Transportation Research Part E 71 (2014) 111–128

Contents lists available at ScienceDirect

Transportation Research Part E

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

A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges

Ángel Felipe [a], M. Teresa Ortuño [a], Giovanni Righini [b], Gregorio Tirado [a,*]

[a] Departamento de Estadística e Investigación Operativa, Universidad Complutense de Madrid, Spain
[b] Dipartimento di Informatica, Universitá degli Studi di Milano, Italy

ARTICLE INFO

Article history:
Received 13 January 2014
Received in revised form 2 August 2014
Accepted 8 September 2014

Keywords:
Vehicle routing
Electric vehicles
Heuristics
Simulated Annealing

ABSTRACT

This paper presents several heuristics for a variation of the vehicle routing problem in which the transportation fleet is composed of electric vehicles with limited autonomy in need for recharge during their duties. In addition to the routing plan, the amount of energy recharged and the technology used must also be determined. Constructive and local search heuristics are proposed, which are exploited within a non deterministic Simulated Annealing framework. Extensive computational results on varying instances are reported, evaluating the performance of the proposed algorithms and analyzing the distinctive elements of the problem (size, geographical configuration, recharge stations, autonomy, technologies, etc.).

1. Introduction

Electric vehicles (EV) represent a promising opportunity for reducing costs and pollution caused by transport and mobility operations. Although their diffusion is still hampered by a number of limitations, such as limited driving range, long recharging times and high initial cost, there is continuous technological progress to improve the reliability and durability of batteries, charging infrastructures and plug-in solutions. Analogous progress is needed on the side of fleet management and logistics optimization. As pointed out by Touati-Moungla and Jost (2010), whereas the development of EV through battery autonomy grows extensively, the research dedicated to EV routing, recharge stations location and vehicles redistribution has not been significant until the last years. Some works concerning charging network planning can be found in Frade et al. (2011),Chen et al. (2013), Nie and Ghamami (2013) and Cavadas et al. (2014), among others.

Nevertheless, routing is a major aspect of EV management, because efficient EV routing plays a major role for encouraging EV use (for a survey on vehicle routing problems see, for example, Toth and Vigo (2002) and Golden et al. (2008)). One of the new challenges is the EV routing problem, which in addition to the traditional issues of the routing problems, must also consider the particularities of autonomy, charge and battery degradation of the EV.

Among the works dealing with EV routing, in Touati-Moungla and Jost (2010) the authors present a mathematical formulation of the energy shortest path problem and the energy routing problem and expose some relationships between these problems and other well-known routing problems. Barco et al. (2013) present a scheme that coordinates the routing, scheduling of charge and operating costs of EV. Conrad and Figliozzi (2011) introduce the recharging vehicle routing problem (RVRP) wherein vehicles with limited range must service a predetermined set of customers but may recharge at certain

* Corresponding author at: Departamento de Estadística e Investigación Operativa, Facultad de Matemáticas, UCM, Plaza de Ciencias 3, 28040 Madrid, Spain. Tel.: +34 913944377; fax: +34 913944606.
E-mail address: gregoriotd@mat.ucm.es (G. Tirado).

http://dx.doi.org/10.1016/j.tre.2014.09.003
1366-5545/

customer locations in order to continue a tour. The problem is introduced as an extension of the distance-constrained VRP (DCVRP) and theoretical bounds are derived to predict and test solution behavior. Instead of performing recharges only at customer locations, Erdogan and Miller-Hooks (2012) consider the existence of locations devoted only to recharge stations. The problem introduced in their paper is called the Green Vehicle Routing Problem (GVRP). It is a variation of the VRP in which the fleet is composed of alternative fuel powered vehicles with limited autonomy in need for recharge during the execution of their duties. This means that the computation of an optimal set of routes must also take into account the problem of deciding where and when each vehicle is recharged (if needed). GVRP considers a limited fuel capacity of the vehicles and the possibility to refuel at alternative fuel stations. The maximum duration of a route is restricted, and for each refueling operation, as well as for each customer visit, a fixed service time is considered.

The model introduced in Erdogan and Miller-Hooks (2012) is, to the best of our knowledge, the first routing model that considers recharge stations. Schneider et al. (2014) extend their work by introducing the Electric Vehicle Routing Problem with Time Windows and Recharge Stations (E-VRPTW), which incorporates the possibility of recharging at any of the available stations with non fixed recharging times, but dependent on the battery charge of the vehicle on arrival at the station. Moreover, capacity constraints on vehicles and customer time windows are included. E-VRPTW aims at minimizing the number of employed vehicles and total traveled distance. More recently, Lin et al. (2014) present a survey of green vehicle routing problems, pointing out that the research on GVRP deals with the optimization of energy consumption of transportation. Some of the GVRP works deal with computing and minimizing fuel consumption, (see Kara et al. (2007),Kuo (2010), Preis et al. (2012),Xiao et al. (2012); or Küçükoolu et al. (2013)), whereas Erdogan and Miller-Hooks (2012) and Schneider et al. (2014) address problems related to optimizing when and where to refuel or recharge the vehicles in order to minimize total energy cost. The authors of this survey suggest that those last models should be extended in order to introduce additional constraints.

This paper approaches the Green Vehicle Routing Problem with Multiple Technologies and Partial Recharges (GVRP-MTPR). It extends the GVRP introduced by Erdogan and Miller-Hooks (2012) in an alternative way as in Schneider et al. (2014), by including several realistic considerations. First, we consider the possibility of performing a partial recharge at a station. This implies that for each vehicle we have to decide where and when to recharge, but also how much. This extension is motivated by the potential cost savings that can be obtained by using partial recharges, because they could save recharge time and thus facilitate meeting the maximum duration constraint and reduce the number of vehicles needed, and also the energy that was not recharged *en route* could be recharged at the depot at a much cheaper cost.

The cost due to battery amortization is also considered in the objective function. Besides, a battery recharge operation can be done in different ways with different technologies, implying different recharging time and cost. A charging point is comparable to a petrol pump, but in the case of EV it is done by plugging the car to the electric grid. With the kind of recharge on a conventional household plug, the charging usually takes several hours, and thus it is usually used to recharge the vehicles at the depot during the night. This is the cheapest technology available and, as a consequence, vehicles will usually depart from the depot with fully charged batteries. There are, however, some existing technologies that allow for faster, though more expensive, recharges. One example is the CHAdeMO protocol (see, for example, Paschero et al. (2013)), that allows for full battery recharges in around one hour. Furthermore, there are also some wireless charging systems that work while the vehicles stand on a platform and allow for recharges around twice as fast as CHAdeMO, at the expense of a higher cost. In any case, this area is highly research active nowadays.

To the best of authors' knowledge, a problem with the above described characteristics, the GVRP-MTPR, has not been approached in the literature yet. In this paper a mathematical programming formulation for the GVRP-MTPR is proposed, which can be used to solve small instances of the problem. However, since solving it to optimality for realistic sized instances is prohibitive, we decided to focus on heuristic approaches. Many different heuristics have been successfully applied to vehicle routing problems with additional constraints, providing near optimal solutions within a short computational time. A good example of this is the heuristic presented in Pisinger and Ropke (2007), that can be applied to different versions of the vehicle routing problem. In Vidal et al. (2012), a survey on heuristics and metaheuristics for different versions of the Vehicle Routing Problem can also be found. Furthermore, previous approaches to problems similar to the one approached in this paper also focused on heuristics: Erdogan and Miller-Hooks (2012) proposed two constructive heuristics, a modified Clarke and Wright savings heuristic and a Density-Based Clustering algorithm, and Schneider et al. (2014) used a hybrid approach combining Variable Neighborhood Search and Tabu Search. This paper introduces several constructive and local search heuristics that are exploited within a Simulated Annealing (SA) algorithm. Simulated Annealing has been successfully applied to many different kinds of problems (Suman and Kumar (2006), provide a survey on applications of SA), and in particular to vehicle routing problems (see, for example, Kuo (2010), where a SA algorithm is proposed for finding the vehicle routing with the lowest total fuel consumption).

The main contributions of the paper can be summarized as follows. We study a routing problem with electric vehicles (the GVRP-MTPR) that extends previously formulated problems by considering additional realistic elements, such as the use of different recharge technologies and the possibility of performing partial recharges. Both of them have a wide potential to provide energy consumption savings. A mathematical formulation for this new problem is provided, together with several fast heuristics that are able to construct good quality solutions for real size instances. Finally, a comprehensive computational study to evaluate their performance and the impact of the new elements of the problem is presented. For this purpose, we have created a benchmark for the GVRP-MTPR, including instances with different configurations, and we have also considered some additional instances available in the literature for similar problems: the E-VRPTW instances of Schneider et al.

