Transportation Research Part E 89 (2016) 32–52

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

The Electric Traveling Salesman Problem with Time Windows

R. Roberti [a,*], M. Wen [b]

[a] *Technical University of Denmark, 2800 Kgs. Lyngby, Denmark*
[b] *Xi'an Jiaotong-Liverpool Univeristy, 111 Ren Ai Road, Suzhou, Jiangsu 215123, China*

ARTICLE INFO

Article history:
Received 3 July 2015
Received in revised form 14 December 2015
Accepted 18 January 2016
Available online 4 March 2016

Keywords:
Electric vehicles
Green logistics
Traveling Salesman
General Variable Neighborhood Search
Dynamic Programming
Time windows

ABSTRACT

To minimize greenhouse gas emissions, the logistic field has seen an increasing usage of electric vehicles. The resulting distribution planning problems present new computational challenges.

We address a problem, called *Electric Traveling Salesman Problem with Time Windows*. We propose a mixed integer linear formulation that can solve 20-customer instances in short computing times and a Three-Phase Heuristic algorithm based on General Variable Neighborhood Search and Dynamic Programming.

Computational results show that the heuristic algorithm can find the optimal solution in most small-size instances within a tenth of a second and achieves goods solutions in instances with up to 200 customers.

1. Introduction

According to US Environmental Protection Agency (2015), the total US greenhouse gas emissions amounted for 6673 million metric tons CO_2 equivalent mass units in 2013, showing an increase of 5.9% from 1990. Approximately 82.5% of total greenhouse gas emissions by human activities was CO_2. Along with passenger cars, which generated 42.7% of CO_2 emissions, the second largest source of CO_2 emissions in transportation was freight trucks (22.8%). To tackle this environmental problem, in the logistic field, *Electric Commercial Vehicles* (ECVs) are considered as a valid alternative to *Internal Combustion Commercial Vehicles* (ICCVs) because they are environmentally friendly and produce minimal noise.

Due to these practical considerations along with political factors (e.g., in 2009, the US Government granted 2.4 billion dollars to "accelerate the manufacturing and deployment of the next generation of US batteries and electric vehicles", see US Department of Energy (2009)), ECVs are more and more common in last-mile delivery distribution, for example in small-package shipping or in the distribution of food and beverages, and several companies have started deploying ECVs for their daily operations (see FedEx, 2010; Motavalli, 2010). This gives birth to a whole new field of research concerning the conditions under which ECVs are more convenient than ICCVs.

A recent study by Davis and Figliozzi (2013) compared the overall costs of three different vehicles, one diesel truck and two electric trucks, over a long planning horizon and showed that electric vehicles are competitive especially when the traveling distance is long, congestion is prevalent, and customer stops are frequent. Davis and Figliozzi (2013) also pointed out the importance of efficient and tailored distribution plans when utilizing electric vehicles. The *Vehicle Routing Problems* (VRPs) arising when dealing with ECVs present new challenges for researchers and practitioners who want to provide such optimized distribution plans.

* Corresponding author.
E-mail addresses: rorob@transport.dtu.dk (R. Roberti), Min.Wen@xjtlu.edu.cn (M. Wen).

http://dx.doi.org/10.1016/j.tre.2016.01.010
1366-5545/

The literature about optimization methods for traditional ICCVs is rich (see Vigo and Toth (2014) for a comprehensive survey on the topic), while only few recent papers provide optimization algorithms for electric VRPs. One of the seminal papers on the subject can be considered Schneider et al. (2014), where a hybrid heuristic combining *Variable Neighborhood Search* (VNS) and tabu search is proposed to solve the *Electric Vehicle Routing Problem with Time Windows and Recharging Stations* (E-VRPTW). The E-VRPTW is the problem of defining a least-cost distribution plan for capacitated electric vehicles, located at a central depot, that are used to satisfy the demands of a set of customers within given time windows; because of the limited capacity of the batteries of such vehicles, stops at recharging stations may be needed along the routes.

In this paper, we address the single-vehicle version of the E-VRPTW under two recharging policies: *full* (the battery is fully recharged at each stop) and *partial* (any amount can be recharged at each stop). To the best of our knowledge, this problem, which we call *Electric Traveling Salesman Problem with Time Windows* (E-TSPTW), has not been addressed in the literature yet. The E-TSPTW can be easily stated as the problem of finding a shortest Hamiltonian tour for visiting a set of customers within given time windows in such a way that the battery level is always non-negative – this can be achieved by stopping at intermediate recharging stations to recharge the battery. The E-TSPTW is a generalization of the well-known and well-studied *Traveling Salesman Problem with Time Windows* (TSPTW), so it is also NP-hard.

Although the market share of electric vehicles is still limited in many countries today, the deployment of electric freight vehicles is expected to grow because of upcoming restrictions on vehicle emissions and because more and more incentives have been provided for using environmentally-friendly vehicles. The application of the E-TSPTW is potentially wide, in particular in last-mile delivery of parcels in urban areas. In FREVUE's reports (Nesterova et al., 2013), a two-phase delivery is considered as an interesting logistics concept for electric freight vehicles. Goods are first sent to an urban consolidation center (UCC), a logistics facility that is close to the denser urban area, by conventional trucks and later are transferred to the electric vehicles for last mile deliveries. This concept has been successfully adopted in many cities, such as Leiden, Bristol, Malaga and La Rochelle, in all of which the electric vehicles are used for transport in the city center zone (van Duin et al., 2010). Some large logistic companies, e.g. FedEx, have also put electric vehicles into use for delivering parcels in urban areas (FedEx, 2010). It can be further expected that when charging infrastructure is deployed along routes connecting cities and when the driving range of electric vehicles is extended, the intercity parcel delivery will gain its momentum (Pelletier et al., in press).

Having efficient solution methods for the E-TSPTW is important not only for solving practical applications of the problem, but also for solving more involved Electric VRPs. It is well-known (see Desaulniers et al., 2005; Vigo and Toth, 2014) that the state-of-the-art exact algorithms for a wide range of VRPs are based on the column generation framework. The most relevant issue when developing these algorithms is arguably the resolution of the pricing problem, which is usually an NP-hard problem for which exact algorithms are time-consuming. Therefore, many column generation algorithms generate columns by means of heuristic algorithms and rely on exact methods only at the very last iterations. We believe that future exact algorithms for Electric VRPs will rely on column generation (a first example from the literature is the exact method of Desaulniers et al. (2014) for the E-VRPTW), and the resulting pricing problems will present most of the challenges tackled when solving the E-TSPTW considered in this paper.

The main contributions of this paper are the following. We define the E-TSPTW and model it as a compact *Mixed Integer Linear Problem* (MILP). We propose an alternative MILP model, both for the full and the partial recharge policies, that has an exponential number of variables (with respect to the number of recharging stations) and defined several rules to limit the number of variables necessary to achieve an optimal solution. Then, we describe a Three-Phase Heuristic algorithm based on *General VNS* (GVNS) and Dynamic Programming to find near-optimal solutions of the E-TSPTW, where simple adaptations are required to consider the full recharge policy instead of the partial recharge policy (and vice versa). Finally, we introduce two sets of benchmark instances derived from well-known TSPTW instances from the literature and show the computational performance of the proposed MILP model and of the Three-Phase Heuristic algorithm for both recharging policies.

The rest of the paper is organized as follows. Section 2 is the literature review. A formal definition of the E-TSPTW and a compact MILP formulation are reported in Section 3. The alternative formulation with exponentially many variables is illustrated in Section 4. The proposed Three-Phase Heuristic algorithm is developed in Section 5. Section 6 reports the computational results. Some conclusions are drawn in Section 7.

2. Literature review

The E-TSPTW is a generalization of the well-known TSPTW. The TSPTW has been extensively addressed in the literature both with exact and heuristic methods. Gendreau et al. (1998) proposed an insertion heuristic that gradually builds the solution by inserting a vertex in its neighborhood and performing a local re-optimization, and, once a feasible solution is achieved, tries to improve it through removal and reinsertion of all vertices. Ohlmann and Thomas (2007) described a variant of Simulated Annealing, called Compressed Annealing, that embeds a variable penalty method to consider time windows constraints that are relaxed and stochastic search. A hybrid method that combines Beam search with Ant Colony Optimization, called Beam-ACO, was proposed by López-Ibáñez and Blum (2010).

More recently, da Silva and Urrutia (2010) and Mladenović et al. (2012) showed the potential of solving the TSPTW by mean of GVNS and proposed two heuristic algorithms that can be considered the state-of-the-art for solving the TSPTW. The algorithm of da Silva and Urrutia (2010) is composed of a constructive stage followed by an optimization stage. In the constructive stage, the goal is to achieve a feasible TSPTW solution by using a VNS; in the optimization phase, the solution returned in the first phase is improved with a GVNS heuristic. Mladenović et al. (2012) described another efficient GVNS that differs from the algorithm of da Silva and Urrutia (2010) because more efficient data structures and different neighborhoods are used.

An alternative heuristic approach that is competitive with the algorithms of da Silva and Urrutia (2010) and Mladenović et al. (2012) is the Variable Iterated Greedy algorithm of Karabulut and Fatih Tasgetiren (2014).

A variety of different approaches can be found in the literature to solve the TSPTW to optimality: branch-and-cut algorithms (Ascheuer et al., 2001; Dash et al., 2010), constraints-programming-based methods (Focacci et al., 2002), and dynamic programming recursions (Dumas et al., 1995; Mingozzi et al., 1997; Balas and Simonetti, 2001). The state-of-the-art exact algorithm can be considered the column generation based method of Baldacci et al. (2011), where different state space relaxation techniques are used to derive tight lower bounds and are combined with dynamic programming to find optimal solutions; this solution method was able to close all but one test instance (with up to 233 vertices) from the literature in reasonable computing times.

Another stream of research closely related to our work is optimization methods for VRPs of the electric/alternative fuel vehicles. Conrad and Figliozzi (2011) studied the *Recharging VRP* where vehicles with limited driving range are allowed to recharge en route at certain customer locations. Customer service time windows, full recharge and fixed charging times are assumed in the problem. The authors developed an iterated route construction and improvement algorithm and investigated the impact of driving range, recharging time and time window existence on the solutions. Erdoğan and Miller-Hooks (2012) introduced the *Green Vehicle Routing Problem* (G-VRP), in which the time window and the vehicle capacity are not considered, the vehicle can only be charged to full capacity when it visits a station, and the charging time is assumed to be constant. Felipe et al. (2014) extended the GVRP by considering different types of recharging stations with different costs and recharging speeds. Furthermore, in their problem, partial charging is allowed and the recharging time is assumed to be a linear function of the amount of energy recharged, which brings extra decision variables in the problem on how much capacity to be charged at the station. Several local search methods as well as a Simulated Annealing heuristic were developed.

Schneider et al. (2014) extended the G-VRP to the E-VRPTW by considering the customer time windows, unlimited number of recharging per route and a variable recharging time which depends on the remaining fuel level when a vehicle arrives at the recharging station. They developed a hybrid VNS and Tabu Search heuristic. Preis et al. (2014) extended the E-VRPTW by considering a load-dependent energy consumption. A Tabu Search heuristic was presented to minimize the overall energy consumption. Hiermann et al. (submitted for publication) addressed the Electric Fleet Size and Mix VRPTW, a combination of E-VRPTW and Fleet Size Mix VRPTW, where different types of electric vehicles with different battery capacities, load capacities, energy consumptions and recharging rates are considered. Goeke and Schneider (2015) extended the E-VRPTW to a mixed fleet consisting of electric vehicles and conventional internal combustion commercial vehicles. In contrast to a simple traveling-distance-dependent energy consumption assumed in most of the existing studies, a more realistic energy consumption incorporating the factors of speed, gradient and cargo load was adopted. Both Hiermann et al. (submitted for publication) and Goeke and Schneider (2015) developed Adaptive Large Neighborhood Search heuristics to solve their problems. Desaulniers et al. (2014) focused on the exact algorithms for four variations of the E-VRPTW, which differ in the number of allowed recharges per route (single or multiple) and the type of recharge (partial or full). Their branch-price-and-cut algorithms can solve most of the instances involving up to 100 customers with narrow time windows to optimality.

3. A compact formulation of the E-TSPTW

In this section, we formally introduce the E-TSPTW along with the notation used throughout the paper. We also model the E-TSPTW with a compact MILP formulation derived from the MILP formulation described by Schneider et al. (2014) for the E-VRPTW. The presented formulation is a simple adaption of the formulation of Schneider et al. (2014) obtained by ignoring capacity constraints and by considering one vehicle only.

3.1. Definition of the E-TSPTW and notation

Let V be a set of vertices defined as $V = \{o, d\} \cup C \cup S$, where vertex o (d, respectively) is the initial (final, resp.) vertex of the tour to find, C is a set of n customers to visit, and S is a set of m recharging stations. Vertex d is simply a copy of vertex o that is needed for notational purposes. Moreover, let $C_o, C_d, C_{od} \subset V$ be three subsets of the set V defined as $C_o = C \cup \{o\}, C_d = C \cup \{d\}, C_{od} = C \cup \{o, d\}$.

A time window $[e_i, l_i]$ is associated with each vertex $i \in C_{od}$, where $e_i \in \mathbb{Z}_+$ ($l_i \in \mathbb{Z}_+$, resp.) represents the earliest (latest, resp.) time when service at vertex i can start; as commonly done in the literature, we assume that time windows are *hard*, meaning that a vertex can be visited before time e_i (in this case, the service is delayed to time e_i) but not later than l_i. Furthermore, we assume that recharging stations are always available over the planning horizon, so no time window constraints are imposed on them.

Let A be a set of arcs defined as $A = A_C \cup A_S$, where $A_C = \{(i,j) : i \in C_o, j \in C, i \neq j\} \cup \{(i,j) : i \in C, j \in C_d, i \neq j\}$ and $A_S = \{(i,j) : i \in C_o, j \in S\} \cup \{(i,j) : i \in S, j \in C_d\} \cup \{(i,j) : i,j \in S, i \neq j\}$. Travel distance d_{ij}, travel time $t_{ij} \in \mathbb{Z}_+$, and battery consumption $q_{ij} \in \mathbb{Z}_+$ are associated with each arc $(i,j) \in A$. The capacity of the battery is denoted with Q. To simplify the notation throughout the paper, we can assume, without loss of generality, that travel time t_{ij} includes the service time at vertex i.

As done in Schneider et al. (2014), we make the following assumptions:

(1) the battery consumption for traversing arc $(i,j) \in A$ is proportionally linear to the travel distance d_{ij} with respect to a consumption rate h, that is, $q_{ij} = h \cdot d_{ij}$;
(2) the recharging time is proportionally linear to the desired quantity to recharge with respect to a recharging rate g;
(3) the consumption rate h is equal for all arcs;
(4) the recharging rate g is equal for all stations.

Moreover, we assume that multiple recharges can be performed at the same station along the tour and that vertex o can act as a recharging station.

In the literature, two policies are usually considered to determine the amount of battery recharged at each stop: full and partial. In the full-recharge policy, the battery is always fully recharged, while, in the partial-recharge policy, any quantity can be recharged at each stop as long as the capacity of the battery is not exceeded. Throughout the paper, we consider both policies by, first, addressing the full-recharge policy and, then, showing what needs to be changed to handle the partial-recharge policy.

The E-TSPTW is the problem of finding a shortest tour of graph $G = (V, A)$ that starts from vertex o, visits all customers of the set C within their time windows, ends at d, possibly stops at recharging stations within their time windows, and such that the battery level is always between 0 and Q.

