio Tirado. A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transportation Research Part E: Logistics and Transportation Review*, 71(C):111–128, 2014.

[9] Gerald Gamrath, Daniel Anderson, Ksenia Bestuzheva, Wei-Kun Chen, Leon Eifler, Maxime Gasse, Patrick Gemander, Ambros Gleixner, Leona Gottwald, Katrin Halbig, Gregor Hendel, Christopher Hojny, Thorsten Koch, Pierre Le Bodic, Stephen J. Maher, Frederic Matter, Matthias Miltenberger, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Christine Tawfik, Stefan Vigerske, Fabian Wegscheider, Dieter Weninger, and Jakob Witzig. The SCIP Optimization Suite 7.0. Technical report, Optimization Online, March 2020.

[10] Stefan Irnich and Guy Desaulniers. *Shortest path problems with resource constraints*, pages 33–65. Springer, 2005.

[11] Merve Keskin, Gilbert Laporte, and Bulent Catay. Electric vehicle routing problem with time-dependent waiting times at recharging stations. *Computers & Operations Research*, 107:77–94, 2019.

[12] Cagri Koc, Ola Jabali, and Gilbert Laporte. Long-haul vehicle routing and scheduling with idling options. *Journal of the Operational Research Society*, 69(2):235–246, 2018.

[13] Wang Li-ying and Yuanbin Song. Multiple charging station location-routing problem with time window of electric vehicle. *Journal of Engineering Science and Technology Review*, 8:190–201, 11 2015.

[14] J.E. Mendoza, C. Guéret, M. Hoskins, H. Lobit, V. Pillac, T. Vidal, and D. Vigo. Vrp-rep: the vehicle routing community repository. In *Third meeting of the EURO Working Group on Vehicle Routing and Logistics Optimization (VeRoLog). Oslo, Norway*, 2014.

[15] Samuel Pelletier, Ola Jabali, and Gilbert Laporte. Goods distribution with electric vehicles: Review and research perspectives. *Transportation Science*, 50:3–22, 2015.

[16] 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. working paper or preprint, October 2014.

[17] Michael Schneider. Personal communication, 2014.

[18] Michael Schneider, Andreas Stenger, and Dominik Goeke. The electric vehicle routing problem with time windows and recharging stations. *Transportation Science*, 48:500–520, 2014.

[19] Mariangela Scorrano, Romeo Danielis, and Marco Giansoldati. Electric light commercial vehicles for a cleaner urban goods distribution. are they cost competitive? *Research in Transportation Economics*, 85:101022, 2021. Policies prompting sustainable transport in urban areas.

Table 1: Critical resource of instances in Dataset A

instance	$\|\mathcal{N}\|$	$\|\mathcal{R}\|$	K	Q	T	B	c.r.	A	B	C
A-C101-10	10	5	2	200	1236	77	T	T	T	T
A-C101-5	5	3	2	90	1236	77	Q	Q	Q	Q
A-C103-15	15	5	2	200	1236	77	T	T	T	T
A-C103-5	5	2	1	90	1236	77	Q	T	Q	Q
A-C104-10	10	4	2	180	1236	77	T	Q	Q	Q
A-C106-15	15	3	3	170	1236	77	T	Q	Q	T
A-C202-10	10	5	1	220	3390	77	Q	T	Q	Q
A-C202-15	15	5	2	240	3390	77	Q	T	Q	Q
A-C205-10	10	3	2	180	3390	77	Q	T	Q	Q
A-C206-5	5	4	1	70	3390	77	Q	T	Q	Q
A-C208-15	15	4	2	250	3390	77	Q	T	Q	Q
A-C208-5	5	3	1	100	3390	77	Q	T	Q	Q
A-R102-10	10	4	3	155	230	60	T	Q	Q	T
A-R102-15	15	8	5	191	230	60	T	Q	Q	T
A-R103-10	10	3	2	139	230	60	T	Q	Q	Q
A-R104-5	5	3	2	86	230	60	T	Q	Q	Q
A-R105-15	15	6	3	200	230	60	T	T	T	T
A-R105-5	5	3	2	58	230	60	T	Q	Q	Q
A-R201-10	10	4	1	181	1000	60	Q	T	Q	Q
A-R202-15	15	6	2	261	1000	60	Q	T	Q	Q
A-R202-5	5	3	1	61	1000	60	Q	T	Q	Q
A-R203-10	10	5	1	109	1000	60	Q	T	Q	Q
A-R203-5	5	4	1	83	1000	60	Q	T	Q	Q
A-R209-15	15	5	1	247	1000	60	Q	Q	Q	Q
A-RC102-10	10	4	3	181	240	77	T	Q	Q	Q
A-RC103-15	15	5	3	200	240	77	T	T	T	T
A-RC105-5	5	4	2	145	240	77	T	Q	Q	T
A-RC108-10	10	4	3	146	240	77	T	Q	Q	Q
A-RC108-15	15	5	3	200	240	77	T	T	T	T
A-RC108-5	5	4	2	109	240	77	T	Q	Q	Q
A-RC201-10	10	4	1	160	960	77	Q	T	Q	Q
A-RC202-15	15	5	2	233	960	77	Q	T	Q	Q
A-RC204-15	15	7	1	287	960	77	Q	T	Q	Q
A-RC204-5	5	4	1	130	960	77	Q	T	Q	Q
A-RC205-10	10	4	2	154	960	77	Q	T	Q	Q
A-RC208-5	5	3	1	82	960	77	Q	T	Q	Q