(2014) and the GVRP instances of Erdogan and Miller-Hooks (2012). The experiments have focused on evaluating the performance of the proposed algorithms and analyzing some of the main characteristics of the problems considered, mainly regarding the size and geographical configuration of the instances, the number of recharge stations, the use of multiple technologies and partial recharges and the autonomy of the vehicles.

The rest of the paper is organized as follows. A detailed description of the considered problem, together with its mathematical formulation, are given in Section 2. Section 3 presents the solution methods, including constructive algorithms, deterministic local search heuristics and a metaheuristic based on a Simulated Annealing framework. Extensive computational experiments are presented in Section 4, where the performance of the proposed algorithms is evaluated, and in Section 5, which analyzes the main distinctive features of the problem. Section 6 concludes with some final remarks and ideas for possible further extensions.

2. Problem description and formulation

The Green Vehicle Routing Problem with Multiple Technologies and Partial Recharges (GVRP-MTPR) can be formulated as follows.

Let $\mathcal{G} = (\mathcal{N} \cup \mathcal{R}, \mathcal{A})$ be a given weighted directed graph whose node set is the union of a set $\mathcal{N}$ of customers and a set $\mathcal{R}$ of recharge stations. One distinguished recharge station in $\mathcal{R}$ is the depot where vehicle routes start and terminate.

Each customer $i \in \mathcal{N}$ is characterized by a demand q_i (expressed in kg) and a service time s_i (expressed in hours). All customer nodes in $\mathcal{N}$ must be visited by a single vehicle; split delivery is not allowed.

Nodes in $\mathcal{R}$ can be visited at any time. Multiple visits to them are allowed (also simultaneously) and partial recharge is also allowed. We consider a set $\mathcal{T}$ of different technologies for battery recharge. For each technology $t \in \mathcal{T}$ we assume a given recharge speed ρ_t (expressed in KWh per hour), and a given recharge unit cost γ_t (expressed in Euro per KWh). We indicate by $\mathcal{T}_i \subseteq \mathcal{T}$ the subset of technologies available at each station $i \in \mathcal{R}$. An additional recharge technology to be used during the night is always available at the depot; this technology allows for a cheaper recharge, but it is much slower than all other available technologies and thus it cannot be used during the day. The recharge unit cost of this technology is denoted by γ^*.

Non-negative coefficients d_{ij} and tt_{ij} are associated with each arc $(i,j) \in \mathcal{A}$, to represent respectively the distance (expressed in km) and the travel time (expressed in hours) for traveling from $i \in \mathcal{N} \cup \mathcal{R}$ to $j \in \mathcal{N} \cup \mathcal{R}$.

We consider a fleet made of m identical vehicles with given capacity Q (expressed in kg), and equipped with batteries of capacity B (expressed in KWh). The energy consumption is assumed to be proportional to the distance traveled through a given coefficient π (expressed in KWh per km), and thus the autonomy of a vehicle is $AT = \frac{B}{\pi}$. The duration of each route is required not to exceed a given limit T (expressed in hours) representing the duration of drivers' work shifts. The autonomy of the vehicles, together with the driving time limit, can give an idea of the number of *en route* recharges that may be needed. In many real cases in which the shifts correspond to working days it can be observed that one single *en route* recharge is enough to complete any route within the driving time limit, and thus this assumption can be made for some classes of realistic instances. This will be later discussed in Section 4.

A feasible route is a sequence of nodes complying with the following set of constraints:

- the route must start and end at the depot;
- the overall amount of goods delivered along the route, given by the sum of the demands q_i for each visited customer $i \in \mathcal{N}$, must not exceed the vehicle capacity Q;
- the total duration of a route must not exceed the total allowed duration T; the route duration is given by three terms:
  - traveling time, i.e. the sum of the terms tt_{ij} for each traversed arc $(i,j) \in \mathcal{A}$;
  - service time, i.e. the sum of the terms s_i for each visited vertex $i \in \mathcal{N}$;
  - recharge time at the stations: there is a fixed recharge time, denoted by f_i, and a variable time, owing to the possibility of partial recharge, which is a decision variable;
- the level of battery charge must always be between 0 and B, taking into account that
  - whenever the vehicle travels from node $i \in \mathcal{N} \cup \mathcal{R}$ to node $j \in \mathcal{N} \cup \mathcal{R}$ the associated energy consumption is given by πd_{ij};
  - the amount of energy recharged at a station $i \in \mathcal{R}$ with technology $t \in \mathcal{T}_i$ is given by ρ_t times the recharge time at $i \in \mathcal{R}$;

A set of feasible routes is a feasible solution if all customers are visited once and no more than m vehicles are used.

The objective is the minimization of the total recharging cost, which is composed by a fixed and a variable cost component. Since batteries allow for a limited number of recharge cycles during their operational life, we associate a fixed cost with each recharge operation; this cost, indicated by ϑ (expressed in Euro), is given by the cost of a battery divided by the estimated number of recharge cycles after which it must be replaced.

To simplify the proposed formulation, let F_0 be a set containing several copies of the recharge stations, including the depot considered as a recharge point, so that multiple visits are not restricted, and let D be a set with m copies of the depot, so that each vehicle will depart from one different node of D. The new graph containing these dummy nodes is denoted by $\mathcal{G}' = (V' = \mathcal{N} \cup F_0 \cup D, \mathcal{A}')$, where $\mathcal{A}'$ is the new set of arcs after adding the new dummy nodes. Note that the distances and travel times associated with the arcs of $\mathcal{A}'$ correspond to the ones of the arcs of $\mathcal{A}$ according to the correspondence of the dummy nodes with the original nodes.

The variable cost of a recharge is proportional to the amount of energy recharged, but it also depends on the chosen recharge technology $t \in \mathcal{T}$ through the coefficient γ_t.

Some of the decision variables and constraints of the model introduced in what follows extend the ones used in the model by Erdogan and Miller-Hooks (2012).

Decision variables:

x_{ij} binary variable stating if there is a vehicle travelling from i to j or not
y_j^A amount of energy available when arriving at node j (KWh)
y_j^L amount of energy available when leaving node j (KWh)
l_j amount of load left in the vehicle after visiting node j (kg)
z_{jt} amount of energy recharged at node j using technology t (KWh)
g_v amount of energy recharged by vehicle v at the depot (KWh)
τ_j departure time from node j (h)

Mathematical formulation:

$$\min \sum_{v\in D}\gamma^* g_v + \sum_{j\in F_0}\sum_{t\in\mathcal{T}_j}\gamma_t z_{jt} + \vartheta \sum_{i\in V'}\sum_{j\in F_0\cup D} x_{ij} \tag{1}$$

$$\sum_{j\in V'} x_{ij} = 1, \quad \forall i \in \mathcal{N} \tag{2}$$

$$\sum_{j\in V'} x_{ij} \leqslant 1, \quad \forall i \in F_0 \tag{3}$$

$$\sum_{j\in V'} x_{ji} - \sum_{j\in V', j\neq i} x_{ij} = 0, \quad \forall i \in V' \tag{4}$$

$$\sum_{j\in V'} x_{vj} = \sum_{j\in V'} x_{jv} \leqslant 1, \quad \forall v \in D \tag{5}$$

$$y_v^L \leqslant g_v, \quad \forall v \in D \tag{6}$$

$$\tau_j \geqslant \tau_i + tt_{ij} + s_i - M_1(1 - x_{ij}), \quad \forall i \in \mathcal{N}, \forall j \in V',\ i \neq j \tag{7}$$

$$\tau_j \geqslant \tau_i + tt_{ij} + \sum_{t\in\mathcal{T}_j}\frac{1}{\rho_t} z_{jt} + f_j - M_1(1 - x_{ij}), \quad \forall i \in F_0, \forall j \in V',\ i \neq j \tag{8}$$

$$\tau_j \geqslant tt_{vj} - M_1(1 - x_{vj}), \quad \forall v \in D, \forall j \in V',\ v \neq j \tag{9}$$

$$0 \leqslant \tau_v \leqslant T, \quad \forall v \in D \tag{10}$$

$$y_j^A \leqslant y_i^L - \pi d_{ij} x_{ij} + B(1 - x_{ij}), \quad \forall (i,j) \in A' \tag{11}$$

$$y_j^L \leqslant B, \quad \forall j \in F_0 \tag{12}$$

$$g_v \leqslant B, \quad \forall v \in D \tag{13}$$

$$y_j^L = y_j^A, \quad \forall j \in \mathcal{N} \tag{14}$$

$$y_j^L = y_j^A + \sum_{t\in\mathcal{T}_j} z_{jt}, \quad \forall j \in F_0 \tag{15}$$

$$l_j \geqslant l_i + q_j - M_2(1 - x_{ij}), \quad \forall i \in V', j \in \mathcal{N}, i \neq j \tag{16}$$

$$l_j \geqslant l_i - M_2(1 - x_{ij}), \quad \forall i \in V', j \in F_0, i \neq j \tag{17}$$

$$l_j \leqslant Q, \quad \forall j \in V' \tag{18}$$

$$x_{ij} \in \{0,1\};\ y_j^L, y_j^A, l_j, z_{jk}, p_j, g_v, \tau_j \geqslant 0, \quad \forall i,j,k,v \tag{19}$$

In the objective function (1), measured in monetary units, the first term represents the cost of the energy recharged at the depot during the night, the second term the cost of the energy recharged at all stations (including the depot) during the day and the third term the fixed costs associated with the battery cycles. Constraints (2) ensure that all customers are visited exactly once and constraints (3) enforce that each copy of a recharge station is visited at most once, while flow conservation constraints are given by (4). Constraints (5) impose that each of the m available vehicles is used at most once and that every used vehicle starts and ends its route at the depot. Constraints (6) ensure that the amount of energy available when leaving the depot is not larger than the amount recharged during the night. Note that all vehicles will arrive at the depot with empty batteries in any optimal solution, and if any recharge is performed during one route, that vehicle will depart with a full battery from the depot, because the night recharge is the cheapest one. However, if no recharge is necessary, it may be sufficient to perform a partial recharge at the depot during the night.