3.2. A compact formulation for the full-recharge policy

The compact formulation illustrated in this section uses graph $\widehat{G} = (\widehat{V}, \widehat{A})$, where the set of vertices $\widehat{V}$ and the arc set $\widehat{A}$ are defined as follows. Let $\widehat{S}$ be a set of dummy stations containing multiple copies of each recharging station of the set S (where each copy represents a different visit to the corresponding recharging station), and let $\widehat{V} = \{o, d\} \cup C \cup \widehat{S}$ be the set of vertices of graph $\widehat{G}$. The set of arcs $\widehat{A}$ is defined as $\widehat{A} = \{(o,j) : j \in C \cup \widehat{S}\} \cup \{(i,d) : i \in C \cup \widehat{S}\} \cup \{(i,j) : i,j \in C \cup \widehat{S}, i \neq j, e_i + t_{ij} \leqslant l_j\}$.

By introducing the following decision variables:

- $x_{ij} \in \{0,1\}$: binary variable equal to 1 if arc $(i,j) \in \widehat{A}$ is traversed (0 otherwise);
- $z_i \in \mathbb{Z}_+$: time when the service at vertex $i \in C_{od}$ starts and when the recharge at vertex $i \in \widehat{S}$ starts (undefined if $i \in \widehat{S}$ is not visited);
- $y_i \in \mathbb{Z}_+$: battery level upon arriving at vertex $i \in \widehat{V}$ (undefined if $i \in \widehat{S}$ is not visited);

the E-TSPTW, when the full-recharge policy is applied, can be formulated as

$$z^* = \min \sum_{(i,j) \in \widehat{A}} d_{ij} x_{ij} \tag{3.1}$$

$$\text{s.t.} \sum_{(i,j) \in \widehat{A}} x_{ij} = 1 \qquad i \in C_o \tag{3.2}$$

$$\sum_{(i,j) \in \widehat{A}} x_{ij} \leqslant 1 \qquad i \in \widehat{S} \tag{3.3}$$

$$\sum_{(i,k) \in \widehat{A}} x_{ik} = \sum_{(k,j) \in \widehat{A}} x_{kj} \qquad k \in C \cup \widehat{S} \tag{3.4}$$

$$z_o = e_o \tag{3.5}$$

$$e_i \leqslant z_i \leqslant l_i \qquad i \in C_d \tag{3.6}$$

$$z_i + (t_{ij} + M) x_{ij} \leqslant z_j + M \qquad (i,j) \in \widehat{A} : i \in C_o \tag{3.7}$$

$$z_i + (t_{ij} + M + gQ) x_{ij} - g y_i \leqslant z_j + M \qquad (i,j) \in \widehat{A} : i \in \widehat{S} \tag{3.8}$$

$$y_o = Q \tag{3.9}$$

$$y_j + (q_{ij} + Q) x_{ij} \leqslant y_i + Q \qquad (i,j) \in \widehat{A} : i \in C_o \tag{3.10}$$

$$y_j + q_{ij} x_{ij} \leqslant Q \qquad (i,j) \in \widehat{A} : i \in \widehat{S} \tag{3.11}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in \widehat{A} \tag{3.12}$$

$$z_i, y_i \in \mathbb{Z}_+ \qquad i \in \widehat{V} \tag{3.13}$$

where M is a proper bigM value.

The objective function (3.1) asks for minimizing the total distance of the tour. Constraints (3.2) ensures that the optimal tour visits all customers exactly once and starts from vertex o. Constraints (3.3) stipulate that each dummy station $i \in \widehat{S}$ can be visited at most once. Constraints (3.4) are flow conservation constraints. Time windows are imposed by constraints (3.5) and (3.6). Constraints (3.7) and (3.8) set the times at which each vertex is visited; in particular, constraints (3.7) refer to arcs originating from a customer $i \in C$ or from o, and constraints (3.8) to arcs originating from a recharging station $i \in \widehat{S}$. Constraint (3.9) sets the initial battery level at vertex o. The battery level upon arriving at each vertex is set through constraints (3.10) and (3.11). Constraints (3.12) and (3.13) define the ranges of the three sets of variables.

Notice that the number of constraints and variables of formulation (3.1)–(3.13) depend on the cardinality of the set $\widehat{S}$; this affects the effectiveness of the formulation in solving the E-TSPTW. Moreover, to guarantee that an optimal solution is found, a proper number of copies of each station must be included. Unless a proper upper bound on the number of copies needed for each station is computed, $n+1$ copies are needed for each station.

3.3. A compact formulation for the partial-recharge policy

In order to model the partial-recharge policy, formulation (3.1)–(3.13) has to be slightly modified, as suggested by Bruglieri et al. (2015) for the E-VRPTW. In particular, in addition to the three sets of variables used, we introduce a variable $r_i \in \mathbb{Z}_+$ for each station $i \in \widehat{S}$ that represents the amount of battery recharged at vertex i if i is visited along the tour (it is undefined if i is not visited).

The resulting formulation has objective function (3.1) and constraints (3.2)–(3.7), (3.9), (3.10), (3.12), (3.13), but replaces constraints (3.8) and (3.11) with

$$z_i + (t_{ij} + M)x_{ij} + gr_i \leqslant z_j + M \qquad (i,j) \in \widehat{A} : i \in \widehat{S} \tag{3.14}$$

and

$$y_j + (q_{ij} + Q)x_{ij} \leqslant y_i + r_i + Q \qquad (i,j) \in \widehat{A} : i \in \widehat{S} \tag{3.15}$$

respectively, and also adds the following two sets of constraints

$$r_i + y_i \leqslant Q \qquad i \in \widehat{S} \tag{3.16}$$

and

$$r_i \in \mathbb{Z}_+ \qquad i \in \widehat{S} \tag{3.17}$$

Constraints (3.14) modifies (3.8) to take into account that the time spent to recharge at $i \in \widehat{S}$ depends on the quantity recharged (i.e., r_i) and is not simply $g(Q - y_i)$. Constraints (3.15) establish the relationship between y_j and y_i if a recharge is performed at $i \in \widehat{S}$. Constraints (3.16) state that the amount recharged at a vertex $i \in \widehat{S}$ plus the level of the battery when arriving at i cannot exceed Q. Integrality constraints on r-variables are imposed through constraints (3.17).

3.4. Improving the compact formulation for both recharge policies

Valid Inequalities – The linear relaxation of formulation (3.1)–(3.13) can be tightened by adding the following set of valid inequalities

$$x_{ij} + x_{ji} \leqslant 1, \quad i,j \in C : (i,j), (j,i) \in \widehat{A}, \tag{3.18}$$

which stipulate that, for each pair of customers $i,j \in C$, at most one of the two arcs (i,j) and (j,i) can be selected. Even though inequalities (3.18) are trivial, they increase the lower bound provided by the linear relaxation of (3.1)–(3.13) and decrease the total computing time when solving this formulation with a general-purpose MILP solver, so we will consider them in the computational experiments reported in Section 6.

It is trivial to observe that inequalities (3.18) are valid also for the partial-recharge policy.

Lifting Constraints (3.10) – We can also observe that inequalities (3.10) for each arc $(i,j) \in \widehat{A}$ such that $i,j \in C$ and $(j,i) \in \widehat{A}$ can be lifted as follows

$$y_j + (q_{ij} + Q)x_{ij} \leqslant y_i + Q - (Q - q_{ji})x_{ji}, \quad (i,j) \in \widehat{A} : i,j \in C, \ (j,i) \in \widehat{A}. \tag{3.19}$$

Let us consider a pair of customers $i,j \in C$ such that both (i,j) and (j,i) belong to $\widehat{A}$. There are three interesting cases to consider: (i) if $x_{ij} = 0$ and $x_{ji} = 0$, (ii) if $x_{ij} = 1$ and $x_{ji} = 0$, and (iii) if $x_{ij} = 0$ and $x_{ji} = 1$. In Case (i), inequalities (3.19) for arcs (i,j) and (j,i) simply become redundant. In case (ii), we have $y_j + q_{ij} \leqslant y_i$ and $y_i \leqslant y_j + q_{ij}$, so $y_i = y_j + q_{ij}$ (i.e., the battery left upon arriving at customer j is equal to the battery left upon arriving at customer i minus the battery consumption along arc (i,j)). Similarly, in case (iii), $y_j = y_i + q_{ji}$ (i.e., the battery left upon arriving at customer i is equal to the battery left upon arriving at customer j minus the battery consumption along arc (j,i)).

It is easy to observe that these lifted inequalities (3.19) are valid under both recharge policies.

4. An alternative formulation based on recharging paths

In this section, we present an alternative formulation of the E-TSPTW that contains a binary variable for each *recharging path* between each couple of vertices. A recharging path between two vertices i and j is a path that starts from i, visits one or more recharging stations, and ends at j. In principle, the resulting formulation has a number of variables that grows exponentially with the number of recharging stations, but we will describe some rules to eliminate a priori a significant number of variables.

For any arc $(i,j) \in A_C$, let us call $\mathcal{A}_{ij}$ the set of all recharging paths that start from vertex i, end at vertex j, and visit any subset of the vertices of the set S_o in any order. For a given path $p \in \mathcal{A}_{ij}$, where $p = (i = v_0, v_1, v_2, \ldots, v_k, v_{k+1} = j)$ and $v_1, v_2, \ldots, v_k \in S_o$, let d^p_{ij} (t^p_{ij}, resp.) be the total distance (travel time, resp.) of path p, given by the sum of the distances $d_{v_\alpha v_{\alpha+1}}$ (travel times $t_{v_\alpha v_{\alpha+1}}$, resp.) with $\alpha = 0, 1, \ldots, k$ of the arcs traversed by path p. The consumption q^p_{ij} of path $p \in \mathcal{A}_{ij}$ is defined as $q^p_{ij} = h \cdot d^p_{ij}$. For each path $p \in \mathcal{A}_{ij}$, we also indicate by f^p_{ij} (ℓ^p_{ij}, resp.) the consumption of the battery to traverse the first (last, resp.) arc of the path, namely, $f^p_{ij} = q_{iv_1}$ and $\ell^p_{ij} = q_{v_k j}$. Clearly, the cardinality of each set $\mathcal{A}_{ij}$ for a given arc $(i,j) \in A$ is exponential in the number of recharging stations.

The alternative formulation we propose uses the following sets of variables:

- $x_{ij} \in \{0,1\}$: binary variable equal to 1 if arc $(i,j) \in A_C$ is traversed (0 otherwise), meaning that no recharge takes place between the two visits to vertices i and j;
- $w^p_{ij} \in \{0,1\}$: binary variable equal to 1 if the recharging path $p \in \mathcal{A}_{ij}$ is used (0 otherwise);
- $z_i \in \mathbb{Z}_+$: time when the service at vertex $i \in C_{od}$ starts;
- $y_i \in \mathbb{Z}_+$: battery level upon arriving at vertex $i \in C_{od}$;
- $r_i \in \mathbb{Z}_+$: amount of battery recharged along recharging path $p \in \mathcal{A}_{ij}$ when traveling from vertex $i \in C_o$ to another vertex $j \in C_d$.

The proposed alternative formulation for the E-TSPTW with the full-recharge policy reads as follows

$$z^* = \min \sum_{(i,j) \in A_C} \left(d_{ij} x_{ij} + \sum_{p \in \mathcal{A}_{ij}} d^p_{ij} w^p_{ij} \right) \tag{4.1}$$

$$\text{s.t.} \sum_{(k,j) \in A_C} \left(x_{kj} + \sum_{p \in \mathcal{A}_{kj}} w^p_{kj} \right) = 1 \qquad k \in C_o \tag{4.2}$$

$$\sum_{(i,k) \in A_C} \left(x_{ik} + \sum_{p \in \mathcal{A}_{ik}} w^p_{ik} \right) = 1 \qquad k \in C_d \tag{4.3}$$

$$z_j \geqslant z_i + (t_{ij} + M) x_{ij} + \sum_{p \in \mathcal{A}_{ij}} (t^p_{ij} + M) w^p_{ij} + g r_i - M \qquad (i,j) \in A_C \tag{4.4}$$

$$e_k \leqslant z_k \leqslant l_k \qquad k \in C_d \tag{4.5}$$

$$y_i \geqslant y_j + (q_{ij} + M) x_{ij} + \sum_{p \in \mathcal{A}_{ij}} (q^p_{ij} + M) w^p_{ij} - r_i + (M - q_{ji}) x_{ji} - M \qquad (i,j) \in A_C \tag{4.6}$$

$$\sum_{(k,j) \in A_C} \left(q_{kj} x_{kj} + \sum_{p \in \mathcal{A}_{kj}} f^p_{kj} w^p_{kj} \right) \leqslant y_k \qquad k \in C \tag{4.7}$$

$$y_k \leqslant Q - \sum_{(i,k) \in A_C} \left(q_{ik} x_{ik} + \sum_{p \in \mathcal{A}_{ik}} \ell^p_{ik} w^p_{ik} \right) \qquad k \in C \tag{4.8}$$

$$r_k \leqslant M \sum_{(k,j) \in A_C} \sum_{p \in \mathcal{A}_{kj}} w^p_{kj} \qquad k \in C_o \tag{4.9}$$

$$r_i + y_i \geqslant \sum_{(i,j) \in A_C} \sum_{p \in \mathcal{A}_{ij}} (Q + q^p_{ij} - \ell^p_{ij}) w^p_{ij} \qquad i \in C_o \tag{4.10}$$

$$z_o = e_o \tag{4.11}$$

$$y_o = Q \tag{4.12}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in A_C \tag{4.13}$$

$$w^p_{ij} \in \{0,1\} \qquad (i,j) \in A_C,\ p \in \mathcal{A}_{ij} \tag{4.14}$$

$$z_i, y_i \in \mathbb{Z}_+ \qquad i \in C_{od} \tag{4.15}$$

$$r_i \in \mathbb{Z}_+ \qquad i \in C_o \tag{4.16}$$

The objective function (4.1) asks for minimizing the sum of the distances of the selected arcs and recharging paths. Constraints (4.2) stipulate that, given vertex $k \in C_o$, either an arc or a recharging path going toward another vertex j, such that $(k,j) \in A_C$, must be selected. Similarly, constraints (4.3) stipulate that any given vertex $k \in C_d$ must be reached through either a direct arc or a recharging path starting from vertex i such that $(i,k) \in A_C$.

Time windows constraints are modeled through constraints (4.4) and (4.5). In particular, constraints (4.4) set the visit times of vertices i and j by taking into account the travel times of the direct arc $(i,j) \in A_C$, the travel times of the recharging paths of the set $\mathcal{A}_{ij}$, and the time taken to recharge, i.e., gr_i. Constraints (4.5) define the earliest and latest arrival time at each vertex.

Constraints (4.6)–(4.10) model battery capacity constraints. Constraints (4.6) set the level of the battery when going from vertex i to vertex j (either directly or through a recharging path). In particular, it is worth noticing that the term $(M - q_{ji})x_{ji}$ is not necessary for the correctness of such inequalities, but allows to lift them and to break some symmetries. Indeed, consider an arc $(i,j) \in A_C$ and assume that no recharging path of the set $\mathcal{A}_{ij}$ is selected (thus, $w^p_{ij} = 0$ for all $p \in \mathcal{A}_{ij}$, and $r_i = 0$), there are two interesting cases to consider: (i) $x_{ij} = 1$ and $x_{ji} = 0$, and (ii) $x_{ij} = 0$ and $x_{ji} = 1$. In Case (i), inequality (4.6) for arc (i,j) reduces to $y_i \geqslant y_j + q_{ij}$, while inequality (4.6) for arc (j,i) reduces to $y_j \geqslant y_i - q_{ij}$, thus implying that $y_j = y_i - q_{ij}$. Similarly, in Case (ii), inequality (4.6) for arc (i,j) reduces to $y_i \geqslant y_j - q_{ji}$, while inequality (4.6) for arc (j,i) reduces to $y_j \geqslant y_i + q_{ji}$, thus implying that $y_i = y_j - q_{ji}$.