Table 2: Comparison between critical resource selection criteria

Criterion	Accuracy	Precision Q	Precision T	Recall Q	Prec. T
A	0.19	0.10	0.31	0.15	0.22
B	0.69	1.00	0.31	0.64	1.00
C	0.81	1.00	0.56	0.74	1.00

Table 3: Solutions for dataset A

Instance	$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{R}\rvert$	K	Pricer calls		B&B nodes		Time (s)	
				path	route	path	route	path	route
A-C101-10	10	5	2	36015	11	71301	1	4386	0.74
A-C101-5	5	3	2	320	2	591	1	1.72	0.01
A-C103-15	15	5	2	8092	18	15875	1	2871	20.86
A-C103-5	5	2	1	43	2	79	1	0.09	0.03
A-C104-10	10	4	2	3665	15	7201	1	165	0.95
A-C106-15	15	3	3	5778	66	11455	3	794	287.67
A-C202-10	10	5	1	788	30	1551	1	53.2	28.08
A-C202-15	15	5	2	261	35	489	1	67.1	130.60
A-C205-10	10	3	2	933	10	1802	1	22	0.56
A-C206-5	5	4	1	5	7	1	1	0.04	0.03
A-C208-15	15	4	2	295	154	438	2	29.7	7200*
A-C208-5	5	3	1	73	5	125	1	0.28	0.03
A-R102-10	10	4	3	3672	6	7207	1	198	0.03
A-R102-15	15	8	5	901	128	862	9	7200*	8.13
A-R103-10	10	3	2	439	11	793	1	10.07	3.72
A-R104-5	5	3	2	313	3	611	1	1.47	0.03
A-R105-15	15	6	3	15077	33	15656	3	7200*	1.58
A-R105-5	5	3	2	975	4	1928	1	4.96	0.03
A-R201-10	10	4	1	100	36	163	1	2.12	10.28
A-R202-15	15	6	2	156	32	208	1	62.7	48.25
A-R202-5	5	3	1	84	2	155	1	0.20	0.03
A-R203-10	10	5	1	9	30	1	1	0.55	3.24
A-R203-5	5	4	1	34	4	59	1	0.19	0.04
A-R209-15	15	5	1	28	129	32	1	3.31	7200*
A-RC102-10	10	4	3	77513	16	152467	3	4822	0.21
A-RC103-15	15	5	3	40870	15	45417	1	7200*	0.13
A-RC105-5	5	4	2	6444	4	12843	1	65.6	0.01
A-RC108-10	10	4	3	33827	4	66936	1	2075	0.06
A-RC108-15	15	5	3	34319	32	43382	3	7200*	0.37
A-RC108-5	5	4	2	2647	3	5031	1	18.1	0.01
A-RC201-10	10	4	1	284	26	547	1	9.82	6.25
A-RC202-15	15	5	2	5017	39	9499	1	1336	67.52
A-RC204-15	15	7	1	1237	113	2305	1	863	7200*
A-RC204-5	5	4	1	209	6	401	1	1.07	0.04
A-RC205-10	10	4	2	10146	10	20177	1	443	0.21
A-RC208-5	5	3	1	118	4	212	1	0.43	0.02