Constraints (7)–(9) take travel, recharge and service times into account for the definition of the arrival time variables, while constraints (10) ensure that all vehicles get back to the depot on time. Constraints (11) relate the amount of energy consumed for transportation and/or recharged at any station with the energy level variables and constraints 12,13 ensure that the capac-

ity of the battery is not exceeded. Constraints (14) state that the energy levels when reaching and leaving a customer are the same, while constraints (15) ensure that the energy level when leaving a recharge station equals the energy level when arriving plus the amount of energy recharged. Finally, constraints (16)–(18) impose the capacity constraints on the vehicles and constraints (19) define the domain of the variables. M_1 and M_2 are appropriate bounds so that the corresponding constraint becomes redundant if $x_{ij} = 0$, and that y_j^L variables depend on y_j^A and z_{jt}, so they could be eliminated in preprocessing.

Furthermore, imposing that only one technology can be used during each visit to a recharge station can be done by simply stating that, for each $j \in F_0$, the set $\{z_{jk} | k \in \mathcal{T}_j\}$ is a Special Order Set of type 1 (SOS1). This constraint is easily managed internally by most commercial solver packages. However, this condition could also be imposed directly by using a new set of binary variables δ_{jk}, taking value 1 if technology k is used at station j and 0 otherwise, and adding constraints (20) and (21).

$$z_{jk} \leqslant B\delta_{jk} \qquad \forall j \in F_0, \forall k \in \mathcal{T}_j \tag{20}$$

$$\sum_{k \in \mathcal{T}_j} \delta_{jk} \leqslant 1 \qquad \forall j \in F_0 \tag{21}$$

According to the tests we performed, CPLEX failed to solve this model within 12 h of time for instances with only 10 customers. Furthermore, this result is in line with previous results on similar problems, for which several instances of the same size could not be solved to optimality (see, for example, the E-VRPTW instances in Schneider et al. (2014)). Provided that the contribution is rather limited, we decided to focus on heuristics, to obtain good solutions for realistic instances.

3. Solution methods

This section presents several solution methods to solve the GVRP-MTPR. Section 3.1 is focused on constructive algorithms, aimed to produce feasible solutions within short computational time. In Section 3.2 several neighborhood structures based on different operators are described. Each of them defines a local search procedure, that could be used individually or in combination to build improving solutions for the problem. Finally, Section 3.3 presents an alternative solution method based on Simulated Annealing.

3.1. Constructive algorithms

In this section a constructive heuristic based on a greedy generation method is presented. It creates feasible solutions by starting from an empty solution and extending it iteratively until a complete solution is constructed. This algorithm aims to produce a variety of initial feasible solutions with small computational effort to feed the local search algorithms that are presented in the following sections and be used as an initialization method for them.

Two sorted lists containing the closest customers and the closest recharge stations to each location are needed by the algorithm. However, since they do not change during its execution, they are calculated just once at the beginning of the algorithm.

The proposed constructive heuristic, called k-PseudoGreedy, is presented in Algorithm 1. If $k = 1$, it reduces to a nearest neighbor greedy algorithm, visiting the closest customer and recharging at the closest recharge station when needed, until the capacity or time limit is reached. If $k > 1$, the next node to be visited is chosen randomly from the set of k closest candidates (see Step 3), allowing the generation of different feasible solutions in different runs.

This algorithm is focused more on feasibility and diversification, producing multiple different feasible solutions, rather than on quality or cost, that are to be improved later. However, if k is small, the produced solutions are expected to have an acceptable quality.

Algorithm 1. k-PseudoGreedy (k-PG)

1: Initialize solution by setting $h := 1$ and $i := 0$.
2: **repeat**
3: Find up to k unvisited customers that are closest to i and are reachable according to capacity, autonomy and time availability, and select one of them, say j, at random.
4: **if** (it is possible to reach the depot directly from j) **then**
5: Add j to route h and set $i := j$.
6: **else**
7: **if** (it is possible to reach the depot from j by visiting a recharge node r in between) **then**
8: Add j and r to route h, performing a full recharge with the fastest technology available, and set $i := r$.
9: **else**
10: Add the depot to route h, adjusting recharges so that the battery level is null when arriving at the depot, and set $h := h + 1, i := 0$.
11: **end if**
12: **end if**
13: **until** all customers are served

The solutions generated by this algorithm are used as initial solutions for the heuristics presented in the next two sections.

3.2. Deterministic local search

Local search (see, for example, Aarts and Lenstra (2003)) is a technique that has been used successfully in many hard optimization problems. It is based on the iterative improvement of the solutions by performing appropriate modifications on them via moves, which are usually defined through neighborhood structures. In the next three subsections several neighborhood structures are proposed. These can be used in single neighborhood local search algorithms and they can also be combined to produce multiple neighborhood local search algorithms, as described in SubSection 3.2.4.

3.2.1. Recharge Relocation

If one route of a feasible solution visits at least one recharge station, it may be improved by optimally locating one single recharge at a certain point. Following this idea, the operator Recharge Relocation is intended to find, if needed (and possible), the optimal location of a single recharge point in a route, without modifying the sequence of visits to the customers. This is explained in Algorithm 2.

Algorithm 2. Recharge Relocation (RR)

Input.
- S: Initial feasible solution.

Output.
- S': Feasible solution produced.

Pseudocode.
1: Set $S' := S$.
2: **for** (each route r in S) **do**
3: Let $(p_1, \cdots, p_h)$ be the sequence of customers of route r, excluding recharge stations (p_1 and p_h are necessarily the depot).
4: Calculate $l := \sum_{i=1}^{h-1} d(p_i, p_{i+1})$ and set $r' := r$.
5: **if** $(l > AT)$ **then**
6: Calculate the interval $[a, b]$ in which a single recharge station could be located, where $a := \min\{s = 1, \cdots, h-1 | \sum_{i=s+1}^{h} d(p_{i-1}, p_i) \leqslant AT\}$ and $b := \max\{u = 2, \cdots, h | \sum_{i=1}^{u-1} d(p_i, p_{i+1}) \leqslant AT\}$.
7: **if** $(a \leqslant b)$ **then**
8: For each $i \in [a, b]$ and each recharge station c reachable from p_i, calculate the minimum amount of recharge needed to complete the route $(p_1, \cdots, p_i, c, p_{i+1}, \cdots, p_h)$, using the cheapest technology available verifying the maximum duration constraint. Choose the cheapest alternative and update r'.
9: **end if**
10: **end if**
11: **end for**

3.2.2. 2-opt

For each route of any feasible solution, this operator aims to perform a local search based on 2-interchanges on each leg of the route determined by two consecutive recharges (including the depot as the beginning or end of a route). A 2-interchange is determined by the two edges that are to be removed, since there is only one possible way to reconnect the solution without using the removed edges. If the given route does not visit any recharge station, then it is composed by a single closed leg. However, if any recharge station is visited, the route is composed by several open legs, each one determined by a sequence of edges between the depot and a recharge station or between two recharge stations, if that were the case.

Then, a 2-opt best improve local search is applied sequentially to each leg, implementing the best 2-interchange available at each iteration until no more improving 2-interchanges exist (as shown by Algorithm 3). This procedure is standard for the symmetric TSP, both for closed or open routes. For more details see, for example, Syslo et al. (1983).

Algorithm 3. 2-opt (2O)

Input.
- S: Initial feasible solution.

Output.
- S': New solution produced.