Constraints (4.7) impose on the level of the battery upon arriving at vertex $k \in C$ to be at least equal to the consumption of the corresponding selected outgoing arc. Constraints (4.8) impose on the level of the battery upon arriving at vertex $k \in C$ not to exceed the capacity of the battery minus the consumption of the corresponding selected incoming arc. Constraints (4.9) guarantee that no recharge takes place if none of the outgoing recharging arcs of vertex $k \in C$ is selected. Constraints (4.10) impose on each stop to fully recharge the battery.

The visit time and the initial level of the battery at vertex o are defined through constraints (4.11) and (4.12). Finally, constraints (4.13)–(4.16) define the range of the five sets of decision variables involved in the formulation.

It is worth noting the difference between the $\boldsymbol{r}$-variables in the compact formulation of Section 3 and in this alternative formulation. In both formulations, each variable r_i represents the amount of battery recharged, but, in the compact formulation, each variable r_i is associated to a visit to a recharging station (i.e., $i \in \widehat{S}$), whereas, in the alternative formulation, r_i is associated to a vertex $i \in C_o$. Indeed, the recharging stations do not explicitly appear in the alternative formulation but are hidden in the concept of recharging paths. Yet, because any feasible E-TSPTW solution is an elementary tour with regards to the set of customers C, each variable r_i does not need to be related to the endpoints j of the recharging paths of the sets $\mathcal{A}_{ij}$ for a given vertex $i \in C_o$ as at most one of the recharging paths of the set $\cup_{(i,j)\in A_C}\mathcal{A}_{ij}$ can be selected.

Partial Recharge Policy. In order to allow partial recharges, it is sufficient to remove constraints (4.10) from formulation (4.1)–(4.16).

4.1. Valid inequalities

The linear relaxation of formulation (4.1)–(4.16) can be tightened by adding the following set of valid inequalities

$$x_{ij} + \sum_{p\in\mathcal{A}_{ij}} w^p_{ij} + x_{ji} + \sum_{p\in\mathcal{A}_{ji}} w^p_{ji} \leqslant 1, \qquad (i,j) \in A_C, \tag{4.17}$$

which are a sort of generalization of inequalities (3.18). Even though inequalities (4.17) are trivial cuts, we observed that they contribute to decrease the computing times when solving (4.1)–(4.16) with a general-purpose MILP solver. Therefore, in the computational experiments reported in Section 6, they are added to (4.1)–(4.16).

4.2. Decreasing the number of variables

As previously indicated, one of the main issues of formulation (4.1)–(4.16) is that the number of decision variables w^p_{ij} increases exponentially with the number of recharging stations m. The following observations contribute to rule out some of the paths of the sets $\mathcal{A}_{ij}$ that cannot be part of any optimal E-TSPTW solution. Along with the conditions explicitly mentioned in each of the observations, we also assume that $d_{ij} = t_{ij}$ for each arc $(i,j) \in A$.

Observation 1. *Any optimal E-TSPTW solution cannot contain a recharging path that visits more than two recharging stations if the following conditions hold*

- $q_{ij} \leqslant Q$ *for each arc* $(i,j) \in A : i,j \in S$;
- *the triangle inequality holds for all triplets of arcs in the subgraph inducted by the recharging stations, that is,* $d_{ij} \leqslant d_{ik} + d_{kj}$ *for all* $i,j,k \in S : i \neq j \neq k$.

Observation 2. *Given an arc* $(i,j) \in A_C$ *and two recharging stations* $s_1, s_2 \in S$ *if the following inequalities hold*

$$d_{is_1} \leqslant d_{is_2} \quad \text{and} \quad d_{s_1j} \leqslant d_{s_2j}$$

then recharging path (i, s_2, j) *cannot be in any optimal solution.*

Observation 3. *Given an arc* $(i,j) \in A_C$ *and three recharging stations* $s_1, s_2, s_3 \in S$ *if the following inequalities hold*

$$d_{is_1} \leqslant d_{is_2} \qquad \text{and} \qquad d_{s_1j} \leqslant d_{s_3j}$$

and at least one of the two is strictly satisfied, then the recharging path (i, s_2, s_3, j) *cannot be used in any optimal solution because is dominated by the recharging path* (i, s_1, j).

Observation 4. *Given a vertex* $i \in C_o$ *and three recharging stations* $s_1, s_2, s_3 \in S$ *if the following inequalities hold*

$$d_{is_1} \leqslant d_{is_3} \qquad \text{and} \qquad d_{s_1s_2} \leqslant d_{s_3s_2}$$

and at least one of the two is strictly satisfied, then, in any optimal solution, there cannot be any recharging path (i, s_3, s_2, j) *for any vertex* $j \in C_d \setminus \{i\}$.

Observation 5. *Given a vertex* $j \in C_o$ *and three recharging stations* $s_1, s_2, s_3 \in S$ *if the following inequalities hold*

$$d_{s_1s_2} \leqslant d_{s_1s_3} \qquad \text{and} \qquad d_{s_2j} \leqslant d_{s_3j}$$

and at least one of the two is strictly satisfied, then, in any optimal solution, there cannot be any recharging path (i, s_1, s_3, j) *for any vertex* $i \in C_o \setminus \{j\}$.

A graphical representation of Observations 2–5 is given in Fig. 1.

5. A Three-Phase Heuristic algorithm

In this section, we describe a Three-Phase Heuristic algorithm for the E-TSPTW. The algorithm is based on the heuristic algorithms proposed by da Silva and Urrutia (2010) and Mladenović et al. (2012) for the TSPTW, which can be considered the state-of-the-art heuristic methods for the problem. The proposed algorithm generates a sequence of random Hamiltonian tours that may not satisfy time window and battery capacity constraints. For each of these tours, the following three phases are executed: (1) a *Variable Neighborhood Descent* (VND) algorithm is applied to reach time window feasibility; (2) a local search based on VND is applied to improve the cost of the tour; (3) an attempt to attain a feasible E-TSPTW solution is made by running a Dynamic Programming algorithm to add intermediate recharging stops and achieve battery capacity feasibility.

A step-by-step description of the proposed algorithm is provided in Algorithm 1. The algorithm has four parameters, i.e., `MaxIter`, `MinLevel`, `MaxLevel`, and Δ, that will be described in the following, and returns an E-TSPTW solution $\boldsymbol{y}^*$.

The algorithm initializes (see Line 2) the best-found TSPTW solution, $\boldsymbol{x}^*$, and the best-found E-TSPTW solution, $\boldsymbol{y}^*$, to nil and sets the iteration at which $\boldsymbol{x}^*$ was found, `iterBest`, equal to 0. Then, a random TSP tour is generated (Line 3); this tour $\boldsymbol{x}$ does not necessarily satisfy time window and battery level constraints.

The main loop of the algorithm (Lines 4–19) is iterated `MaxIter` times, and, at each iteration `iter`, Lines 5–19 are executed for `level` that goes from `MinLevel` up to `MaxLevel`, where `level` represents the number of random perturbations that are performed to escape for a local minimum.

In the perturbation phase (Lines 6–9), either the best-found TSPTW solution $\boldsymbol{x}^*$ or the incumbent tour $\boldsymbol{x}$ is perturbed. Solution $\boldsymbol{x}^*$ is selected for the perturbation if it has been found in one of the last Δ iterations (i.e., if $iterBest + \Delta \geqslant iter$), this allows to intensify the search around $\boldsymbol{x}^*$; otherwise, diversification is performed by perturbing the incumbent tour $\boldsymbol{x}$. The perturbation phase is taken from da Silva and Urrutia (2010) and consists of randomly relocating `level` customers forward by ignoring the feasibility of the resulting tour in terms of time windows and battery level and by considering precedence constraints

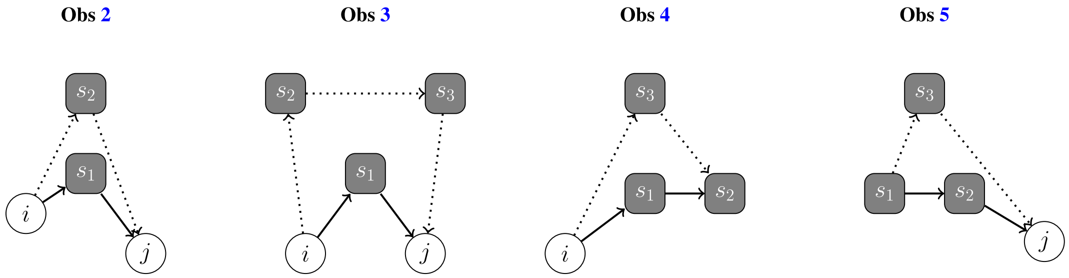

Fig. 1. Examples of dominated recharging paths according to Observations 2–5 – paths represented with straight lines dominate dotted paths.

only: customer j must precede customer i in any feasible TSPTW and E-TSPTW solution if $e_i + sp_{ij} > l_j$, where sp_{ij} is the length of the shortest path between i and j over graph G as defined in Section 3.

Once a tour $\boldsymbol{x}$ is determined, procedure *MakeTWFeasible(*$\boldsymbol{x}$*)* makes an attempt to derive a feasible TSPTW solution from $\boldsymbol{x}$ (Line 10). This procedure is Phase 1 of our algorithm and is described in details in Section 5.1. If a feasible TSPTW is found, procedure *ApplyLocalSearch(*$\boldsymbol{x}$*)* (Phase 2) tries to improve the overall cost of $\boldsymbol{x}$ by mean of a VND search (this procedure is described in Section 5.2). The best-found TSPTW solution is updated (see Lines 13–15) if the solution found by the local search is better than $\boldsymbol{x}^*$. Moreover, if the cost of $\boldsymbol{x}$ is not greater or equal to the cost of the best-found E-TSPTW solution $\boldsymbol{y}^*$, procedure *MakeETWFeasible(*$\boldsymbol{x}$*)* is run to derive an E-TSPTW solution from tour $\boldsymbol{x}$; indeed, $\boldsymbol{x}$ is a permutation of the n customers that satisfy all time window constraints but not necessarily the battery level constraints, so intermediate stop at recharging stations may be needed. Procedure *MakeETWFeasible(*$\boldsymbol{x}$*)*, which is Phase 3, is a dynamic programming algorithm that adds recharging stops in between tour $\boldsymbol{x}$ without changing the order of visit to the customers (a detailed description of the procedure is given in Section 5.3).

Algorithm 1. Three-Phase Heuristic algorithm.

1: **procedure** THREE-PHASEHEURISTIC(MaxIter,MinLevel,MaxLevel,Δ)
2: $\boldsymbol{x}^* \leftarrow nil; \boldsymbol{y}^* \leftarrow nil; iterBest \leftarrow 0$
3: $\boldsymbol{x} \leftarrow BuildRandomTour()$
4: **for** $iter = 1, \dots, MaxIter$ **do**
5: **for** $level = MinLevel, \dots, MaxLevel$ **do**
6: **if** $iterBest + \Delta \geqslant iter$ **then**
7: $\boldsymbol{x} \leftarrow Perturb(\boldsymbol{x}^*, level)$
8: **else**
9: $\boldsymbol{x} \leftarrow Perturb(\boldsymbol{x}, level)$
10: $\boldsymbol{x} \leftarrow MakeTWFeasible(\boldsymbol{x})$ ▹Phase 1
11: **if** $\boldsymbol{x}$ is TW-feasible **then**
12: $\boldsymbol{x} \leftarrow ApplyLocalSearch(\boldsymbol{x})$ ▹Phase 2
13: **if** $cost(\boldsymbol{x}) < cost(\boldsymbol{x}^*)$ **then**
14: $\boldsymbol{x}^* \leftarrow \boldsymbol{x}$
15: $iterBest \leftarrow iter$
16: **if**$cost(\boldsymbol{x}) < cost(\boldsymbol{y}^*)$ **then**
17: $\boldsymbol{y} \leftarrow MakeETWFeasible(\boldsymbol{x})$ ▹Phase 3
18: **if** $cost(\boldsymbol{y}) < cost(\boldsymbol{y}^*)$ **then**
19: $\boldsymbol{y}^* \leftarrow \boldsymbol{y}$
20: **return** $\boldsymbol{y}^*$

5.1. Phase 1 – Reach Time Window Feasibility

Phase 1 aims at modifying an input TSP tour $\boldsymbol{x}$ to make it satisfy time window constraints. The procedure uses a VND framework (see Hansen et al., 2008). The objective function used in this procedure is the sum of all positive differences between the visit time at each customer and the end of its window, that is, $\sum_{i \in C} \max\{0, z_i - l_i\}$, where z_i is the time visit of vertex i. A pseudo-code of this procedure is given in Algorithm 2.

This procedure tries to decrease the infeasibility of the incumbent tour $\boldsymbol{x}$ by performing a sequence of moves, namely, relocating customers backward, relocating customers forward, and swapping pairs of customers. The moves are repeated in this order as long as either some changes can be applied to decrease the objective function (i.e., the time window infeasibility) or $\boldsymbol{x}$ satisfies all time window constraints.

Algorithm 2. Phase 1: Reach Time Window Feasibility.

procedure MAKETWFEASIBLE($\boldsymbol{x}$)
while $\boldsymbol{x}$ is TW-Infeasible **do**
$\boldsymbol{x}' \leftarrow \boldsymbol{x}$
$\boldsymbol{x} \leftarrow RelocateBkw(\boldsymbol{x})$
$\boldsymbol{x} \leftarrow RelocateFrw(\boldsymbol{x})$
$\boldsymbol{x} \leftarrow Swap(\boldsymbol{x})$
if $\boldsymbol{x} = \boldsymbol{x}'$ **then**
break
return $\boldsymbol{x}$

5.2. Phase 2 – Local Search

In this phase, a local search procedure based on VND to improve the traveled distance of an input tour $\boldsymbol{x}$ that satisfies time window constraints is performed. As indicated in the pseudo-code of Algorithm 3, four neighborhood are iteratively explored, namely, relocate backward, relocate forward, two-opt, and swap, and for each of them the best improving move that maintains time window feasibility is performed. The procedure ends as soon as no improving moves can be found in any of the neighborhoods. For implementation issues, the reader is referred to da Silva and Urrutia (2010) and Mladenović et al. (2012).

Algorithm 3. Phase 2: Local Search.

procedure APPLYLOCALSEARCH($\boldsymbol{x}$)
  repeat
    $\boldsymbol{x}' \leftarrow \boldsymbol{x}$
    $\boldsymbol{x} \leftarrow RelocateBkw(\boldsymbol{x})$
    $\boldsymbol{x} \leftarrow RelocateFrw(\boldsymbol{x})$
    $\boldsymbol{x} \leftarrow TwoOpt(\boldsymbol{x})$
    $\boldsymbol{x} \leftarrow Swap(\boldsymbol{x})$
   until $\boldsymbol{x}' \neq \boldsymbol{x}$
   return $\boldsymbol{x}$

5.3. Phase 3 – Reach Battery Capacity Feasibility

Procedure *MakeETWFeasible* takes a feasible TSPTW solution $\boldsymbol{x}$ as input and returns, as output, a least-cost E-TSPTW solution $\boldsymbol{y}$ (if such a solution exists) where the customers are visited in the same order of solution $\boldsymbol{x}$. In other words, the procedure tries to insert, in an optimal way, the stops at the recharging stations in between the different customers while maintaining the level of the battery between 0 and Q at any point along the tour.