Table 4: Solutions for dataset B

Instance	$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{R}\rvert$	K	Pricer calls		B&B nodes		Time (s)	
				path	route	path	route	path	route
B-10-N10	10	9	4	560	19	635	3	4740	0.10
B-11-N10	10	9	4	280	8	366	1	3664	0.02
B-12-N10	10	9	4	813	6	814	1	7200*	0.03
B-13-N10	10	9	4	523	6	526	1	7200*	0.04
B-14-N10	10	9	5	856	17	826	3	7200*	0.07
B-15-N10	10	9	4	110	15	128	3	1244	0.11
B-16-N10	10	9	4	379	57	433	11	7200*	0.60
B-17-N10	10	9	5	373	20	389	3	4174	0.07
B-18-N10	10	9	5	336	15	414	3	4544	0.06
B-19-N10	10	9	4	185	41	255	7	3888	0.37
B-20-N10	10	5	4	167	10	172	3	101	0.15
B-21-N10	10	5	4	279	16	399	3	379	0.32
B-22-N10	10	5	4	1739	21	3116	5	2614	0.21
B-23-N10	10	5	3	3660	52	3694	9	7200*	0.24
B-24-N10	10	5	4	5223	12	6396	3	7200*	0.22
B-25-N10	10	5	3	3280	7	3260	1	7200*	0.08
B-26-N10	10	5	4	5190	12	8993	3	7200*	0.12
B-27-N10	10	5	4	2118	35	3675	7	1760	0.17
B-28-N10	10	5	4	185	11	309	3	110	0.09
B-29-N10	10	5	4	1579	16	2141	3	744	0.20

Table 5: Solutions for dataset C

Instance	$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{R}\rvert$	K	Pricer calls		B&B nodes		Time (s)	
				path	route	path	route	path	route
C-0-N030	30	5	7	57	72	84	5	849	1.79
C-1-N030	30	5	6	2334	261	2350	13	7200*	7.62
C-2-N030	30	5	6	1429	79	1426	5	7200*	1.70
C-3-N030	30	5	7	1883	54	2152	3	6910	0.62
C-4-N030	30	5	7	1475	143	1631	9	7200*	2.30
C-5-N030	30	5	6	14	446	4	19	7200*	32.49
C-6-N030	30	5	6	1218	291	1307	15	4611	10.18
C-7-N030	30	5	6	6	24	1	1	247	0.29
C-8-N030	30	5	6	167	96	170	5	1125	1.72
C-9-N030	30	5	6	61	20	63	1	951	0.24