Pseudocode.
1: Set $S' := S$.
2: **for** (each route r in S') **do**

(*continued*)

Algorithm 3. 2-opt (2O)

```
3:  for (each route leg l of r) do
4:    repeat
5:      Evaluate all 2-interchanges on l and choose the best one, called v.
6:      if (v improves l) then
7:        Perform move v on S'.
8:      end if
9:    until (v is non-improving)
10:  end for
11: end for
```

3.2.3. Reinsertion

In most real cases the autonomy of the vehicles, together with the limited average speed of traveling and the maximum duration of routes, make that feasible routes visit one recharge station at most. So for most instances assuming this property of the routes is not unrealistic and it provides remarkable advantages from the computational viewpoint. The *Reinsertion* neighborhood is then explored when this assumption can be made. Customers are relocated from one route to another with the purpose of eliminating routes, eliminating recharges or just decreasing energy consumption.

First, the savings achieved by removing each customer from its route are calculated. To do this, the lengths of the links that are not traversed after removing the corresponding node are subtracted and the length of the new link needed to reconnect the route is added, as it is done in standard vehicle routing problems. However, now we must also check if the removal of the customer allows to eliminate any recharge along the route, due to the energy saved; in that case, additional savings are obtained by skipping the corresponding recharge station. This information should be updated every time the solution is modified.

Algorithm 4. Reinsertion$_b^u$ ($\mathbf{R}_b^u$)

Input.
- S: Initial feasible solution.
- n: Number of iterations.

Output.
- S': New solution produced.

Pseudocode.

```
1: Set S' := S.
2: Calculate the savings g_i produced by the removal of each customer i of S' and sort them.
3: for (j := 1, n) do
4:   Consider the customers in decreasing order of g_i. For each of these customers c, select the route r of the customer
     that is closest to c and visited by a different vehicle and calculate the total cost saving that would be obtained if
     customer c is removed from its current route and inserted in route r without modifying the sequence of the other
     customers already being served by r.
5:   if (b = 1) then
6:     Select the move v involving customer i with the highest saving.
7:   else
8:     Select the first move v found involving customer i with a positive saving.
9:   end if
10:  if (no improving move v is found) then
11:    return
12:  else
13:    Perform move v on S'.
14:    if (u = 1) then
15:      Update the savings g_i concerning the customers i belonging to routes being affected by the move performed
     and sort them again.
16:    else
17:      Eliminate g_i from the list of savings.
18:    end if
19:  end if
20: end for
```

The customer to be relocated and its new location can be calculated according to two parameters:

- b: a binary parameter indicating whether first-improve (0) or best-improve (1) strategy is used;
- u: a binary parameter indicating whether the values of savings are updated (1) or not (0) at each iteration.

The values of these two parameters determine the version of the algorithm, as described in Algorithm 4.

3.2.4. Local search combinations

Algorithms $RR, 2O$ and R_b^u are improving heuristics that can be applied to any initial feasible solution. Furthermore, varying the values of u and b, there are 4 possible versions of algorithm R_b^u, for a total of 6 single local search algorithms.

However, they could also be combined and applied consecutively, leading to more complex local search procedures. RR and $2O$ can be combined together or with any version of R_b^u, leading to a total of 18 different algorithms (note that the order is also important). Some of the possibilities considered are, for instance, RR + 2O, 2O + RR, RR+R_1^1, 2O+R_0^1, etc.

It could also be possible to combine RR and $2O$ together and with any of the versions of R_b^u, that leads to 24 different algorithms if all possible orderings are considered (2O + RR+R_1^1, RR+R_1^0+2O, R_0^0+RR + 2O, etc.). Hence, there is a total of 48 combinations of improving algorithms. Based on them, we define one more algorithm, called 48A, consisting of implementing all 48 combinations independently and selecting the best overall solution.

3.3. Simulated Annealing

The algorithms introduced in the previous section are based on deterministic local search procedures, that by definition end up when a local optimum is reached. They can lead to significant improvements on the solution quality in short computational times, but they do not implement any mechanism to escape from poor local optima. In order to provide the algorithm with this kind of mechanism, we have chosen to use a Simulated Annealing framework, using a modified version of Reinsertion as the basic move of that metaheuristic. The detailed pseudocode of the proposed method is given in Algorithm 5. For more details on the standard Simulated Annealing we refer the reader to Suman and Kumar (2006).

The customer to be removed is selected at random with probabilities proportional to the savings produced. The route in which it is reinserted is selected at random with uniform probability, to introduce diversification, and the optimal insertion point is computed. Every time the current best known solution is updated, neighborhoods RR and 2O are explored with a deterministic local search to possibly improve it even more. Furthermore the same is done with neighborhood R_1^1 at the end of the SA algorithm, as a final intensification step. The calibration of the initial and final temperatures, the cooling schedule and the number of iterations per temperature is described in Section 4.2.2.

4. Evaluation of the algorithms performance

This section presents the computational study carried out to analyze the performance of the proposed algorithms. First, the instance sets considered for experimentation, together with some implementation details, are given in SubSection 4.1. SubSection 4.2 analyzes the results obtained with local search and Simulated Annealing. The results obtained on another set of instances adapted from the ones proposed in Schneider et al. (2014) are shown in SubSection 4.3, together with a study on the impact of the capacity of the battery. Finally, since there is not any GVRP-MTPR benchmark available in the literature and thus a direct comparison with the state-of-the-art methods could not be performed, some additional tests on a particular case (the GVRP) are presented in Section 4.4 in order to further evaluate the performance of the algorithms.

Algorithm 5. Simulated Annealing (SA)

Input.
- S: initial feasible solution.
- T_s, T_f: starting and final temperatures.
- ϖ: Temperature decreasing rate.
- i^*: maximum number of iterations per temperature.

Output.
- S': New solution produced.

Pseudocode.

1: Set $S' := S, T := T_s$ and $nP := \left\lceil \frac{\log(T_f/T_s)}{\log \varpi} \right\rceil$.
2: **for** $[p := 1, nP]$ **do**
3: **for** $[ip := 1, i^*]$ **do**
4: Select the customer to be moved, the new route to be considered and the new position of the customer. Let $\widehat{S}$ be the solution after implementing the move.
5: Generate a random number $u \in (0, 1)$.

(*continued*)

Algorithm 5. Simulated Annealing (SA)

```
      if (exp((z(S) − z(Ŝ))/T) ⩾ u) then
        Set S := Ŝ.
      end if
      if (z(S) < z(S′)) then
        Call RR(S) and 2O(S) iteratively until a local optimum S is obtained. Set S′ := S.
      end if
    end for
    Set T := ϖT.
end for
Call R¹₁(S′) until a possibly new local optimum S′ is reached.
```

4.1. Data sets

The proposed algorithms have been implemented in Fortran 95 and executed on an Intel Core i5, 2.8 Ghz, 8 Gb RAM, running Windows 7.

Several sets of new instances to be used for experimentation were randomly generated based on real technical data (see Energy Lab (2011)), as follows.

- Number of customers (N): 100, 200, 400.
- Number of vehicles available: $\frac{N}{4}$.
- Battery capacity: 20 KWh (equivalent to an autonomy of 160 km).
- Energy consumption: 0.125 KWh/km.
- Average speed: 25 km/h.
- Vehicle capacity: 2300 kg.
- Maximum route duration: 8 h.
- Available technologies:
  - Slow (S): 3.600 KWh/h and cost 0.160 €/KWh (conventional household technology, only used at the depot for night recharge).
  - Medium (M): 20.000 KWh/h and cost 0.176 €/KWh (following the CHAdeMO protocol, available at some recharge stations).
  - Fast (F): 45.000 KWh/h and cost 0.192 €/KWh (following a wireless protocol, available at some recharge stations).
- Fixed cost of recharge: 2.270 €/cycle.

As it is done in the classical Solomon instances for the VRP, the number of vehicles available in our instances is one fourth of the number of customers. The autonomy of the vehicles is 160 km, but it could reach up to 320 km with a single *en route* recharge, which is significantly higher than the maximum route length (25 km/h × 8 h = 200 km). Hence, since the slow technology is the cheapest one and all vehicles can leave the depot fully charged, this implies that at most one recharge should be sufficient. Note also that the instances have been generated such that the two fastest technologies are not available at all stations.

We considered two types of configurations: in configuration A the depot is centrally located and there are nine recharge stations (including the depot); in configuration B the depot is at a corner and there are five recharge stations (including the depot), as shown in Fig. 1. In both cases, the customers are randomly generated with uniform probability distribution. For each configuration, three sets of ten instances with 100, 200 and 400 customers were generated, leading to a total of 60 instances. The instances of type A and B are labeled 10 – 19 and 20 – 29 respectively, followed by the number of customers. All these instances are available at http://www.mat.ucm.es/∼gregoriotd/GVRPen.htm and in what follows they will be referred to as FORT instances.

We also used the instances of Schneider et al. (2014) that were proposed for a similar problem with time windows but without partial recharges or multiple technologies. The results obtained on these instances (SSG instances, in the remainder) are given in Section 4.3. Finally we used the GVRP instances of Erdogan and Miller-Hooks (2012), indicated by E&MH instances in the remainder; results are presented in Section 4.4.