Let $\boldsymbol{x} = (o = i_0, i_1, i_2, \ldots, i_n, i_{n+1} = d)$ be the TSPTW solution given in input, where the first and the last visited vertices are o and d, respectively. The procedure applies a forward mono-directional labeling algorithm, where a partial path from o to a vertex i_k (with $0 \leqslant k \leqslant n+1$) is represented by a label $L = (k, t, q, c)$, whose components are:

- k: index of the last visited vertex i_k;
- t: arrival time at vertex i_k;
- q: level of the battery upon arriving at vertex i_k;
- c: cost of the partial path.

Each label $L = (k, t, q, c)$ is feasible if and only if $e_{i_k} \leqslant t \leqslant l_{i_k}$ and $0 \leqslant q \leqslant Q$.

The labels are computed by starting from the initial label $L = (0, e_o, Q, 0)$ and by extending each label $L = (k, t, q, c)$ as follows:

1. through arc (i_k, i_{k+1}), by generating label $\widehat{L} = (\hat{k}, \hat{t}, \hat{q}, \hat{c})$ defined as
   - $\hat{k} = k + 1$;
   - $\hat{t} = \max\{e_{i_{k+1}}, t + t_{i_k i_{k+1}}\}$;
   - $\hat{q} = q - q_{i_k i_{k+1}}$;
   - $\hat{c} = c + d_{i_k i_{k+1}}$;
2. through each recharging path $p \in \mathcal{A}_{i_k i_{k+1}}$ such that $q \geqslant f^p_{i_k i_{k+1}}$, by generating a label $\widehat{L} = (\hat{k}, \hat{t}, \hat{q}, \hat{c})$ for each possible recharging quantity r (such that $\max\{0, q^p_{i_k i_{k+1}} - q\} \leqslant r \leqslant Q - q + q^p_{i_k i_{k+1}} - \ell^p_{i_k i_{k+1}}$) defined as
   - $\hat{k} = k + 1$;
   - $\hat{t} = \max\{e_{i_{k+1}}, t + t^p_{i_k i_{k+1}} + \lceil gr \rceil\}$;
   - $\hat{q} = q - q^p_{i_k i_{k+1}} + r$;
   - $\hat{c} = c + d^p_{i_k i_{k+1}}$.

The number of labels to generate can significantly be decreased by applying the following dominance rule.

Observation 6. *Let $\widehat{L} = (\hat{k}, \hat{t}, \hat{q}, \hat{c})$ and $\widetilde{L} = (\tilde{k}, \tilde{t}, \tilde{q}, \tilde{c})$ be two labels corresponding to two paths ending at the same vertex (i.e., $\hat{k} = \tilde{k}$). If $\hat{t} \leqslant \tilde{t}, \hat{q} \geqslant \tilde{q}, \hat{c} \leqslant \tilde{c}$, and at least one of the three inequalities is strictly satisfied, then label $\widetilde{L}$ is dominated and can be discarded.*

The previous propagation rules are valid when dealing with the partial recharge policy. In order to apply the full recharge policy, it is enough to extend each label through a recharging path only once with $r = Q - q + q^p_{i_k i_{k+1}} - \ell^p_{i_k i_{k+1}}$.

6. Computational results

This section is devoted to the computational results. First, we describe the test instances we generated. Then, we report the computational performance of the compact formulation (3.1)–(3.13) (hereafter called CF) and the alternative formulation (4.1)–(4.16) (hereafter called PBF) on the test instances for both the full recharge and the partial recharge policy. Finally, we illustrate the results of the heuristic algorithm described in Section 5. All results reported in this section were obtained by using an Intel Core i7-4800MQ @2.70 GHz equipped with 8 GB of RAM, and all computing times are reported in seconds.

6.1. Description of the test instances

We generated two sets of test instances, for a total of 100 instances, by starting from TSPTW benchmark instances available in the literature. The first set of instances contains 50 small instances (with 20 customers) derived from the 20-customer instances proposed in Gendreau et al. (1998) – hereafter, we refer to this instance set as `G`. The second set of instances consists of 50 large instances (30 instances with 150 customers and 20 instances with 200 customers) generated from the TSPTW instances proposed in Ohlmann and Thomas (2007) – this set is referred to as `OT` in the following. All these instances feature distances, d_{ij}, equal to the travel times t_{ij}.

On all instances, the following parameters were fixed: the consumption ratio $h = 1$, and the recharging ratio $g = 0.25$. The number of recharging stations is either five or ten. In particular, from each TSPTW instance, we derived an instance with five stations and another instance with ten stations. In the instance with five stations, one of the stations is the main depot, whereas the other four stations were evenly located in the four quadrants of the box defined by the customer locations, namely, the station locations were defined as follows: given the Cartesian coordinates (α_i, β_i) of each customer $i \in C$, the minimum and maximum values of α and β were computed ($\underline{\alpha} = \min_{i\in C}\{\alpha_i\}, \overline{\alpha} = \max_{i\in C}\{\alpha_i\}, \underline{\beta} = \min_{i\in C}\{\beta_i\}, \overline{\beta} = \max_{i\in C}\{\beta_i\}$), then the locations of the four stations were set equal to $\left(\lfloor\underline{\alpha} + \frac{1}{4}\Delta_\alpha\rfloor, \left\lfloor\underline{\beta} + \frac{1}{4}\Delta_\beta\right\rfloor\right), \left(\lfloor\underline{\alpha} + \frac{1}{4}\Delta_\alpha\rfloor, \left\lfloor\underline{\beta} + \frac{3}{4}\Delta_\beta\right\rfloor\right), \left(\lfloor\underline{\alpha} + \frac{3}{4}\Delta_\alpha\rfloor, \left\lfloor\underline{\beta} + \frac{1}{4}\Delta_\beta\right\rfloor\right)$ and $\left(\lfloor\underline{\alpha} + \frac{3}{4}\Delta_\alpha\rfloor, \left\lfloor\underline{\beta} + \frac{3}{4}\Delta_\beta\right\rfloor\right)$, where $\Delta_\alpha = \overline{\alpha} - \underline{\alpha}$ and $\Delta_\beta = \overline{\beta} - \underline{\beta}$. The instance with ten stations contains the same set of stations of the previous instance plus five additional stations randomly located in the square $(\underline{\alpha}, \overline{\alpha}) \times (\underline{\beta}, \overline{\beta})$, so as to guarantee that any feasible solution of the instance with five stations is a valid upper bound to the instance with ten stations. The distances/travel times between the recharging stations and the customers (as well as between recharging stations) are computed as Euclidean distances rounded to the nearest integer.

The capacity of the battery, Q, was set as $Q = \lceil \frac{z^*_{\mathrm{TW}}}{es+1} \rceil$, where z^*_{TW} is the cost of the best-known solution for the corresponding TSPTW instances that can be found in the literature (i.e., from Ohlmann and Thomas (2007), da Silva and Urrutia (2010) or López-Ibáñez and Blum (2010)), and *es* is a parameter that corresponds to the minimum number of expected stops at recharging stations in the instance; *es* was set equal to 2 for the `G` instances and equal to 5 for the `OT` instances. This setting of parameter *es* was decided to have battery capacities roughly in the range 80–120, which, by assuming that distances are given in kilometers, represent the driving range in many real-life applications of electric vehicles.

In order to guarantee that a feasible solution exists for both recharging policies (full and partial) in all instances, we modified the original time windows as follows. For the instance with five stations, we computed a value δ representing the increase of l_i for all vertices $i \in C_{od}$ with respect to the original TSPTW instance such that the best-known solution of the TSPTW instance can be transformed into a feasible (not necessarily optimal) solution of the E-TSPTW under the full-recharge policy; then, the same time windows were used in the other instance with ten stations. Value δ was computed by using the following MILP model.

Let $(o = v_0, v_1, v_2, \ldots, v_n, v_{n+1} = d)$ be the best-known TSPTW solution of cost z^*_{TW}. For each vertex v_i $(i = 0, 1, \ldots, n)$, let $s_i \in S$ be the recharging station having the cheapest insertion cost between v_i and v_{i+1} (i.e., $s_i = \arg\min_{k\in S}\{t_{v_i k} + t_{k v_{i+1}} - t_{v_i v_{i+1}}\}$).

The following seven sets of variables are used:

- $x_i \in \{0, 1\}$: equals 1 if vertex v_i $(i = 0, 1, \ldots, n)$ is visited right before vertex v_{i+1} without any recharge in between (0 otherwise);
- $w_i \in \{0, 1\}$: equals 1 if vertex v_i $(i = 0, 1, \ldots, n)$ is visited right before vertex v_{i+1} with a recharge in between (0 otherwise);
- $z_i \in \mathbb{Z}_+$: time at which service starts at vertex $v_i, i = 0, 1, \ldots, n+1$;
- $y_i \in \mathbb{Z}_+$: battery level upon arriving at vertex v_i $(i = 0, 1, \ldots, n+1)$;
- $u_i \in \mathbb{Z}_+$: time spent to recharge when traveling from v_i to v_{i+1} $(i = 0, 1, \ldots, n)$;

- $r_i \in \mathbb{Z}$: amount recharged when traveling from v_i to v_{i+1} through recharging station s_i $(i = 0, 1, \ldots, n)$;
- $\delta \in \mathbb{Z}_+$: necessary increase of time windows to guarantee that a feasible E-TSPTW solution can be obtained from the input TSPTW solution.

The MILP model to compute δ reads as follows

$$\min M\delta + \sum_{i=0}^{n+1} z_i + \sum_{i=0}^{n} u_i \tag{6.1}$$

$$\text{s.t. } x_i + w_i = 1 \qquad i = 0, 1, \ldots, n \tag{6.2}$$

$$e_{v_i} \leqslant z_i \leqslant l_{v_i} + \delta \qquad i = 0, 1, \ldots, n+1 \tag{6.3}$$

$$gr_i - Mx_i \leqslant u_i \leqslant \lceil gQ \rceil w_i \qquad i = 0, 1, \ldots, n \tag{6.4}$$

$$r_i = y_{i+1} - y_i + q_{v_i v_{s_i}} + q_{v_{s_i} v_{i+1}} \qquad i = 0, 1, \ldots, n \tag{6.5}$$

$$z_{i+1} \geqslant z_i + t_{v_i v_{i+1}} x_i + (t_{v_i s_i} + t_{s_i v_{i+1}}) w_i + u_i \qquad i = 0, 1, \ldots, n \tag{6.6}$$

$$y_i \geqslant y_{i+1} + q_{v_i v_{i+1}} - Mw_i \qquad i = 0, 1, \ldots, n \tag{6.7}$$

$$y_i \geqslant q_{v_i s_i} w_i \qquad i = 1, \ldots, n \tag{6.8}$$

$$Qw_{i-1} - q_{s_{i-1} v_i} \leqslant y_i \leqslant Q - q_{s_{i-1} v_i} w_{i-1} \qquad i = 1, \ldots, n+1 \tag{6.9}$$

$$z_0 = e_0 \tag{6.10}$$

$$y_0 = Q \tag{6.11}$$

$$x_i, w_i \in \mathbb{B} \qquad i = 0, 1, \ldots, n \tag{6.12}$$

$$z_i, y_i \in \mathbb{Z}_+ \qquad i = 0, 1, \ldots, n+1 \tag{6.13}$$

$$u_i \in \mathbb{Z}_+ \qquad i = 0, 1, \ldots, n \tag{6.14}$$

$$r_i \in \mathbb{Z} \qquad i = 0, 1, \ldots, n \tag{6.15}$$

$$\delta \in \mathbb{Z}_+ \tag{6.16}$$

The objective function (6.1) aims at minimizing the increase (δ) of the end of the time windows, l_i, in order to obtain a feasible solution, plus the arrival and the recharge times at the different vertices. Constraints (6.2) impose on each vertex v_i $(i = 0, 1, \ldots, n)$ to either go directly to vertex v_{i+1} (if $x_i = 1$) or to pass through recharging station s_i (if $w_i = 1$). For each vertex v_i, the arrival time cannot exceed the end of the time window plus δ (see constraints (6.3)) and cannot be less than the beginning of the time window e_{v_i}. The relationship between variables w_i, r_i and u_i is stipulated through constraints (6.4), in particular, for vertex v_i, the time to recharge u_i can be strictly positive if and only if vertex v_{i+1} is reached through recharging station s_i. Constraints (6.5) set the amount recharged at each vertex $v_i, i = 0, 1, \ldots, n$. Constraints (6.6) set the arrival time at the different vertices taking into account the travel times and the recharging times. The battery level after visiting each vertex i is set through constraints (6.7). Constraints (6.8) guarantee that the battery level is such that the recharging station can be reached if a recharge is performed. Constraints (6.9) guarantee that the battery level does not exceed the capacity of the battery minus the consumption from a recharging station to the next vertex and that only full recharges are performed. Constraints (6.10) and (6.11) set the initial level of the battery and the visit time of vertex o. The range of the variables is set through constraints (6.12)–(6.16).

Table 1 reports the features of the benchmark instances. The left part of the table refers to the `G` instances, while the right part of the table to the `OT` instances. The following data is indicated for each instance: instance name of the original TSPTW instance (`Inst`) in the format `nXXwYYY.Z`, where `XX` is the number of customers, `YYY` is the width of the time windows in the original instance, and `Z` is the instance number; cost (z^*_{TW}) of the best-known TSPTW solution from the literature; battery capacity `Q`; increase (δ) of the time windows with respect to the original TSPTW instance; number of stops (`ns`) in the optimal solution of problem (6.1)–(6.16); valid upper bound (`UB`) derived by computing the distance of the optimal solution of (6.1)–(6.16). As previously explained, for each of these TSPTW instance, two E-TSPTW instances were derived: the name format used will be `nXXwYYYsW.Z`, where `XX`, `YYY`, and `Z` are derived from the original TSPTW and `W` represents the number of recharging stations (either five or ten).

6.2. Computational results of CF and PBF on `G` Instances

In this section, we summarize the performance of formulations CF and PBF when solving the `G` instances through the general-purpose MILP solver Cplex version 12.6.0.0 with a time limit of two hours. For the sake of conciseness, full detailed results are reported in the appendix (see Tables 11–14).

With regards to CF, when considering the full-recharge policy, we used formulation (3.1)–(3.13) plus valid inequalities (3.18) and the lifted version (3.19) of constraints (3.10); whereas, under the partial-recharge policy, constraints (3.8) and (3.11) were removed and constraints (3.14)–(3.17) were added.

Table 1
Features of the benchmark instances.