Table 6: Full B&B results for dataset A

Instance	PB		DB		Gap	
	path	route	path	route	path	route
A-C101-10	303.00	303.00	302.70	303.00	0.10%	0.00%
A-C101-5	214.00	214.00	214.00	214.00	0.00%	0.00%
A-C103-15	290.00	290.00	289.71	290.00	0.10%	0.00%
A-C103-5	157.00	157.00	157.00	157.00	0.00%	0.00%
A-C104-10	281.00	281.00	280.72	281.00	0.10%	0.00%
A-C106-15	253.00	253.00	252.75	253.00	0.10%	0.00%
A-C202-10	234.00	234.00	233.79	234.00	0.09%	0.00%
A-C202-15	332.00	332.00	331.67	332.00	0.10%	0.00%
A-C205-10	233.00	233.00	232.83	233.00	0.07%	0.00%
A-C206-5	205.00	205.00	205.00	205.00	0.00%	0.00%
A-C208-15	269.00	269.00	268.75	265.00	0.09%	1.51%
A-C208-5	161.00	161.00	161.00	161.00	0.00%	0.00%
A-R102-10	230.00	230.00	229.81	230.00	0.08%	0.00%
A-R102-15	345.00	317.00	269.22	317.00	28.15%	0.00%
A-R103-10	171.00	169.00	168.17	169.00	1.68%	0.00%
A-R104-5	142.00	142.00	142.00	142.00	0.00%	0.00%
A-R105-15	308.00	297.00	238.67	297.00	29.05%	0.00%
A-R105-5	160.00	160.00	159.88	160.00	0.08%	0.00%
A-R201-10	189.00	189.00	188.92	189.00	0.04%	0.00%
A-R202-15	286.00	286.00	285.86	286.00	0.05%	0.00%
A-R202-5	147.00	147.00	147.00	147.00	0.00%	0.00%
A-R203-10	252.00	252.00	252.00	252.00	0.00%	0.00%
A-R203-5	185.00	185.00	185.00	185.00	0.00%	0.00%
A-R209-15	264.00	376.00	264.00	$*$	0.00%	$*$
A-RC102-10	428.00	428.00	427.57	428.00	0.10%	0.00%
A-RC103-15	387.00	367.00	296.67	367.00	30.45%	0.00%
A-RC105-5	220.00	220.00	219.80	220.00	0.09%	0.00%
A-RC108-10	355.00	355.00	354.65	355.00	0.10%	0.00%
A-RC108-15	384.00	384.00	358.50	384.00	7.11%	0.00%
A-RC108-5	259.00	259.00	258.74	259.00	0.10%	0.00%
A-RC201-10	258.00	258.00	257.79	258.00	0.08%	0.00%
A-RC202-15	305.00	305.00	304.70	305.00	0.10%	0.00%
A-RC204-15	295.00	484.00	294.72	$*$	0.09%	$*$
A-RC204-5	182.00	182.00	182.00	182.00	0.00%	0.00%
A-RC205-10	320.00	320.00	319.71	320.00	0.09%	0.00%
A-RC208-5	172.00	172.00	172.00	172.00	0.00%	0.00%

Table 7: Full B&B results for dataset B

Instance	PB		DB		Gap	
	path	route	path	route	path	route
B-10-N10	22.92	22.92	22.91	22.92	0.06%	0.00%
B-11-N10	24.86	24.86	24.83	24.86	0.10%	0.00%
B-12-N10	24.15	23.51	22.93	23.51	5.31%	0.00%
B-13-N10	25.04	23.70	21.99	23.70	13.85%	0.00%
B-14-N10	$*$	31.26	28.98	31.26	$*$	0.00%
B-15-N10	21.13	20.24	21.11	20.24	0.08%	0.00%
B-16-N10	20.20	20.20	19.73	20.20	2.38%	0.00%
B-17-N10	28.88	28.11	28.86	28.11	0.07%	0.00%
B-18-N10	28.07	25.39	28.05	25.39	0.07%	0.00%
B-19-N10	20.11	20.11	20.11	20.11	0.04%	0.00%
B-20-N10	19.24	19.24	19.22	19.24	0.08%	0.00%
B-21-N10	20.29	19.99	20.27	19.99	0.10%	0.00%
B-22-N10	20.17	20.17	20.15	20.17	0.10%	0.00%
B-23-N10	$*$	23.21	20.86	23.21	$*$	0.00%
B-24-N10	22.60	22.46	21.33	22.46	5.94%	0.00%
B-25-N10	$*$	23.26	20.59	23.26	$*$	0.00%
B-26-N10	20.98	20.98	20.85	20.98	0.64%	0.00%
B-27-N10	24.49	24.49	24.47	24.49	0.10%	0.00%
B-28-N10	22.51	22.51	22.49	22.51	0.09%	0.00%
B-29-N10	22.26	22.26	22.24	22.26	0.09%	0.00%

Table 8: Full B&B results for dataset C

Instance	PB		DB		Gap	
	path	route	path	route	path	route
C-0-N030	39.01	39.01	38.99	39.01	0.04%	0.00%
C-1-N030	1918.43	39.78	37.66	39.78	4993%	0.00%
C-2-N030	$*$	36.76	36.43	36.76	$*$	0.00%
C-3-N030	41.62	40.44	41.58	40.44	0.10%	0.00%
C-4-N030	43.08	42.72	41.70	42.72	3.30%	0.00%
C-5-N030	$*$	31.80	31.48	31.80	$*$	0.00%
C-6-N030	37.62	37.62	37.59	37.62	0.10%	0.00%
C-7-N030	30.64	30.64	30.64	30.64	0.00%	0.00%
C-8-N030	33.83	33.83	33.83	33.83	0.01%	0.00%
C-9-N030	33.53	33.52	33.51	33.52	0.05%	0.00%