4.2. Local search and SA on FORT instances

4.2.1. Local search

In this section we report the results obtained with local search algorithms RR, 2O and R_1^1, and with Algorithm 48A including all tested combinations, starting from 500 pseudo-greedy initial solutions. The average costs and running times on the

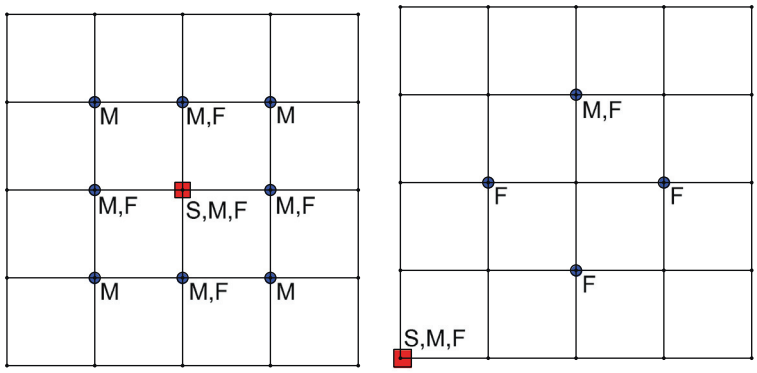


Fig. 1. FORT instances: configurations A and B. The depot is represented by a square and the recharge stations by circles. Recharge technologies are indicated for each station (S = Slow, M = Medium and F = Fast).

FORT instances provided by the selected algorithms are reported in Table 1, where each row is associated with an instance size and each column with a local search method.

As expected, R_1^1 clearly outperforms RR and 2O. Furthermore, it also provides results that are significantly close to the ones obtained by 48A, that includes all considered combinations. These results show that R_1^1 is able to provide significant improvements over RR and 2O, which are very fast, with a short additional running time, while additional improvements provided by 48A require a higher computational effort.

4.2.2. SA calibration

Before running the Simulated Annealing, it is necessary to tune its main parameters (initial and final temperature, cooling rate, number of iterations, number of initial solutions). For this purpose we performed preliminary experiments on twelve additional random instances of different types. The algorithm proved to be robust and from the results obtained we selected the following values: initial temperature $T_s = 42$ (corresponding to a probability 0.7 of accepting a cost increase of 15 euros); final temperature $T_f = 0.05$; cooling rate $\varpi = 0.99$. The number of iterations i^* for each temperature and the number of initial solutions τ determine the trade-off between the computational effort and the expected quality of the solutions.

The best compromise was obtained for $(\tau = 5, i^* = 12500)$ or $(\tau = 25, i^* = 2500)$, for a moderate running time, and for $(\tau = 25, i^* = 12500)$ or $(\tau = 125, i^* = 2500)$, for five times that computation effort. The parameter values used in our experiments depended on the instance size and the available running time. The main conclusions we obtained from our tests is that for small and medium sized instances it is more convenient to use more initial solutions with fewer iterations per solution, in order to increase diversity; however, as the instance size increases, the problem becomes more difficult to solve and it is harder to reach good local optima, being preferable to use fewer initial solutions but to run more iterations for each of them.

We remark that the calibration was performed on a small data set with varying characteristics. We observed that the results did not change significantly when using other parameter values leading to a similar computational effort. Furthermore, the results presented in the remainder show that the algorithm performs well when applied to other sets of instances with different structure.

4.2.3. Comparison between local search and SA

Table 2 displays the costs of the solutions provided by SA and the best solutions found by all proposed algorithms. The last two rows for each set, labeled *%imp over* R_1^1 and *%imp over 48A*, show the average percentage improvement provided by SA with respect to R_1^1 and 48A, respectively. SA clearly outperforms R_1^1 on all sets but one, in which a small negative improvement of −0.39% is observed. However, although SA is superior to 48A in most A instances, this is not the case for B instances. This fact highlights that the structure of the instances affects the performance of the algorithms.

Table 1
Average costs and running times (in seconds) on FORT instances for each instance size (rows) with some selected local search methods (columns).

	Average cost				Average time			
	2O	RR	R_1^1	48A	2O	RR	R_1^1	48A
N = 100	88.03	92.34	66.42	64.96	<1	<1	2	57
N = 200	155.41	161.72	114.65	111.48	<1	<1	15	346
N = 400	279.84	287.62	208.31	203.53	<1	<1	91	1982

Table 2
Costs of the solutions of FORT instances obtained by SA with different configurations of $(\tau \times i^*)$. Columns labeled *Best* report the best known solutions and rows *%imp over* R_1^1 and *%imp over 48A* show the percentage gap between SA and R_1^1 and 48A, respectively.

	$N = 100$			$N = 200$			$N = 400$		
	25×2500	125×2500	Best	25×2500	125×2500	Best	$5 \times 12{,}500$	$25 \times 12{,}500$	Best
Instance 10	75.26	73.77	73.45	120.83	121.37	120.83	200.75	200.24	200.25
Instance 11	65.97	65.97	65.97	108.36	108.95	108.36	197.67	197.67	197.67
Instance 12	68.65	68.47	68.29	113.51	110.56	110.20	199.86	194.97	194.97
Instance 13	71.12	71.12	70.84	111.70	108.03	108.03	194.05	190.20	190.20
Instance 14	81.29	80.91	80.49	121.55	118.73	118.73	192.80	192.80	192.80
Instance 15	75.31	73.59	73.30	113.79	114.27	113.79	202.93	202.93	202.93
Instance 16	72.63	73.41	72.63	110.61	113.66	110.61	197.49	194.11	194.11
Instance 17	62.58	62.60	62.54	106.05	105.49	105.49	199.22	199.22	199.22
Instance 18	72.36	72.18	72.15	112.64	110.74	110.74	196.58	196.58	196.58
Instance 19	69.75	69.55	66.60	114.79	115.13	112.88	201.28	201.28	201.28
%imp over R_1^1	1.14	1.59		3.54	4.18		5.10	5.78	
%imp over 48A	−0.77	−0.33		0.56	1.17		2.82	3.49	
Instance 20	51.32	51.32	51.32	98.04	97.49	97.49	199.73	194.90	194.90
Instance 21	55.45	55.45	55.45	102.93	102.93	102.93	204.80	204.80	204.80
Instance 22	62.87	62.83	62.83	106.64	106.08	105.34	217.12	214.96	194.58
Instance 23	62.96	62.30	55.41	114.04	113.38	112.18	217.38	217.38	201.13
Instance 24	66.06	66.06	61.99	106.77	106.34	106.34	203.02	202.00	202.00
Instance 25	55.59	55.59	55.59	111.78	111.53	109.00	207.85	207.85	207.85
Instance 26	62.56	62.56	56.65	114.83	113.26	108.23	224.78	220.22	194.01
Instance 27	57.04	57.04	57.04	106.27	105.37	105.37	209.82	209.82	202.38
Instance 28	56.69	56.93	56.67	106.22	106.55	106.22	197.76	197.76	197.76
Instance 29	63.71	62.99	59.01	115.83	114.60	108.96	208.79	206.38	203.21
%imp over R_1^1	1.97	2.16		3.35	3.89		−0.39	0.33	
%imp over 48A	−0.66	−0.47		0.60	1.14		−2.81	−2.11	

Table 3 shows the running time: that of algorithms R_1^1 and 48*A* increases much faster with the instance size than that of SA, since the total number of iterations performed by the first two algorithms is not bounded, as it is for the Simulated Annealing. For example, R_1^1 is much faster than SA (with a total of 62500 iterations) for instances with 100 and 200 customers, but the time is similar for 400-customers instances.

R_1^1 and 48*A* take more time for A instances than for B instances, while the opposite holds for SA. This can be explained intuitively: since A instances have a centrally located depot and 9 recharge stations, while in B instances the depot is at a corner and there are only 5 stations, the number of feasible routes is likely to be larger in the former ones. This forces local search algorithms to explore larger neighborhoods. SA, that chooses tentative relocations at random, does not suffer because of the number of feasible solutions in A instances, but rather for the difficulty of finding feasible moves in B instances.

4.3. Local search and SA on SSG instances

We present some additional results obtained by 48A and SA on a set of 56 instances with 100 customers proposed by Schneider et al. (2014) for a similar problem. They were adapted to the GVRP-MTPR by removing all time windows but the one of the depot, considering a single technology and permitting partial recharges.

According to the geographical distribution of customer locations, these instances are divided into three classes: random (R), clustered (C) and random-clustered (RC). According to the maximum duration of the routes, they are classified into two classes: long (L) and short (S) driving time limit. Within each of the six resulting sets, the original instances differ for the time windows and sometimes also for other characteristics such as the capacity of the battery and a few customer locations. C-S

Table 3
Average computational time (seconds) per instance for different instance sets (rows) using algorithms R_1^1, 48A and SA with a total of 62500 and 312,500 iterations (columns).