G						OT					
Inst	z^*_{TW}	Q	δ	ns	UB	Inst	z^*_{TW}	Q	δ	ns	UB
n20w120.1	267	92	5	3	271	n150w120.1	734	124	0	8	789
n20w120.2	218	76	0	3	244	n150w120.2	677	116	14	10	759
n20w120.3	303	104	1	3	331	n150w120.3	747	128	13	8	787
n20w120.4	300	100	0	3	318	n150w120.4	763	128	28	9	828
n20w120.5	240	80	0	4	265	n150w120.5	689	116	15	8	737
n20w140.1	176	60	0	4	192	n150w140.1	762	128	13	11	830
n20w140.2	272	92	0	3	279	n150w140.2	755	128	18	9	840
n20w140.3	236	80	0	3	251	n150w140.3	613	104	13	8	702
n20w140.4	255	88	0	4	279	n150w140.4	676	116	0	7	726
n20w140.5	225	76	0	4	231	n150w140.5	663	112	5	10	719
n20w160.1	241	84	0	6	275	n150w160.1	706	120	0	9	788
n20w160.2	201	68	0	4	224	n150w160.2	711	120	48	9	743
n20w160.3	201	68	0	3	212	n150w160.3	608	104	8	10	700
n20w160.4	203	68	0	5	232	n150w160.4	672	112	20	8	719
n20w160.5	245	84	0	4	271	n150w160.5	658	112	0	10	713
n20w180.1	253	88	0	4	276	n200w120.1	799	136	0	8	871
n20w180.2	265	92	1	4	302	n200w120.2	721	124	0	8	801
n20w180.3	271	92	9	4	300	n200w120.3	880	148	0	7	928
n20w180.4	201	68	13	5	220	n200w120.4	777	132	0	7	818
n20w180.5	193	68	0	4	232	n200w120.5	841	144	0	7	892
n20w200.1	233	80	0	4	244	n200w140.1	834	140	10	10	890
n20w200.2	203	68	0	5	238	n200w140.2	760	128	5	9	834
n20w200.3	249	84	0	4	280	n200w140.3	758	128	0	8	790
n20w200.4	293	100	0	4	309	n200w140.4	816	136	15	8	890
n20w200.5	227	76	14	4	252	n200w140.5	822	140	2	9	867
Avg			1.7	3.9					9.1	8.6	

With regards to PBF, when applying the full-recharge policy, we used formulation (4.1)–(4.16) plus valid inequalities (4.17); whereas, for the partial-recharge policy, constraints (4.10) were removed. All instances satisfy the conditions of Observation 1, so it was possible to generate all variables of PBF by pure enumeration.

As previously indicated, in order to guarantee that the optimal solution found by CF is such, $n+1$ copies of each recharging station have to be done. We tested CF on all the G by having $n+1$ copies, and none of the instances was closed within the time limit of two hours. Therefore, by knowing from the optimal solutions found by PBF that no station is visited more than three times, we tested CF with just three copies per station on all instances and report the corresponding results. This means that the comparison between CF and PBF is biased toward CF.

A summary of the results is illustrated in Table 2. For each group of five instances having the same time window width, the following columns are reported: the number of recharging stations $|S|$; the recharging policy applied (i.e., Policy, where F stands for *full*, and P for *partial*); and, for both formulations, the average percentage lower bound of the linear relaxation (%LP), the average percentage of the final lower bound (%bLB), the average percentage of the best upper bound found (%bUB), the number of instances (out of five) where a feasible solution was found (Feas), the number of instances (out of five) solved to optimality (Opt), and the average computing time (Time). All percentage values of columns %LP, %bLB, and %bUB are computed over the best-known upper bounds.

Table 2 shows that PBF outperforms CF by solving all but one instance to optimality, while CF could close only 17 of the 100 instances. This is probably due to the better quality of the linear relaxation of PBF (78.7% vs 70.9%). Moreover, PBF was able to find a feasible solution on all instances, whereas CF could find a feasible solution on 57 instances, only.

It is interesting to notice that, by comparing the same groups of instances solved with different recharging policies, the computing time when applying the partial-recharge policy is generally lower, even though the quality of the lower bound provided by the linear relaxation is just slightly better. We believe that this happens because Cplex strongly benefits from finding good upper bounds very early in the search tree, and, under the partial-recharge policy, finding feasible solutions was way easier.

It is also worth noting that the increase of recharging stations made instances harder when solved under the full-recharge policy (indeed, the only open instance has ten stations and applies the full-recharge policy), whereas the increase of computing times observed under the partial-recharge policy is less significant.

Further statistics about PBF are reported in Table 3. In particular, for all the instances with the same number of stations (either five or ten), we report the average number of variables (Vars) in the formulation, and the average number of variables corresponding to recharging paths eliminated by each of the four observations (Observations 2–5). Notice that the number of variables of PBF does not depend on the recharging policy applied.

Table 2
Summary of the computational results of CF and PBF on `G` instances.

Group	$\|S\|$	Policy	CF						PBF					
			%LP	%bLB	%bUB	Feas	Opt	Time	%LP	%bLB	%bUB	Feas	Opt	Time
n20w120	5	F	72.6	89.0	100.0	2	2	5173.0	86.5	100.0	100.0	5	5	277.5
n20w140	5	F	71.3	84.5	100.0	1	1	6234.1	75.9	100.0	100.0	5	5	319.8
n20w160	5	F	71.6	90.5	100.5	4	1	6146.2	76.7	100.0	100.0	5	5	28.2
n20w180	5	F	73.5	93.3	100.0	3	2	5592.2	76.7	100.0	100.0	5	5	686.7
n20w200	5	F	73.7	87.4	100.0	3	1	6357.3	75.3	100.0	100.0	5	5	176.7
n20w120	5	P	73.4	93.0	100.0	3	2	4557.8	87.5	100.0	100.0	5	5	60.1
n20w140	5	P	71.4	91.4	100.0	3	3	4605.7	76.0	100.0	100.0	5	5	68.2
n20w160	5	P	71.6	91.8	100.2	5	2	5399.2	76.7	100.0	100.0	5	5	13.9
n20w180	5	P	74.0	95.0	102.3	4	0	7200.0	77.3	100.0	100.0	5	5	73.8
n20w200	5	P	73.7	88.5	100.0	4	1	6110.3	75.3	100.0	100.0	5	5	129.8
n20w120	10	F	66.2	81.1		0	0	7200.0	86.8	100.0	100.0	5	5	450.1
n20w140	10	F	68.7	81.0	100.0	1	0	7200.0	76.7	100.0	100.0	5	5	538.6
n20w160	10	F	69.6	82.2	100.0	1	0	7200.0	77.9	100.0	100.0	5	5	68.8
n20w180	10	F	70.7	85.7	100.0	1	1	6886.0	76.8	100.0	100.0	5	5	362.3
n20w200	10	F	69.9	83.4	100.2	4	0	7200.0	76.7	97.8	100.0	5	4	1521.0
n20w120	10	P	66.4	82.3	101.8	2	0	7200.0	87.0	100.0	100.0	5	5	76.1
n20w140	10	P	68.8	80.9	100.1	3	1	6243.1	76.8	100.0	100.0	5	5	115.1
n20w160	10	P	69.6	84.0	100.9	4	0	7200.0	77.9	100.0	100.0	5	5	22.6
n20w180	10	P	70.9	86.8	101.1	5	0	7200.0	77.0	100.0	100.0	5	5	86.1
n20w200	10	P	69.9	82.7	100.4	4	0	7200.0	76.7	100.0	100.0	5	5	122.7
Avg			70.9	86.7		57	17	6853.7	78.7	99.9	100.0	100	99	259.9

Table 3
Variables of PBF eliminated with observations 2–5 when solving `G` instances.

$\|S\|$	Vars	Observation 2	Observation 3	Observation 4	Observation 5
5	1852	1227	5351	279	197
10	3092	3059	25138	1756	982

6.3. Computational results of the Three-Phase Heuristic

In this section, we report the computational results achieved by the Three-Phase Heuristic (hereafter called 3P-Heu) described in Section 5 on the 100 benchmark instances under the two recharging policies.

On all instances, the following parameter setting was used `MaxIter` = 500, `MinLevel` = $\frac{n}{12}$, `MaxLevel` = $\frac{n}{2}$, $\Delta = 20$, and 10 runs where performed on each instance. To define such a setting, we performed some preliminary experiments on five `OT` instances and defined the four parameters in a sort of layer approach. First, we fixed `MaxIter` and `Delta`, and we tested 3P-Heu for all possible combinations of `MinLevel` $= \frac{n}{12}, \frac{n}{10}, \frac{n}{8}, \frac{n}{6}$ and `MaxLevel` $= \frac{n}{5}, \frac{n}{4}, \frac{n}{3}, \frac{n}{2}$. Even though there were not significant differences between different parameter settings, we observed that the algorithm benefits from having a wide range of `MinLevel`–`MaxLevel`. So we fixed `MinLevel` $= \frac{n}{12}$ and `MaxLevel` $= \frac{n}{2}$. Similarly, we determined parameters Δ and `MaxIter`.

The computational results achieved by 3P-Heu are reported in Tables 4–7. In particular, Tables 4,5 concern `G` instances with five and ten recharging stations, respectively, and Tables 6,7 concern `OT` instances with five and ten recharging stations, respectively.

Tables 4–7 indicate the following data: instance name (`Inst`); the optimal E-TSPTW solution cost (z^*_{ETW}) when available; the cost of the best solution found (`Best`) over ten runs along with its percentage deviation from z^*_{ETW}, computed as $\frac{100 \cdot \mathtt{Best}}{z^*_{\mathrm{ETW}}}$; the average (`Avg`) of the solution cost of the best solution found in each run and the corresponding percentage deviation from z^*_{ETW}, computed as $\frac{100 \cdot \mathtt{Avg}}{z^*_{\mathrm{ETW}}}$; the standard deviation of the best solution found at each iteration (σ); the average computing time (`Time`) over the ten runs; and the number of stops (`ns`) in the best found solution.

Table 4 shows that 3P-Heu could find the optimal solution in all ten runs on 23 instances under both recharging policies. On the remaining two instances (i.e., `n20w200s5.3` and `n20w200s5.4`), on all ten runs the best solution found cost a unit more than the optimal solution. As expected, the number of stops to recharge increases when applying the partial recharge policy and the solution cost slightly decreases. The computing time of 3P-Heu is negligible.

The results achieved by 3P-Heu on the `G` instances with ten recharging stations are reported in Table 5. Even on these instances, 3P-Heu performed well and was able to find an optimal solution on all instances on all ten runs, even though we do not know if `295` is the cost of the optimal solution of instance `n20w200s10.4` when the full-recharge policy is considered. We notice that by allowing partial recharges (instead of imposing full recharges) only a small decrease in the average optimal solution cost is achieved. The computing time of 3P-Heu remains quite negligible.

Table 4
Computational results of 3P-Heu on G instances with five stations under both recharging policies.

Inst	Full recharge policy								Partial recharge policy							
	z^*_{ETW}	Best	%	Avg	%	σ	Time	ns	z^*_{ETW}	Best	%	Avg	%	σ	Time	ns
n20w120s5.1	271	271	0.0	271.0	0.0	0.0	0.05	3	271	271	0.0	271.0	0.0	0.0	0.05	4
n20w120s5.2	233	233	0.0	233.0	0.0	0.0	0.11	3	225	225	0.0	225.0	0.0	0.0	0.09	3
n20w120s5.3	317	317	0.0	317.0	0.0	0.0	0.07	3	311	311	0.0	311.0	0.0	0.0	0.06	4
n20w120s5.4	314	314	0.0	314.0	0.0	0.0	0.07	4	312	312	0.0	312.0	0.0	0.0	0.07	4
n20w120s5.5	249	249	0.0	249.0	0.0	0.0	0.06	3	249	249	0.0	249.0	0.0	0.0	0.06	3
n20w140s5.1	181	181	0.0	181.0	0.0	0.0	0.08	4	180	180	0.0	180.0	0.0	0.0	0.08	4
n20w140s5.2	279	279	0.0	279.0	0.0	0.0	0.06	4	278	278	0.0	278.0	0.0	0.0	0.07	4
n20w140s5.3	238	238	0.0	238.0	0.0	0.0	0.06	4	238	238	0.0	238.0	0.0	0.0	0.06	4
n20w140s5.4	265	265	0.0	265.0	0.0	0.0	0.10	3	265	265	0.0	265.0	0.0	0.0	0.10	4
n20w140s5.5	229	229	0.0	229.0	0.0	0.0	0.06	5	229	229	0.0	229.0	0.0	0.0	0.06	5
n20w160s5.1	246	246	0.0	246.0	0.0	0.0	0.09	3	246	246	0.0	246.0	0.0	0.0	0.09	3
n20w160s5.2	219	219	0.0	219.0	0.0	0.0	0.07	3	219	219	0.0	219.0	0.0	0.0	0.07	4
n20w160s5.3	210	210	0.0	210.0	0.0	0.0	0.07	3	210	210	0.0	210.0	0.0	0.0	0.07	3
n20w160s5.4	208	208	0.0	208.0	0.0	0.0	0.09	4	208	208	0.0	208.0	0.0	0.0	0.09	4
n20w160s5.5	253	253	0.0	253.0	0.0	0.0	0.08	4	253	253	0.0	253.0	0.0	0.0	0.08	4
n20w180s5.1	262	262	0.0	262.0	0.0	0.0	0.07	4	262	262	0.0	262.0	0.0	0.0	0.07	5
n20w180s5.2	273	273	0.0	273.0	0.0	0.0	0.08	3	273	273	0.0	273.0	0.0	0.0	0.08	3
n20w180s5.3	282	282	0.0	282.0	0.0	0.0	0.12	4	271	271	0.0	271.0	0.0	0.0	0.10	5
n20w180s5.4	206	206	0.0	206.0	0.0	0.0	0.10	4	206	206	0.0	206.0	0.0	0.0	0.10	4
n20w180s5.5	201	201	0.0	201.0	0.0	0.0	0.10	3	201	201	0.0	201.0	0.0	0.0	0.10	3
n20w200s5.1	241	241	0.0	241.0	0.0	0.0	0.09	3	241	241	0.0	241.0	0.0	0.0	0.09	3
n20w200s5.2	221	221	0.0	221.0	0.0	0.0	0.09	3	221	221	0.0	221.0	0.0	0.0	0.09	3
n20w200s5.3	254	255	0.4	255.0	0.4	0.0	0.08	4	254	255	0.4	255.0	0.4	0.0	0.09	4
n20w200s5.4	295	296	0.3	296.0	0.3	0.0	0.10	3	295	296	0.3	296.0	0.3	0.0	0.09	3
n20w200s5.5	240	240	0.0	240.0	0.0	0.0	0.12	4	240	240	0.0	240.0	0.0	0.0	0.12	4
Avg	247.5	247.6	0.03	247.6	0.03	0.00	0.08	3.5	246.3	246.4	0.03	246.4	0.03	0.00	0.08	3.8

Table 5
Computational results of 3P-Heu on G instances with ten stations under both recharging policies.

Inst	Full recharge policy								Partial recharge policy							
	z^*_{ETW}	Best	%	Avg	%	σ	Time	ns	z^*_{ETW}	Best	%	Avg	%	σ	Time	ns
n20w120s10.1	270	270	0.0	270.0	0.0	0.0	0.05	3	270	270	0.0	270.0	0.0	0.0	0.05	4
n20w120s10.2	222	222	0.0	222.0	0.0	0.0	0.09	3	220	220	0.0	220.0	0.0	0.0	0.09	3
n20w120s10.3	312	312	0.0	312.0	0.0	0.0	0.06	3	311	311	0.0	311.0	0.0	0.0	0.06	5
n20w120s10.4	308	308	0.0	308.0	0.0	0.0	0.07	3	307	307	0.0	307.0	0.0	0.0	0.07	4
n20w120s10.5	243	243	0.0	243.0	0.0	0.0	0.06	4	243	243	0.0	243.0	0.0	0.0	0.06	4
n20w140s10.1	179	179	0.0	179.0	0.0	0.0	0.08	5	178	178	0.0	178.0	0.0	0.0	0.08	4
n20w140s10.2	277	277	0.0	277.0	0.0	0.0	0.06	4	277	277	0.0	277.0	0.0	0.0	0.06	5
n20w140s10.3	237	237	0.0	237.0	0.0	0.0	0.07	4	237	237	0.0	237.0	0.0	0.0	0.07	5
n20w140s10.4	260	260	0.0	260.0	0.0	0.0	0.09	3	260	260	0.0	260.0	0.0	0.0	0.09	5
n20w140s10.5	225	225	0.0	225.0	0.0	0.0	0.06	3	225	225	0.0	225.0	0.0	0.0	0.06	5
n20w160s10.1	245	245	0.0	245.0	0.0	0.0	0.09	3	245	245	0.0	245.0	0.0	0.0	0.09	3
n20w160s10.2	208	208	0.0	208.0	0.0	0.0	0.07	3	208	208	0.0	208.0	0.0	0.0	0.06	3
n20w160s10.3	210	210	0.0	210.0	0.0	0.0	0.07	3	210	210	0.0	210.0	0.0	0.0	0.08	3
n20w160s10.4	208	208	0.0	208.0	0.0	0.0	0.08	4	208	208	0.0	208.0	0.0	0.0	0.09	4
n20w160s10.5	248	248	0.0	248.0	0.0	0.0	0.08	3	248	248	0.0	248.0	0.0	0.0	0.08	3
n20w180s10.1	254	254	0.0	254.0	0.0	0.0	0.07	3	254	254	0.0	254.0	0.0	0.0	0.07	5
n20w180s10.2	272	272	0.0	272.0	0.0	0.0	0.08	4	272	272	0.0	272.0	0.0	0.0	0.08	4
n20w180s10.3	273	273	0.0	273.0	0.0	0.0	0.10	3	270	270	0.0	270.0	0.0	0.0	0.10	5
n20w180s10.4	206	206	0.0	206.0	0.0	0.0	0.11	4	206	206	0.0	206.0	0.0	0.0	0.11	4
n20w180s10.5	199	199	0.0	199.0	0.0	0.0	0.10	4	199	199	0.0	199.0	0.0	0.0	0.10	4
n20w200s10.1	239	239	0.0	239.0	0.0	0.0	0.09	3	239	239	0.0	239.0	0.0	0.0	0.09	3
n20w200s10.2	213	213	0.0	213.0	0.0	0.0	0.09	5	213	213	0.0	213.0	0.0	0.0	0.09	5
n20w200s10.3	250	250	0.0	250.0	0.0	0.0	0.09	4	250	250	0.0	250.0	0.0	0.0	0.09	4
n20w200s10.4	[a]295	295	0.0	295.0	0.0	0.0	0.10	4	295	295	0.0	295.0	0.0	0.0	0.10	4
n20w200s10.5	233	233	0.0	233.0	0.0	0.0	0.13	5	233	233	0.0	233.0	0.0	0.0	0.13	5
Avg	243.4	243.4	0.00	243.4	0.00	0.00	0.08	3.6	243.1	243.1	0.00	243.1	0.00	0.00	0.08	4.1

[a] optimality not proven.

Table 6
Computational results of 3P-Heu on OT instances with five stations under both recharging policies.

Inst	Full recharge policy						Partial recharge policy					
	Best	Avg	%	σ	Time	ns	Best	Avg	%	σ	Time	ns
n150w120s5.1	750	753.3	0.4	2.4	37.41	9	747	750.1	0.4	2.5	35.47	9
n150w120s5.2	663	665.3	0.3	1.1	33.84	7	657	657.8	0.1	0.6	30.60	6
n150w120s5.3	770	772.1	0.3	1.0	44.76	7	769	770.0	0.1	0.6	43.47	8
n150w120s5.4	735	738.6	0.5	2.6	44.80	6	735	737.6	0.4	2.0	47.44	6
n150w120s5.5	708	712.8	0.7	2.0	45.76	7	708	709.4	0.2	1.0	49.90	8
n150w140s5.1	757	757.6	0.1	0.7	117.24	7	753	754.0	0.1	0.8	125.36	8
n150w140s5.2	773	773.9	0.1	1.0	40.17	8	772	772.0	0.0	0.0	39.52	8
n150w140s5.3	632	635.0	0.5	1.3	35.10	8	626	626.5	0.1	0.9	32.66	8
n150w140s5.4	687	687.0	0.0	0.0	27.86	7	687	687.0	0.0	0.0	28.90	8
n150w140s5.5	674	674.0	0.0	0.0	42.05	6	669	669.0	0.0	0.0	39.33	6
n150w160s5.1	734	735.0	0.1	0.4	61.37	8	729	729.4	0.1	0.7	55.46	7
n150w160s5.2	708	712.9	0.7	3.4	66.66	8	696	698.5	0.4	1.1	57.15	7
n150w160s5.3	637	639.4	0.4	1.0	44.15	7	618	623.9	1.0	3.2	37.01	8
n150w160s5.4	692	693.4	0.2	1.2	38.72	8	692	692.9	0.1	0.7	43.13	8
n150w160s5.5	677	677.9	0.1	0.8	31.50	7	677	677.6	0.1	0.5	33.29	8
n200w120s5.1	818	818.3	0.0	0.5	80.97	6	818	818.3	0.0	0.5	97.46	7
n200w120s5.2	748	748.6	0.1	0.7	76.74	7	746	746.5	0.1	0.5	85.86	9
n200w120s5.3	898	900.8	0.3	1.8	95.75	8	898	900.1	0.2	1.4	110.98	9
n200w120s5.4	790	790.9	0.1	0.3	106.17	7	790	790.9	0.1	0.3	113.57	7
n200w120s5.5	863	866.0	0.3	1.8	173.54	8	862	864.0	0.2	1.0	212.83	9
n200w140s5.1	831	836.3	0.6	2.2	89.79	7	831	836.3	0.6	2.2	98.89	8
n200w140s5.2	797	799.9	0.4	1.4	171.91	7	789	791.0	0.3	1.5	127.63	8
n200w140s5.3	764	765.0	0.1	1.7	71.03	7	763	763.9	0.1	1.4	71.05	8
n200w140s5.4	821	821.0	0.0	0.0	86.18	8	816	816.0	0.0	0.0	80.26	11
n200w140s5.5	838	838.0	0.0	0.0	103.40	8	836	836.1	0.0	0.3	110.50	8
Avg	750.6	752.5	0.26	1.18	70.67	7.3	747.4	748.8	0.19	0.95	72.31	7.9

Table 7
Computational results of 3P-Heu on OT instances with ten stations under both recharging policies.

Inst	Full recharge policy						Partial recharge policy					
	Best	Avg	%	σ	Time	ns	Best	Avg	%	σ	Time	ns
n150w120s10.1	746	746.7	0.1	0.5	30.45	7	740	740.9	0.1	0.3	28.45	10
n150w120s10.2	653	654.7	0.3	1.0	28.71	9	653	654.6	0.2	0.9	29.45	9
n150w120s10.3	766	768.3	0.3	1.1	37.61	8	765	766.4	0.2	0.9	39.17	10
n150w120s10.4	721	722.9	0.3	1.4	40.55	6	721	722.1	0.2	0.8	41.85	7
n150w120s10.5	693	694.6	0.2	1.1	26.72	8	693	693.0	0.0	0.0	26.76	9
n150w140s10.1	747	747.3	0.0	0.5	60.50	8	744	745.2	0.2	1.1	69.45	9
n150w140s10.2	768	768.0	0.0	0.0	35.42	8	764	764.0	0.0	0.0	34.83	7
n150w140s10.3	627	628.2	0.2	1.0	32.76	9	623	623.5	0.1	0.9	32.58	9
n150w140s10.4	683	683.0	0.0	0.0	27.22	9	683	683.0	0.0	0.0	28.20	9
n150w140s10.5	673	673.0	0.0	0.0	41.33	8	668	668.0	0.0	0.0	39.19	10
n150w160s10.1	713	713.0	0.0	0.0	29.90	7	713	713.0	0.0	0.0	30.63	7
n150w160s10.2	700	704.1	0.6	3.0	59.46	6	688	689.4	0.2	0.8	52.69	8
n150w160s10.3	628	630.2	0.4	1.7	39.82	7	617	618.7	0.3	1.7	36.79	9
n150w160s10.4	686	686.4	0.1	0.7	34.39	9	686	686.4	0.1	0.7	37.15	9
n150w160s10.5	676	676.1	0.0	0.3	31.09	7	673	673.0	0.0	0.0	32.48	8
n200w120s10.1	806	806.1	0.0	0.3	65.82	8	805	805.9	0.1	0.3	66.96	9
n200w120s10.2	738	738.1	0.0	0.3	60.01	9	736	736.9	0.1	0.3	62.62	10
n200w120s10.3	891	891.5	0.1	0.5	74.06	8	891	891.5	0.1	0.5	80.90	10
n200w120s10.4	790	790.9	0.1	0.3	106.08	7	790	790.9	0.1	0.3	113.74	7
n200w120s10.5	857	857.8	0.1	0.6	89.71	10	854	854.2	0.0	0.4	87.33	10
n200w140s10.1	831	833.4	0.3	1.3	89.53	8	827	829.1	0.3	1.1	85.82	10
n200w140s10.2	789	789.8	0.1	0.6	107.43	8	778	780.1	0.3	1.1	82.85	9
n200w140s10.3	763	764.5	0.2	1.8	72.18	7	762	763.3	0.2	1.4	71.98	9
n200w140s10.4	820	820.0	0.0	0.0	84.88	8	812	812.0	0.0	0.0	70.13	11
n200w140s10.5	830	830.9	0.1	0.3	78.74	8	830	830.9	0.1	0.3	103.84	9
Avg	743.8	744.8	0.13	0.73	55.37	7.9	740.6	741.4	0.11	0.56	55.43	9.0

Tables 6 and 7 report the computational results achieved by 3P-Heu on the OT instances with five and ten recharging stations under both recharging policies. The optimal solutions are not known for these instances, so it is harder to assess the quality of the results.

With five stations and full-recharge policy, the average cost of the best solution found at each iteration is 0.26% far from the best-known and the standard deviation is 1.18, while with the partial-recharge policy, the average distance is 0.19% and the standard deviation 0.95. The average computing time is a bit more than a minute. Not surprisingly, the average number of stops with the partial-recharge policy is higher than with the full-recharge policy (7.9 vs 7.3).

Similar results can be seen for OT instances with ten stations (see Table 7). Under full-recharge policy the best solution found at each iteration is 0.13% far from the best-known and the standard deviation is 0.73, while 0.11% and 0.56 are the corresponding values under the partial-recharge policy. A higher increase in the number of stops can be observed here (7.9 vs 9.0).

6.3.1. Computational performance on TSPTW instances

The Three-Phase Heuristic can easily be adapted to solve TSPTW instances by simply removing Phase 3 and taking the best solution found with the first two phases only. In Table 8, we report a comparison between the computational results of da Silva and Urrutia (2010) and of 3P-Heu on the TSPTW instances of Ohlmann and Thomas (2007). The meaning of each column is the same of the previous tables. The same parameter setting of the previous experiments was kept apart from the number of iterations that was decreased to 100 (i.e., MaxIter = 500) in order to have computing times similar to those of the algorithm of da Silva and Urrutia (2010), which was run on a slower machine (i.e., a Pentium 4 2.40 GHz).

As indicated in Section 5, the first two phases of 3P-Heu are strongly inspired by the algorithms of da Silva and Urrutia (2010) and Mladenović et al. (2012), so the computational performance of the General VNS of da Silva and Urrutia (2010) and 3P-Heu are rather similar.

6.3.2. Tests on E-VRPTW instances of Schneider et al. (2014) with a single vehicle

The E-TSPTW considered in this paper is a special case of the E-VRPTW problem investigated by Schneider et al. (2014), in particular, the E-TSPTW does not consider the capacity (in terms of loading) of the vehicle and a single vehicle, instead of multiple vehicles, is to be used. Schneider et al. (2014) tested their algorithm on an extensive set of instances; of these instances, we considered the 13 instances with a single vehicle, and we tested both PBF and 3P-Heu on them.

Table 9 compares the results of the CF and the PBF on the 13 instances of Schneider et al. (2014). Columns have the same meaning of the previous tables. Notice that the optimal solution of instance RC204-15 is not known. Moreover, formulation CF was tested with $n+1$ copies of each recharging station.

Table 8
Computational results of 3P-Heu on original TSPTW OT instances: comparison with da Silva and Urrutia (2010).

Group	z^*_{TW}	da Silva and Urrutia (2010)						3P-Heu					
		Best	%	Avg	%	σ	Time	Best	%	Avg	%	σ	Time
n150w120	722.0	722.0	0.0	722.3	0.0	0.4	11.8	722.0	0.0	722.2	0.0	0.4	10.5
n150w140	693.8	693.8	0.0	694.8	0.1	0.5	13.3	693.8	0.0	694.6	0.1	0.7	12.3
n150w160	671.0	671.0	0.0	671.2	0.0	0.3	15.0	671.0	0.0	671.6	0.1	0.7	13.1
n200w120	803.6	803.6	0.0	803.9	0.0	0.1	30.3	803.6	0.0	804.2	0.1	0.4	25.8
n200w140	798.0	798.0	0.0	799.5	0.2	1.1	38.0	798.2	0.0	798.9	0.1	1.0	30.0
Avg	737.7	737.7	0.0	738.3	0.1	0.5	21.7	737.7	0.0	738.3	0.1	0.6	18.3

Table 9
Computational results of CF and PBF on the E-VRPTW instances of Schneider et al. (2014) with a single vehicle.

Inst	z^*_{ETW}	CF							PBF						
		LP	%	bLB	%	bUB	%	Time	LP	%	bLB	%	bUB	%	Time
C103-5	176.05	127.85	72.6	176.05	100.0	176.05	100.0	0.2	151.93	86.3	176.05	100.0	176.05	100.0	0.1
C206-5	242.56	149.02	61.4	242.56	100.0	242.56	100.0	0.3	218.13	89.9	242.56	100.0	242.56	100.0	0.0
C208-5	158.48	110.60	69.8	158.48	100.0	158.48	100.0	0.3	110.19	69.5	158.48	100.0	158.48	100.0	0.0
R202-5	128.78	114.46	88.9	128.78	100.0	128.78	100.0	0.1	125.88	97.8	128.78	100.0	128.78	100.0	0.0
R203-5	179.06	135.00	75.4	179.06	100.0	179.06	100.0	0.3	175.63	98.1	179.06	100.0	179.06	100.0	0.0
RC204-5	176.39	98.44	55.8	176.39	100.0	176.39	100.0	0.4	100.96	57.2	176.39	100.0	176.39	100.0	0.1
RC208-5	167.98	95.56	56.9	167.98	100.0	167.98	100.0	0.2	155.25	92.4	167.98	100.0	167.98	100.0	0.2
C202-10	304.06	198.78	65.4	258.78	85.1	304.06	100.0	7200.0	257.41	84.7	304.06	100.0	304.06	100.0	0.1
R201-10	241.51	177.62	73.5	241.51	100.0	241.51	100.0	4.0	214.51	88.8	241.51	100.0	241.51	100.0	0.1
R203-10	218.21	149.87	68.7	218.21	100.0	218.21	100.0	4.0	197.13	90.3	218.21	100.0	218.21	100.0	0.4
RC201-10	412.86	254.39	61.6	335.75	81.3	412.86	100.0	7200.0	404.28	97.9	412.86	100.0	412.86	100.0	0.1
R209-15	313.24	223.11	71.2	263.70	84.2	–	–	7200.0	220.67	70.4	313.24	100.0	313.24	100.0	6.5
RC204-15	[a]384.86	212.65	55.3	234.25	60.9	–	–	7200.0	245.88	63.9	313.69	81.5	389.55	101.2	7200.0
Avg			67.4		93.2		100.0	2216.1		83.6		98.6		100.1	554.4

[a] Optimality not proven.