				SA	SA
		R_1^1	48A	62,500 iterations	312,500 iterations
Configuration A	$N = 100$	3	65	30	156
	$N = 200$	17	436	61	352
	$N = 400$	119	2607	119	777
Configuration B	$N = 100$	2	50	35	268
	$N = 200$	11	257	68	521
	$N = 400$	63	1177	129	1126

and RC-S instances are identical within each class. The instances in sets C-L and R-S differ for the battery capacity and some customer locations. R-L and RC-L instances only differ for battery capacities. For this reason, these last two sets are especially useful to evaluate the impact of increasing the battery capacity.

Table 4 shows the average results on SSG instances of each class (combinations of R, C, RC classes, in the columns, with S, L classes, in the rows). For each 3-column group, the columns with headings 48A and SA display the average costs of the solutions obtained by these algorithms, and the column labeled *gap* shows the average relative percentage gap between 48A and SA. Please note that a positive gap indicates that 48A outperforms SA. Finally, the average gaps for each geographical configuration are given in the last row, labeled *avg*, and for each shift configuration in the last column, labeled *avg gap*.

The results reported in Table 4 indicate that instances with short shifts, that usually require more vehicles (as also observed in Schneider et al. (2014)), show larger gaps between 48A and SA. The gap between the two algorithms is more significant for R instances and almost negligible for RC instances. Algorithm 48A outperforms SA for all sets. This was expected because these instances have only 100 customers.

Fig. 2 illustrates how the solution cost changes with the battery capacity in R-L and RC-L instances. As expected the trend is decreasing, but not monotonically because the plotted solutions are heuristic and thus the solution found for a given battery capacity could be worse than the one found for a lower battery capacity. For this reason we also plotted the minimum cost among the solutions found by both heuristics on instances with a smaller or equal battery capacity (line MIN), because all these values are also valid solutions for each instance. The MIN line is now monotonically decreasing in both figures. This test allows us to conclude that increased autonomy leads to significant cost savings, allowing to perform less and more convenient recharges.

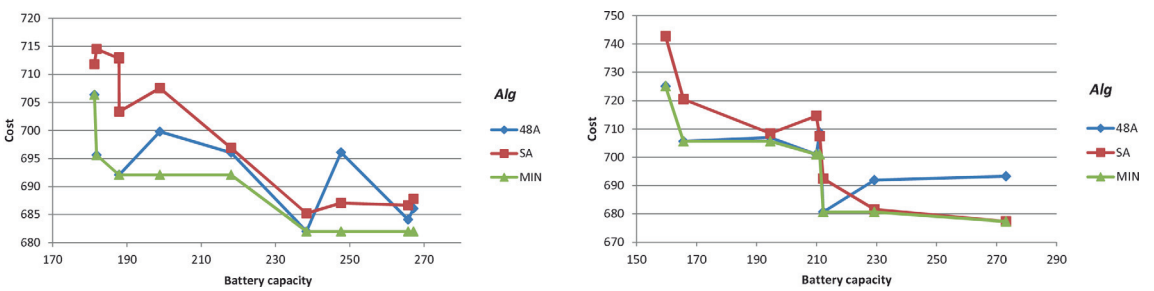
4.4. Results on a particular case: the Green VRP

Since this is the first paper dealing with the GVRP-MTPR and we derived this model as an extension of the one presented by Erdogan and Miller-Hooks (2012), we performed some additional tests on their GVRP instances (in what follows, E&MH instances) in order to further evaluate the performance of our algorithms. Please note that these algorithms have been designed for the generalized problem with multiple technologies and partial recharges and thus, if only full recharges with a single technology are allowed, some of the elements of the heuristics are not used. As a consequence, their efficiency could be improved by dropping those unnecessary elements. However, we believe that the results obtained by more general algorithms on a particular case are interesting because, considering the lack of a benchmark to compare with, they could be a good indicator of their general performance.

E&MH instances were also used by Schneider et al. (2014) to evaluate the performance of VNS-TS on a particular case. VNS-TS, designed to solve an extension of the GVRP with time windows, combines Variable Neighborhood Search (Hansen and Mladenovic, 2001) and Tabu Search (Glover and Laguna, 1997). Then, the results obtained on the E&MH

Table 4
Average results of 48A and SA, including costs for each algorithm and the gap between them, for each SSG instance class: short (S) or long (L) driving shift (in the rows) and clustered (C), random (R) or random-clustered (RC) customer distributions (in the columns).

	C			R			RC			Avg gap
	48A	SA	Gap	48A	SA	Gap	48A	SA	Gap	
S	880.12	888.08	0.90%	964.65	1085.41	11.03%	1136.60	1140.02	0.30%	4.08%
L	666.67	687.80	3.06%	692.94	700.61	1.08%	701.64	705.61	0.53%	1.55%
avg			1.98%			6.05%			0.41%	



(a) R-L instances (b) RC-L instances

Fig. 2. Cost vs. Battery capacity.

instances with our algorithms are compared both with the algorithms from Erdogan and Miller-Hooks (2012) and the VNS-TS from Schneider et al. (2014).

In several E&MH instances a few customers cannot be served due to maximum duration or autonomy constraints (with at most one visit to a recharge station). In those cases, as done in the literature in previous studies, unreachable customers were removed and the instances were solved considering only feasible customers. For further details regarding the data of the instances considered we refer the reader to Erdogan and Miller-Hooks (2012).

Columns of Table 5 under the heading *DBCA* show the results provided by the Density-Based Clustering Algorithm (DBCA) proposed by Erdogan and Miller-Hooks (2012), which, according to the authors, required a computational time of the order of seconds in a machine comparable to ours. The columns under the heading *CPLEX* show the results obtained with an improved version of the model implemented by Schneider et al. (2014) and solved with CPLEX with a 3-hour time limit. The cases in which a feasible solution could not be found within the time limit are tagged with a — sign. We report the cost of each solution (labeled *cost*) and the number of vehicles needed (labeled v).

Columns of Table 5 under the heading *VNS-TS* and *48A* provide the results obtained with algorithm VNS-TS of Schneider et al. (2014) and with 48A starting from 500 initial solutions, respectively. For each case, columns labeled *cost* show the cost of the solution provided for each instance, columns labeled v show the number of vehicles and columns labeled *time* provide the running time in seconds. The VNS-TS algorithm of Schneider et al. (2014) was multi-started for ten times on each instance, while the results with 48A were obtained in a single run.

Table 5
Costs, number of vehicles (v) and running time on small E&MH instances using different solution methods: DBCA, CPLEX, VNS-TS and 48A; average percentage gap with respect to 48A and average running times are given in the last two rows.