Table 10
Computational results of 3P-Heu on the E-VRPTW instances with one vehicle of Schneider et al. (2014).

Inst	z^*_{ETW}	Schneider et al. (2014)			3P-Heu					
		Best	%	Time	Best	%	Avg	%	σ	Time
C103-5	176.05	176.05	100.0	0.12	176.05	100.0	176.05	100.0	0.0	0.02
C206-5	242.56	242.56	100.0	0.14	242.56	100.0	242.56	100.0	0.0	0.02
C208-5	158.48	158.48	100.0	0.11	158.48	100.0	158.48	100.0	0.0	0.02
R202-5	128.78	128.78	100.0	0.11	128.78	100.0	128.78	100.0	0.0	0.03
R203-5	179.06	179.06	100.0	0.15	179.06	100.0	179.06	100.0	0.0	0.06
RC204-5	176.39	176.39	100.0	0.15	176.39	100.0	176.39	100.0	0.0	0.16
RC208-5	167.98	167.98	100.0	0.13	167.98	100.0	167.98	100.0	0.0	0.04
C202-10	304.06	304.06	100.0	0.71	304.06	100.0	304.06	100.0	0.0	0.07
R201-10	241.51	241.51	100.0	0.78	241.51	100.0	241.51	100.0	0.0	0.05
R203-10	218.21	218.21	100.0	0.71	218.21	100.0	218.21	100.0	0.0	0.25
RC201-10	412.86	412.86	100.0	0.90	412.86	100.0	412.86	100.0	0.0	0.10
R209-15	313.24	313.24	100.0	13.73	313.24	100.0	313.24	100.0	0.0	0.51
RC204-15	[a]384.86	384.86	100.0	15.57	384.86	100.0	386.16	100.3	1.7	9.07
Avg			100.0	2.6		100.0		100.0	0.1	0.8

[a] Optimality not proven.

Formulation PBF clearly outperforms CF both in terms of instances solved to optimality and computing time. Instance RC204-15 could not be solved by any of the two formulations. It is worth noticing that the linear relaxation of PBF provides much better lower bounds than the linear relaxation of CF.

Finally, a comparison on the 13 instances between 3P-Heu and the hybrid heuristic of Schneider et al. (2014) is reported in Table 10. The results of Schneider et al. (2014) were achieved on an Intel Core i5 750 processor clocked at 2.67 GHz.

The table indicates the both heuristic algorithms managed to find the best-known solution on all 13 instances. The only instance where 3P-Heu could not find the best known solution in all ten runs was RC204-15. Even in terms of computing times the two algorithms seem equally good, even though 3P-Heu seems to perform better on instance R209-15.

7. Conclusions

In this paper, we addressed a generalization of the well-known *Traveling Salesman Problem with Time Windows* (TSPTW) that arises when electric vehicles are used; the main difference lies in the limited capacity of the batteries of such vehicles that require intermediate stops at recharging stations. We called this new problem *Electric TSPTW* (E-TSPTW).

We proposed a compact formulation of the problem and an alternative formulation with exponentially many variables with respect to the number of recharging stations. We also showed a few preprocessing rules to limit the number of variables. Starting from the state-of-the-art heuristic algorithms for the TSPTW, we illustrated a Three-Phase Heuristic algorithm based on General Variable Neighborhood Search and Dynamic Programming. The proposed approaches have been applied to two different recharging policies: full (the battery is fully recharged at any stop) and partial (the amount to recharge is a decision variable).

Computational results on newly generated instances showed that the proposed alternative formulation could solve 20-customer instances under both recharging policies in short amount of computing times. The heuristic algorithm was able to achieve upper bounds of very good quality on such 20-customer instances in around a tenth of a second. Further computational results of the heuristic algorithm on large-scale instances with 150 and 200 customers were also reported.

We believe that a few possible directions for future research can be: (a) investigating the potential of the alternative formulation in a column-and-cut generation framework; (b) testing a multi-phase approach, where hard constraints are added in sequential phases, on other problems encountered when using electric vehicles instead of traditional internal combustion commercial vehicles; (c) addressing other sources of cost in the objective function along with traveling distances in order to assess the environmental impact/cost of electric vehicles in real-life distribution problems.

Acknowledgments

This work is a part of the SELECT (Suitable ELEctromobility for Commercial Transport) project funded by the Danish Strategic Research Council. This support is gratefully acknowledged. The authors gratefully acknowledge also the four anonymous referees for their valuable comments that contributed to improve the quality of the paper.

Appendix A

This section reports detailed computational results of formulations CF and PBF on all G instances. Columns report the following data: instance name (Inst); optimal solution cost (z^*_{ETW}) – notice that, on instance n20w200s10.4 of Table 13, the optimality of 295 was not proven; lower bound (LP) achieved by the linear relaxation (and, in parenthesis, the corresponding

Table 11
Detailed computational results of CF and PBF on G instances with five stations and full-recharge policy.

Inst	z^*_{ETW}	CF							PBF						
		LP	%	bLB	%	bUB	%	Time	LP	%	bLB	%	bUB	%	Time
n20w120s5.1	271	206.5	(76.2)	271.0	(100.0)	271	(100.0)	2844.5	255.5	(94.3)	271.0	(100.0)	271	100.0	0.3
n20w120s5.2	233	171.3	(73.5)	194.6	(83.5)			7200.0	182.5	(78.3)	233.0	(100.0)	233	100.0	906.2
n20w120s5.3	317	206.5	(65.1)	252.5	(79.7)			7200.0	265.5	(83.8)	317.0	(100.0)	317	100.0	191.9
n20w120s5.4	314	217.0	(69.1)	256.2	(81.6)			7200.0	254.0	(80.9)	314.0	(100.0)	314	100.0	288.6
n20w120s5.5	249	196.5	(78.9)	249.0	(100.0)	249	(100.0)	1420.4	237.0	(95.2)	249.0	(100.0)	249	100.0	0.6
n20w140s5.1	181	138.0	(76.2)	158.4	(87.5)			7200.0	135.0	(74.6)	181.0	(100.0)	181	100.0	133.4
n20w140s5.2	279	193.0	(69.2)	222.5	(79.8)			7200.0	227.0	(81.4)	279.0	(100.0)	279	100.0	530.6
n20w140s5.3	238	177.5	(74.6)	192.5	(80.9)			7200.0	185.0	(77.7)	238.0	(100.0)	238	100.0	22.6
n20w140s5.4	265	180.8	(68.2)	196.9	(74.3)			7200.0	181.0	(68.3)	265.0	(100.0)	265	100.0	906.4
n20w140s5.5	229	156.0	(68.1)	229.0	(100.0)	229	(100.0)	2370.5	177.0	(77.3)	229.0	(100.0)	229	100.0	6.0
n20w160s5.1	246	169.5	(68.9)	223.0	(90.7)	246	(100.0)	7200.0	201.0	(81.7)	246.0	(100.0)	246	100.0	48.6
n20w160s5.2	219	164.0	(74.9)	189.1	(86.3)			7200.0	173.0	(79.0)	219.0	(100.0)	219	100.0	10.0
n20w160s5.3	210	151.0	(71.9)	210.0	(100.0)	210	(100.0)	1931.2	166.5	(79.3)	210.0	(100.0)	210	100.0	1.4
n20w160s5.4	208	149.1	(71.7)	180.6	(86.8)	211	(101.4)	7200.0	153.0	(73.6)	208.0	(100.0)	208	100.0	11.8
n20w160s5.5	253	178.2	(70.4)	224.9	(88.9)	254	(100.4)	7200.0	177.0	(70.0)	253.0	(100.0)	253	100.0	69.3
n20w180s5.1	262	196.4	(74.9)	251.3	(95.9)	262	(100.0)	7200.0	216.0	(82.4)	262.0	(100.0)	262	100.0	6.0
n20w180s5.2	273	192.5	(70.5)	273.0	(100.0)	273	(100.0)	3712.7	197.0	(72.2)	273.0	(100.0)	273	100.0	6.8
n20w180s5.3	282	186.7	(66.2)	240.5	(85.3)			7200.0	208.0	(73.8)	282.0	(100.0)	282	100.0	862.8
n20w180s5.4	206	166.8	(80.9)	206.0	(100.0)	206	(100.0)	2648.5	165.0	(80.1)	206.0	(100.0)	206	100.0	272.5
n20w180s5.5	201	150.5	(74.9)	171.5	(85.3)			7200.0	151.0	(75.1)	201.0	(100.0)	201	100.0	2285.6
n20w200s5.1	241	184.0	(76.3)	227.6	(94.4)	241	(100.0)	7200.0	186.0	(77.2)	241.0	(100.0)	241	100.0	75.8
n20w200s5.2	221	179.0	(81.0)	209.4	(94.7)	221	(100.0)	7200.0	179.0	(81.0)	221.0	(100.0)	221	100.0	109.2
n20w200s5.3	254	154.6	(60.9)	179.2	(70.5)			7200.0	172.0	(67.7)	254.0	(100.0)	254	100.0	40.2
n20w200s5.4	295	204.2	(69.2)	227.4	(77.1)			7200.0	204.0	(69.2)	295.0	(100.0)	295	100.0	514.8
n20w200s5.5	240	194.1	(80.9)	240.0	(100.0)	240	(100.0)	2986.6	195.0	(81.3)	240.0	(100.0)	240	100.0	143.6
Avg			(72.5)		(88.9)		(100.1)	5900.6		(78.2)		(100.0)		(100.0)	297.8

Table 12
Detailed computational results of CF and PBF on G instances with five stations and partial-recharge policy.

Inst	z^*_{ETW}	CF							PBF						
		LP	%	bLB	%	bUB	%	Time	LP	%	bLB	%	bUB	%	Time
n20w120s5.1	271	206.5	(76.2)	271.0	(100.0)	271	(100.0)	670.8	255.5	(94.3)	271.0	(100.0)	271	(100.0)	0.3
n20w120s5.2	225	171.3	(76.1)	215.5	(95.8)	225	(100.0)	7200.0	182.5	(81.1)	225.0	(100.0)	225	(100.0)	138.3
n20w120s5.3	311	206.5	(66.4)	260.8	(83.9)			7200.0	265.5	(85.4)	311.0	(100.0)	311	(100.0)	134.1
n20w120s5.4	312	217.0	(69.6)	267.2	(85.6)			7200.0	254.0	(81.4)	312.0	(100.0)	312	(100.0)	27.3
n20w120s5.5	249	196.5	(78.9)	249.0	(100.0)	249	(100.0)	518.1	237.0	(95.2)	249.0	(100.0)	249	(100.0)	0.6
n20w140s5.1	180	138.0	(76.7)	180.0	(100.0)	180	(100.0)	1149.6	135.0	(75.0)	180.0	(100.0)	180	(100.0)	3.8
n20w140s5.2	278	193.0	(69.4)	223.2	(80.3)			7200.0	227.0	(81.7)	278.0	(100.0)	278	(100.0)	43.7
n20w140s5.3	238	177.5	(74.6)	238.0	(100.0)	238	(100.0)	2049.2	185.0	(77.7)	238.0	(100.0)	238	(100.0)	10.7
n20w140s5.4	265	180.8	(68.2)	203.0	(76.6)			7200.0	181.0	(68.3)	265.0	(100.0)	265	(100.0)	279.6
n20w140s5.5	229	156.0	(68.1)	229.0	(100.0)	229	(100.0)	5429.5	177.0	(77.3)	229.0	(100.0)	229	(100.0)	3.3
n20w160s5.1	246	169.5	(68.9)	215.1	(87.4)	247	(100.4)	7200.0	201.0	(81.7)	246.0	(100.0)	246	(100.0)	12.4
n20w160s5.2	219	164.0	(74.9)	219.0	(100.0)	219	(100.0)	5235.5	173.0	(79.0)	219.0	(100.0)	219	(100.0)	4.4
n20w160s5.3	210	151.0	(71.9)	210.0	(100.0)	210	(100.0)	160.4	166.5	(79.3)	210.0	(100.0)	210	(100.0)	1.6
n20w160s5.4	208	149.1	(71.7)	183.0	(88.0)	208	(100.0)	7200.0	153.0	(73.6)	208.0	(100.0)	208	(100.0)	13.4
n20w160s5.5	253	178.2	(70.4)	211.7	(83.7)	254	(100.4)	7200.0	177.0	(70.0)	253.0	(100.0)	253	(100.0)	37.6
n20w180s5.1	262	196.3	(74.9)	259.3	(99.0)	262	(100.0)	7200.0	216.0	(82.4)	262.0	(100.0)	262	(100.0)	4.3
n20w180s5.2	273	192.5	(70.5)	266.0	(97.4)	273	(100.0)	7200.0	197.0	(72.2)	273.0	(100.0)	273	(100.0)	4.2
n20w180s5.3	271	186.7	(68.9)	246.9	(91.1)			7200.0	208.0	(76.8)	271.0	(100.0)	271	(100.0)	48.5
n20w180s5.4	206	166.8	(80.9)	196.3	(95.3)	225	(109.2)	7200.0	165.0	(80.1)	206.0	(100.0)	206	(100.0)	106.1
n20w180s5.5	201	150.5	(74.9)	185.8	(92.4)	201	(100.0)	7200.0	151.0	(75.1)	201.0	(100.0)	201	(100.0)	205.9
n20w200s5.1	241	184.0	(76.3)	223.0	(92.5)	241	(100.0)	7200.0	186.0	(77.2)	241.0	(100.0)	241	(100.0)	78.0
n20w200s5.2	221	179.0	(81.0)	203.8	(92.2)	221	(100.0)	7200.0	179.0	(81.0)	221.0	(100.0)	221	(100.0)	71.9
n20w200s5.3	254	154.6	(60.9)	204.3	(80.4)	254	(100.0)	7200.0	172.0	(67.7)	254.0	(100.0)	254	(100.0)	38.2
n20w200s5.4	295	204.2	(69.2)	228.3	(77.4)			7200.0	204.0	(69.2)	295.0	(100.0)	295	(100.0)	280.8
n20w200s5.5	240	194.1	(80.9)	240.0	(100.0)	240	(100.0)	1751.7	195.0	(81.3)	240.0	(100.0)	240	(100.0)	180.0
Avg			(72.8)		(92.0)		(100.1)	5574.6		(78.5)		(100.0)		(100.0)	69.2

Table 13
Detailed Computational Results of CF and PBF on G Instances with Ten Stations and Full-Recharge Policy.