Instance	DBCA		CPLEX		VNS-TS			48A		
	Cost	v	Cost	v	Cost	v	Time	Cost	v	Time
20c3sU1	1797.51	6	1797.49	6	1797.49	6	41.40	1805.41	6	1.64
20c3sU2	1613.53	6	1574.77	6	1574.77	6	38.40	1574.78	6	1.48
20c3sU3	1964.57	7	1704.48	6	1704.48	6	38.40	1704.48	6	1.47
20c3sU4	1487.15	6	1482.00	5	1482.00	5	39.00	1482.00	5	1.53
20c3sU5	1752.73	5	1689.37	6	1689.37	6	40.20	1689.37	6	2.15
20c3sU6	1668.16	6	1618.65	6	1618.65	6	40.20	1618.65	6	1.90
20c3sU7	1730.45	6	1713.66	6	1713.66	6	38.40	1713.67	6	2.09
20c3sU8	1718.67	6	1706.50	6	1706.50	6	40.20	1722.78	6	1.76
20c3sU9	1714.43	6	1708.81	6	1708.81	6	39.60	1708.82	6	2.14
20c3sU10	1309.52	5	1181.31	4	1181.31	4	38.40	1181.31	4	1.44
20c3sC1	1300.62	5	1173.57	4	1173.57	4	37.20	1178.97	4	1.50
20c3sC2	1553.53	5	1539.97	5	1539.97	5	34.80	1539.97	5	1.30
20c3sC3	1083.12	4	880.20	3	880.20	3	15.00	880.20	3	0.58
20c3sC4	1091.78	5	1059.35	4	1059.35	4	31.80	1059.35	4	1.11
20c3sC5	2190.68	7	—	7	2156.01	7	36.00	2156.01	7	1.42
20c3sC6	2883.71	9	2758.17	8	2758.17	8	42.60	2758.17	8	1.17
20c3sC7	1701.40	5	1393.99	4	1393.99	4	10.80	1393.99	4	0.28
20c3sC8	3319.74	10	3139.72	9	3139.72	9	37.20	3139.72	9	1.28
20c3sC9	1811.05	6	1799.94	6	1799.94	6	36.00	1799.94	6	1.20
20c3sC10	2644.11	8	—	8	2583.42	8	27.00	2583.42	8	1.11
S1_2i6s	1614.15	6	1578.12	6	1578.12	6	42.60	1578.12	6	1.53
S1_4i6s	1541.46	5	1413.96	5	1397.27	5	45.00	1413.97	5	1.64
S1_6i6s	1616.20	6	1560.49	5	1560.49	5	43.80	1571.30	6	1.73
S1_8i6s	1882.54	6	1692.32	6	1692.32	6	44.40	1692.33	6	1.73
S1_10i6s	1309.52	5	1173.48	4	1173.48	4	42.60	1173.48	4	1.50
S2_2i6s	1645.80	6	1633.10	6	1633.10	6	45.00	1645.80	6	1.64
S2_4i6s	1505.06	6	1555.20	5	1532.96	5	52.80	1505.07	6	1.34
S2_6i6s	3115.10	10	—	7	2431.33	7	46.80	2660.49	8	2.42
S2_8i6s	2722.55	9	2158.35	7	2158.35	7	34.20	2175.66	7	1.15
S2_10i6s	1995.62	6	—	6	1958.46	6	36.60	1585.46	5	1.11
S1_4i2s	1582.20	6	1582.21	6	1582.21	6	37.80	1598.91	6	1.56
S1_4i4s	1580.52	6	1460.09	5	1460.09	5	40.80	1483.19	5	1.58
S1_4i6s	1541.46	5	1397.27	5	1397.27	5	45.00	1413.97	5	1.64
S1_4i8s	1561.29	6	1403.57	6	1397.27	6	49.20	1397.27	6	1.62
S1_4i10s	1529.73	5	1397.27	5	1396.02	5	51.00	1396.02	5	1.67
S2_4i2s	1117.32	5	1059.35	4	1059.35	4	30.60	1059.35	4	1.12
S2_4i4s	1522.72	6	1446.08	5	1446.08	5	36.00	1446.08	5	1.31
S2_4i6s	1730.47	6	1434.14	5	1434.14	5	41.40	1434.14	5	1.37
S2_4i8s	1786.21	6	1434.14	5	1434.14	5	45.00	1434.14	5	1.39
S2_4i10s	1729.51	6	1434.13	5	1434.13	5	46.80	1434.13	5	1.40
Avg. gap 48A	−7.08%		0.12%		0.43%					
Avg. time							39.00			1.47

Finally, the average percentage gap of DBCA, CPLEX and VNS-TS with respect to 48A and the average running time are displayed in the last two rows. The gap is calculated as 100(cost48A − othercost)/(othercost), and thus negative values indicate improvements provided by 48A.

Table 5 shows that 48A clearly outperforms DBCA, obtaining better solutions for most instances and providing an average saving around 7% within one or two seconds of running time. The average gap with respect to CPLEX is nearly zero, obtaining the same solutions in most of the instances and improving 3 of them, and a very similar performance is observed with respect to VNS-TS, with a slightly higher 0.43% average gap and improving only on one instance. However, it should also be noted that 48A requires around 1–2 s to solve each instance, which is significantly faster than both CPLEX, with a 3-hour time limit, and VNS-TS, that requires 30–50 s for each instance. Besides, the difference in computer time requirements is even larger taking into account that VNS-TS results are the best of 10 runs, while 48A is run only once.

The results given in Table 5 show that 48A is competitive with the state of the art solution methods. It achieves significant improvements over DBCA with a similar running time and provides quite similar results than VNS-TS within significantly shorter running time. The results obtained by SA are not reported here because they do not provide a significant contribution over the other methods in these instances. They require significantly larger computational times while not being able to provide significant improvements over Algorithm 48A, due to the fact that the instances considered are relatively small and most of the obtained solutions are likely to be close to the optimum. However, as it will be shown next, when considering larger instances SA becomes better and is actually able to outperform 48A.

Table 6 shows the results on large E&MH instances, which contain between 111 and 500 customers. The two columns under the heading *E&MH* give the cost of the best solution found with MCWS and DBCA heuristics and the number of vehicles required, while the columns under the headings *VNS-TS*, *48A* and *SA* provide the same information but using VNS-TS (Schneider et al., 2014), 48A and SA and including also the running times (in minutes). Average percentage gaps with respect to 48A and SA and running times are provided in the last three rows of each group. Again, negative gaps represent improvements achieved by 48A or SA.

It is shown that 48A and SA outperform the heuristics proposed by Erdogan and Miller-Hooks, improving all solutions, with average savings between 7% and 11%. However, VNS-TS outperforms 48A and SA, with gaps between 2% and 6%. The parameters of Algorithms 48A and SA have been chosen so that the running times are comparable with the ones of VNS-TS, but again we must take into account that VNS-TS is taking the best results over 10 runs, while 48A and SA are performing only one.

On the instances with only 111 customers (top part of the table), 48A outperforms SA, with average costs around 2% lower. However, on the instances with 200 customers or more (bottom part of the table), the trend changes and SA outperforms 48A, with average costs more than 0.5% lower within similar running times. This highlights that 48A is faster and outperforms SA for small and medium instances, but the running time required grows faster with the instance size for 48A than for SA, and thus the latter is recommendable for large instances.

The results presented in this section show that, even though the proposed algorithms are designed for a version of the problem with multiple technologies and partial recharges, they are still competitive when applied to a particular case such as the GVRP.

Table 6
Costs, number of vehicles (v) and running time on large E&MH instances using different solution methods: best of MCWS and DBCA (labeled E&MH), VNS-TS, 48A and SA; average percentage gap of E&MH and VNS-TS with respect to 48A and SA and average running times are given in the last three rows of each group.

Instance	E&MH		VNS-TS			48A			SA		
	Cost	*v*	Cost	*v*	Time	Cost	v	Time	Cost	*v*	Time
111c_21	5626.64	20	4797.15	17	21.76	4960.60	18	21.74	5062.06	18	12.35
111c_22	5610.57	20	4802.16	17	23.56	4914.20	18	21.23	5029.17	18	12.30
111c_24	5412.48	20	4786.96	17	21.90	4952.90	18	21.27	5010.59	18	12.13
111c_26	5408.38	20	4778.62	17	25.12	4934.11	18	21.25	5092.06	18	12.07
111c_28	5331.93	20	4799.15	17	24.17	4971.93	18	21.29	5038.84	18	11.99
Avg. gap 48A	−9.65%		3.21%								
Avg. gap SA	−7.83%		5.29%								
Avg. time					23.30			21.36			12.17
200c	10413.59	35	8963.46	35	76.65	9276.63	32	76.41	9206.28	32	73.84
250c	11886.61	41	10800.18	39	120.90	11007.98	39	120.67	10885.71	38	108.10
300c	14229.92	49	12594.77	46	182.23	12869.17	46	184.27	12827.35	45	302.04
350c	16460.30	57	14323.02	51	232.03	14954.83	54	227.30	14828.63	52	332.38
400c	19099.04	67	16850.21	61	305.12	17351.92	61	302.03	17327.27	60	384.68
450c	21854.17	75	18521.23	68	525.52	19215.38	68	514.68	19085.91	67	456.26
500c	24517.08	84	21170.90	76	356.01	21636.59	76	352.16	21475.71	75	154.45
Avg. gap 48A	−10.00%		2.99%								
Avg. gap SA	−10.59%		2.31%								
Avg. time					256.92			253.93			258.82

5. Analysis of the main features of the GVRP-MTPR

The aim of this section is no longer to validate the algorithms but rather to exploit them to explore some distinctive features of the model we have introduced. In particular we analyze the effect of multiple recharge technologies in SubSection 5.1, the effect of recharge stations distribution in SubSection 5.2 and the usefulness of partial recharges in SubSection 5.3.

5.1. Multiple recharge technologies

This subsection is focused on analyzing the results obtained by making available different combinations of recharge technologies, with the aim of illustrating their effect on the final recharge cost. For this purpose, we solved all the original FORT instances with different recharge technologies. First of all, in order to obtain more illustrative results, to the two original technologies, *Medium (M)* (cheap) and *Fast (F)* (slightly expensive), we have added an additional third technology, *Ultra Fast (U)* (fastest but most expensive). Then, based on the FORT instances, we have created 5 instance sets with different recharge technologies available (apart from the Slow recharge that is always available at the depot for the initial recharge): three sets in which only one technology is available (M, F or U), one set in which all recharge stations are equipped with the two technologies M and F (labeled MF) and another set in which all three technologies are available at all stations (labeled MFU).