Inst	z^*_{ETW}	CF							PBF						
		LP	%	bLB	%	bUB	%	Time	LP	%	bLB	%	bUB	%	Time
n20w120s10.1	270	163.0	(60.4)	222.0	(82.2)			7200.0	246.0	(91.1)	270.0	(100.0)	270	(100.0)	2.6
n20w120s10.2	222	150.0	(67.6)	192.6	(86.7)			7200.0	179.0	(80.6)	222.0	(100.0)	222	(100.0)	2084.7
n20w120s10.3	312	193.0	(61.9)	244.0	(78.2)			7200.0	265.5	(85.1)	312.0	(100.0)	312	(100.0)	92.4
n20w120s10.4	308	202.5	(65.7)	233.0	(75.6)			7200.0	245.0	(79.5)	308.0	(100.0)	308	(100.0)	70.1
n20w120s10.5	243	183.0	(75.3)	201.1	(82.8)			7200.0	237.0	(97.5)	243.0	(100.0)	243	(100.0)	0.7
n20w140s10.1	179	136.0	(76.0)	177.0	(98.9)	179	(100.0)	7200.0	135.0	(75.4)	179.0	(100.0)	179	(100.0)	6.1
n20w140s10.2	277	174.0	(62.8)	210.0	(75.8)			7200.0	227.0	(81.9)	277.0	(100.0)	277	(100.0)	1858.0
n20w140s10.3	237	165.5	(69.8)	189.7	(80.0)			7200.0	185.0	(78.1)	237.0	(100.0)	237	(100.0)	47.5
n20w140s10.4	260	172.0	(66.2)	187.0	(71.9)			7200.0	181.0	(69.6)	260.0	(100.0)	260	(100.0)	774.3
n20w140s10.5	225	155.0	(68.9)	176.0	(78.2)			7200.0	177.0	(78.7)	225.0	(100.0)	225	(100.0)	7.0
n20w160s10.1	245	162.5	(66.3)	190.7	(77.8)			7200.0	201.0	(82.0)	245.0	(100.0)	245	(100.0)	55.6
n20w160s10.2	208	155.0	(74.5)	176.3	(84.7)			7200.0	173.0	(83.2)	208.0	(100.0)	208	(100.0)	8.7
n20w160s10.3	210	147.0	(70.0)	182.0	(86.7)			7200.0	166.5	(79.3)	210.0	(100.0)	210	(100.0)	4.3
n20w160s10.4	208	147.1	(70.7)	176.6	(84.9)	208	(100.0)	7200.0	153.0	(73.6)	208.0	(100.0)	208	(100.0)	14.9
n20w160s10.5	248	165.0	(66.5)	191.0	(77.0)			7200.0	177.0	(71.4)	248.0	(100.0)	248	(100.0)	260.5
n20w180s10.1	254	165.3	(65.1)	200.2	(78.8)			7200.0	202.0	(79.5)	254.0	(100.0)	254	(100.0)	4.0
n20w180s10.2	272	190.5	(70.0)	221.2	(81.3)			7200.0	197.0	(72.4)	272.0	(100.0)	272	(100.0)	17.0
n20w180s10.3	273	183.2	(67.1)	229.8	(84.2)			7200.0	208.0	(76.2)	273.0	(100.0)	273	(100.0)	235.7
n20w180s10.4	206	163.0	(79.1)	206.0	(100.0)	206	(100.0)	5630.2	165.0	(80.1)	206.0	(100.0)	206	(100.0)	135.7
n20w180s10.5	199	144.0	(72.4)	167.1	(83.9)			7200.0	151.0	(75.9)	199.0	(100.0)	199	(100.0)	1419.0
n20w200s10.1	239	182.0	(76.2)	205.1	(85.8)	239	(100.0)	7200.0	186.0	(77.8)	239.0	(100.0)	239	(100.0)	190.2
n20w200s10.2	213	161.0	(75.6)	203.8	(95.7)	213	(100.0)	7200.0	179.0	(84.0)	213.0	(100.0)	213	(100.0)	52.6
n20w200s10.3	250	145.1	(58.0)	176.2	(70.5)	252	(100.8)	7200.0	172.0	(68.8)	250.0	(100.0)	250	(100.0)	108.8
n20w200s10.4	[a]295	194.2	(65.8)	221.9	(75.2)			7200.0	204.0	(69.2)	262.2	(88.9)	295	(100.0)	7200.0
n20w200s10.5	233	171.9	(73.8)	209.1	(89.7)	233	(100.0)	7200.0	195.0	(83.7)	233.0	(100.0)	233	(100.0)	53.0
Avg			(69.0)		(82.7)		(100.1)	7137.2		(79.0)		(99.6)		(100.0)	588.2

[a] optimality not proven.

Table 14
Detailed Computational Results of CF and PBF on G Instances with Ten Stations and Partial-Recharge Policy.

Inst	z^*_{ETW}	CF							PBF						
		LP	%	bLB	%	bUB	%	Time	LP	%	bLB	%	bUB	%	Time
n20w120s10.1	270	163.0	(60.4)	221.0	(81.9)			7200.0	246.0	(91.1)	270.0	(100.0)	270	(100.0)	0.5
n20w120s10.2	220	150.0	(68.2)	196.0	(89.1)	228	(103.6)	7200.0	179.0	(81.4)	220.0	(100.0)	220	(100.0)	36.5
n20w120s10.3	311	193.0	(62.1)	244.0	(78.5)			7200.0	265.5	(85.4)	311.0	(100.0)	311	(100.0)	331.9
n20w120s10.4	307	202.5	(66.0)	238.5	(77.7)			7200.0	245.0	(79.8)	307.0	(100.0)	307	(100.0)	11.3
n20w120s10.5	243	183.0	(75.3)	205.3	(84.5)	243	(100.0)	7200.0	237.0	(97.5)	243.0	(100.0)	243	(100.0)	0.3
n20w140s10.1	178	136.0	(76.4)	178.0	(100.0)	178	(100.0)	2415.5	135.0	(75.8)	178.0	(100.0)	178	(100.0)	4.2
n20w140s10.2	277	174.0	(62.8)	208.9	(75.4)			7200.0	227.0	(81.9)	277.0	(100.0)	277	(100.0)	158.3
n20w140s10.3	237	165.5	(69.8)	189.5	(80.0)	238	(100.4)	7200.0	185.0	(78.1)	237.0	(100.0)	237	(100.0)	24.2
n20w140s10.4	260	172.0	(66.2)	184.9	(71.1)			7200.0	181.0	(69.6)	260.0	(100.0)	260	(100.0)	384.3
n20w140s10.5	225	155.0	(68.9)	176.1	(78.3)	225	(100.0)	7200.0	177.0	(78.7)	225.0	(100.0)	225	(100.0)	4.4
n20w160s10.1	245	162.5	(66.3)	192.0	(78.3)	245	(100.0)	7200.0	201.0	(82.0)	245.0	(100.0)	245	(100.0)	34.7
n20w160s10.2	208	155.0	(74.5)	181.8	(87.4)	208	(100.0)	7200.0	173.0	(83.2)	208.0	(100.0)	208	(100.0)	5.1
n20w160s10.3	210	147.0	(70.0)	185.3	(88.2)			7200.0	166.5	(79.3)	210.0	(100.0)	210	(100.0)	1.2
n20w160s10.4	208	147.1	(70.7)	180.7	(86.9)	212	(101.9)	7200.0	153.0	(73.6)	208.0	(100.0)	208	(100.0)	9.8
n20w160s10.5	248	165.0	(66.5)	196.6	(79.3)	252	(101.6)	7200.0	177.0	(71.4)	248.0	(100.0)	248	(100.0)	62.4
n20w180s10.1	254	165.3	(65.1)	215.9	(85.0)	255	(100.4)	7200.0	202.0	(79.5)	254.0	(100.0)	254	(100.0)	7.1
n20w180s10.2	272	190.5	(70.0)	219.1	(80.5)	272	(100.0)	7200.0	197.0	(72.4)	272.0	(100.0)	272	(100.0)	6.2
n20w180s10.3	270	183.2	(67.8)	236.9	(87.7)	272	(100.7)	7200.0	208.0	(77.0)	270.0	(100.0)	270	(100.0)	80.2
n20w180s10.4	206	163.0	(79.1)	194.4	(94.3)	215	(104.4)	7200.0	165.0	(80.1)	206.0	(100.0)	206	(100.0)	40.9
n20w180s10.5	199	144.0	(72.4)	172.3	(86.6)	199	(100.0)	7200.0	151.0	(75.9)	199.0	(100.0)	199	(100.0)	296.4
n20w200s10.1	239	182.0	(76.2)	208.3	(87.2)	240	(100.4)	7200.0	186.0	(77.8)	239.0	(100.0)	239	(100.0)	89.0
n20w200s10.2	213	161.0	(75.6)	191.8	(90.0)	214	(100.5)	7200.0	179.0	(84.0)	213.0	(100.0)	213	(100.0)	59.6
n20w200s10.3	250	145.1	(58.0)	166.3	(66.5)	252	(100.8)	7200.0	172.0	(68.8)	250.0	(100.0)	250	(100.0)	60.1
n20w200s10.4	295	194.2	(65.8)	218.5	(74.1)			7200.0	204.0	(69.2)	295.0	(100.0)	295	(100.0)	310.9
n20w200s10.5	233	171.9	(73.8)	222.7	(95.6)	233	(100.0)	7200.0	195.0	(83.7)	233.0	(100.0)	233	(100.0)	94.1
Avg			(69.1)		(83.4)		(100.8)	7008.6		(79.1)		(100.0)		(100.0)	84.5

percentage ratio with respect to z^*_{ETW}, computed as $\frac{100 \cdot \mathtt{LP}}{z^*_{ETW}}$); final best lower bound (bLB – and, in parenthesis, the corresponding percentage ratio with respect to z^*_{ETW}, computed as $\frac{100 \cdot \mathtt{bLB}}{z^*_{ETW}}$); final best upper bound (bUB – and, in parenthesis, the corresponding percentage ratio with respect to z^*_{ETW}, computed as $\frac{100 \cdot \mathtt{bUB}}{z^*_{ETW}}$); and the computing time (Time).

Appendix B. Supplementary data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.tre.2016.01.010.

References

Ascheuer, N., Fischetti, M., Grötschel, M., 2001. Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. Math. Program. Ser. A 90, 475–506.

Balas, E., Simonetti, N., 2001. Linear time dynamic-programming algorithms for new classes of restricted TSPs: a computational study. INFORMS J. Comput. 13 (1), 56–75.

Baldacci, R., Mingozzi, A., Roberti, R., 2011. New state-space relaxations for solving the traveling salesman problem with time windows. INFORMS J. Comput. 24 (3), 356–371.

Bruglieri, M., Pezzella, F., Pisacane, O., Suraci, S., 2015. A variable neighborhood search branching for the electric vehicle routing problem with time windows. Electron. Notes Discrete Math. 47, 221–228.

Conrad, R.G., Figliozzi, M.A., 2011. The recharging vehicle routing problem. In: Proceedings Industrial Engineering Research Conference.

da Silva, R.F., Urrutia, S., 2010. A general VNS heuristic for the traveling salesman problem with time windows. Discrete Optim. 7 (4), 203–211.

Dash, S., Günlük, O., Lodi, A., Tramontani, A., 2010. A time bucket formulation for the TSP with time windows. INFORMS J. Comput. 24 (1), 132–147.

Davis, B.A., Figliozzi, M.A., 2013. A methodology to evaluate the competitiveness of electric delivery trucks. Transp. Res. Part E: Logist. Transp. Rev. 49 (1), 8–23.

Desaulniers, G., Desrosiers, J., Solomon, M.M., 2005. Column Generation. Springer US.

Desaulniers, G., Errico, F., Irnich, S., Schneider, M., 2014. Exact Algorithms for Electric Vehicle-Routing Problems with Time Windows. Technical Report, Les Cahiers du GERAD, Montréal, Canada, 2014. <http://www.logistikplanung.tu-darmstadt.de/media/logplanung/publikationen_4/working_paper_1/bpc_evrptw.pdf>.

Dumas, Y., Desrosiers, J., Gélinas, E., Solomon, M.M., 1995. An optimal algorithm for the traveling salesman problem with time windows. Oper. Res. 43 (2), 367–371.

Erdoğan, S., Miller-Hooks, E., 2012. A green vehicle routing problem. Transp. Res. Part E: Logist. Transp. Rev. 48 (1), 100–114.

FedEx, 2010. FedEx Introduces First All-Electric Trucks To Be Used in the U.S. Parcel Delivery Business. FedEx Global Newsroom. <http://www.news.van.fedex.com/node/16470>.

Felipe, Á., Ortuño, M.T., Righini, G., Tirado, G., 2014. A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. Transp. Res. Part E: Logist. Transp. Rev. 71, 111–128.

Focacci, F., Lodi, A., Milano, M., 2002. A hybrid exact algorithm for the TSPTW. INFORMS J. Comput. 14 (4), 403–417.

Gendreau, M., Hertz, A., Laporte, G., Stan, M., 1998. A generalized insertion heuristic for the traveling salesman problem with time windows. Oper. Res. 46 (3), 330–335.

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

Hansen, P., Mladenović, N., Moreno Pérez, J., 2008. Variable neighbourhood search: methods and applications. 4OR 6 (4), 319–360.

Hiermann, G., Puchinger, J., Hartl, R.F., 2014. The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations. Technical Report, submitted for publication. http://dx.doi.org/10.1016/j.ejor.2016.01.038.

Karabulut, K., Fatih Tasgetiren, M., 2014. A variable iterated greedy algorithm for the traveling salesman problem with time windows. Inf. Sci. 279, 383–395.

López-Ibáñez, M., Blum, C., 2010. Beam-ACO for the travelling salesman problem with time windows. Comput. Oper. Res. 37 (9), 1570–1583.

Mingozzi, A., Bianco, L., Ricciardelli, S., 1997. Dynamic programming strategies for the traveling salesman problem with time window and precedence constraints. Oper. Res. 45 (3), 365–377.

Mladenović, N., Todosijević, R., Urošević, D., 2012. An efficient GVNS for solving traveling salesman problem with time windows. Electron. Notes Discrete Math. 39, 83–90.

Motavalli, J., 2010. Frito-Lay Adds Electric Trucks to Its Fleet. NYTimes.com. <http://wheels.blogs.nytimes.com/2010/09/08/frito-lay-adds-electric-trucksto-its-fleet/>.

Nesterova, N., Quak, H., Balm, S., Roche-Cerasi, I., Tretvik, T., 2013. Project FREVUE Deliverable D1.3: State of the Art of the Electric Freight Vehicles Implementation in City Logistics. TNO and SINTEF. European Commission Seventh Framework Programme.

Ohlmann, J.W., Thomas, B.W., 2007. A compressed-annealing heuristic for the traveling salesman problem with time windows. INFORMS J. Comput. 19 (1), 80–90.

Pelletier, S., Jabali, O., Laporte, G., in press. Goods distribution with electric vehicles: review and research perspectives. Transp. Sci., in press

Preis, H., Frank, S., Nachtigall, K., 2014. Energy-optimized routing of electric vehicles in urban delivery systems. In: Operations Research Proceedings 2012. Springer, pp. 583–588.

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

US Department of Energy, 2009. President Obama Announces $2.4 Billion in Grants to Accelerate the Manufacturing and Deployment of the Next Generation of U.S. Batteries and Electric Vehicles. energy.gov. <http://www.apps1.eere.energy.gov/news/news_detail.cfm/news_id=12697>.

US Environmental Protection Agency, 2015. Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990–2013. EPA 430-R-15-004.

van Duin, J.H.R., Quak, H., Muñuzuri, J., 2010. New challenges for urban consolidation centres: a case study in the hague. Proc.-Soc. Behav. Sci. 2 (3), 6177–6188.

Vigo, D., Toth, P. (Eds.), 2014. Vehicle Routing. Society for Industrial and Applied Mathematics, Philadelphia, PA.