Table 7 shows a summary of the average results obtained for each configuration and size instance set. The heading of each pair of columns displays the two technology combinations compared in each case, and the two corresponding columns show the average percentage cost difference between the two combinations and the number of instances in which one combination from each pair performed better than the other one. For example, in heading F/M, the first column indicates the average percentage cost difference between the solutions provided using technology F and the ones using technology M. A positive percentage means that the first technology considered, in this case F, performs better in average than the other one, in this case M. However, in general, a positive average difference does not necessarily mean that one is better than the other for all instances. Besides average values, we provide the number of instances in which each combination performed better. In heading F/M, the first cell of the second column, 7/3, indicates that technology F provided better solutions in 7 of the A instances with size 100, while technology M was better in 3. This column is omitted for the last two comparisons, MF/M and MFU/M, because MF and MFU instances extend M ones and thus M solutions can never improve over MF and MFU ones.

The columns of Table 7 with headings F/M, U/M and U/F show the differences between each technology when considered independently. One of the main insights extracted from this is that, in most cases, none of the technologies dominates the others (note that 10/0 only appears in 3 out of 18 cells in the table). F and U technologies seem better, in average, than technology M, but still there are quite a few instances in which M would be preferred. Furthermore, neither F nor U dominates each other, since the first one is better in 26 cases and the second in 34. This highlights the importance of being able to consider jointly several technologies, in order to choose the most appropriate one in each case. This is also supported by the results obtained for MF/M and MFU/M, which show that significant cost savings can be achieved if two or three technologies are available simultaneously.

Table 7 also shows that the differences between the two fastest technologies and the cheapest one are larger for configuration B than for configuration A. This could be due to the fact that B instances have fewer stations and the depot is at a corner, complicating the visits to distant customers and thus increasing the need of fast recharges. The effect of these elements will be analyzed in the next section.

5.2. Depot location and number of stations

We present some additional experiments performed on four instance sets generated by varying the position of the depot or the number of stations, in order to evaluate the impact of these characteristics. Two sets have nine recharge stations, located as displayed in Fig. 1(a), and two sets have five stations, located as displayed in Fig. 1(b). For each station configu-

Table 7
Average percentage cost savings for different recharge technology combinations (first column of each group) and number of instances in which one combination from each pair improves on the other one (second column of each group).

		Technologies compared: Medium (M), Fast (F), Ultra (U)							
		F/M		U/M		U/F		MF/M	MFU/M
Conf. A	$N = 100$	−0.13%	7/ 3	0.18%	6/ 4	0.31%	5/ 5	0.30%	0.44%
	$N = 200$	0.29%	6/ 4	0.05%	6/ 4	−0.25%	4/ 6	0.76%	1.08%
	$N = 400$	0.41%	7/ 3	0.64%	7/ 3	0.24%	7/ 3	0.75%	1.01%
Conf. B	$N = 100$	0.79%	9/ 1	1.23%	7/ 3	0.44%	4/ 6	0.82%	1.49%
	$N = 200$	1.08%	10/ 0	1.23%	10/ 0	0.15%	6/ 4	1.08%	1.32%
	$N = 400$	1.43%	9/ 1	1.98%	10/ 0	0.56%	8/ 2	1.49%	2.03%

ration, in one set the depot is located in the center and in the other one it is located in a corner. These instances are referred to as FORT'.

Table 8 shows the average running times (in seconds) needed by 48A and SA for each combination of depot location and number of stations (rows) and for different instance sizes (columns), together with the averages over all instances (columns labeled *avg*). We can observe clearly how, for a fixed number of stations, the running times are significantly higher for 48A when the depot is at the center than when it is in a corner (nearly twice as high, in average, either with 5 or 9 stations), while for SA it is higher when the depot is at a corner (around one third higher, in average, also for 5 or 9 stations). However, if the location of the depot is fixed, changing the number of stations does not affect the running times of the algorithms, which are quite similar. These results allow to conclude that the location of the depot can have a significant impact on the running times of algorithms, but this is not the case for the number of recharge stations.

Table 9 shows the average percentage cost savings obtained by using technologies MF (the two cheapest technologies) or MFU (all three technologies) instead of only technology M (the cheapest one) for each configuration. Each row corresponds to one combination of depot location and number of stations and each column to a certain problem size (100, 200 or 400 customers). One additional column, labeled *avg*, provides the averages over all instance sizes.

First, if we fix the number of stations, we can observe how the additional savings achieved by using several technologies are larger when the depot is in the center than when it is in a corner. Similarly, if the location of the depot is fixed, Table 9 shows that when reducing the number of stations from 9 to 5 the average gain of using several technologies increases. This leads to the conclusion that both the location of the depot and the number of stations can have an impact on the need for faster recharges, and apparently this impact is higher for the former than for the latter.

5.3. Comparison between partial and full recharges

One of the novelties of the problem approached in this paper is to allow the vehicles to leave the stations before their batteries are fully recharged, obtaining more time to complete the routes within the driving time limit. This section is focused on analyzing the impact of this extension in the final cost, in comparison with having the obligation to perform full recharges, as it was the case in previous studies (see, for example, Erdogan and Miller-Hooks (2012) and Schneider et al. (2014)). For this purpose, we have solved the FORT' instances both with partial and full recharges and compared the results obtained.

The first important result is that several of the FORT' instances become infeasible if only full recharges are allowed. This happens because there are some customers which are so far away from the depot that the vehicles need to recharge in their way back to it. If partial recharges are available it is possible to visit each customer and come back within the driving time limit, recharging only the energy needed, but if only full recharges can be performed, in some cases the total duration of the route exceeds the driving time limit. This is especially relevant for the instances in which the depot is located at a corner, for which many of the instances are infeasible with full recharges, even using the fastest technologies.

In the case of the instances with a centrally located depot we found that some instances were also infeasible with full recharges if technology M is the only one available at the stations, because it is still too slow in some cases to reach some customers, but all instances were feasible for all the other technology combinations. Hence, to allow for a fair comparison,

Table 8
Average running times (in seconds) using 48A and SA for different problem configurations (in the rows) and problem sizes (in the columns). Averages over all instance sizes are also given in columns *avg*.

Configuration		Running time of 48A				Running time of SA			
Stations	Depot	$N = 100$	$N = 200$	$N = 400$	Avg	$N = 100$	$N = 200$	$N = 400$	Avg
9	Center	67	434	2615	1039	274	533	1181	662
	Corner	52	264	1449	588	354	707	1531	864
5	Center	66	445	2673	1062	268	522	1101	630
	Corner	53	268	1438	586	349	710	1489	849

Table 9
Average percentage cost saving obtained by using technologies MF or MFU instead of technology M for different problem configurations (rows) and problem sizes (columns). Averages over all instance sizes are also given in column *avg*.

Configuration		% improvement of MF over M				% improvement of MFU over M			
Stations	Depot	$N = 100$	$N = 200$	$N = 400$	Avg	$N = 100$	$N = 200$	$N = 400$	Avg
9	Center	0.29	0.97	0.62	0.63	0.32	1.32	1.13	0.92
	Corner	0.94	1.10	1.08	1.04	1.13	1.77	1.91	1.60
5	Center	1.32	0.87	0.66	0.95	1.43	1.13	0.94	1.17
	Corner	0.98	1.19	1.52	1.23	1.33	1.70	2.56	1.87

Table 10
Average percentage cost saving obtained by using partial recharges in comparison with full recharges for different instance configurations (columns) and sizes (rows). Averages are given in the last row and last column.

	Centered depot, 9 stations				Centered depot, 5 stations				Avg.
	F	*U*	MF	MFU	*F*	*U*	MF	MFU	
$N = 100$	1.39	1.29	0.83	0.86	3.00	1.42	0.17	0.20	1.15
$N = 200$	2.48	1.57	0.42	0.66	4.13	1.95	0.34	0.53	1.51
$N = 400$	3.46	1.65	0.47	0.53	6.11	1.16	0.12	0.12	1.70
Avg.	2.44	1.50	0.58	0.68	4.41	1.51	0.21	0.28	

we decided to show the results of the configurations in which all instances are feasible using both partial and full recharges. As a result, Table 10 displays the average percentage cost savings achieved by using partial recharges in comparison with full recharges for instance sets with 9 or 5 stations and F, U, MF and MFU technology combinations (in the columns) and with 100, 200 and 400 customers (in the rows). Besides, average results for each instance size are given in the last column and for each instance configuration in the last row.

Table 10 shows how the impact of partial recharges is higher when only one of the recharge technologies is available, and in particular when the only available technology is F. However, as expected, if there are multiple recharge technologies (MF and MFU), the flexibility to perform recharges is higher and thus the potential savings achieved by partial recharges are smaller. In average, the savings increase with the instance size (see last column). Hence the possibility to perform partial recharges can lead to significant improvements on the solution cost, that may vary depending on the instance configuration and size. Furthermore, it could allow reaching some distant customers that could not be visited if only full recharges were possible, which sometimes might be even more important than potential costs savings because it affects feasibility.

6. Conclusions and future research

In this paper we have approached a version of the vehicle routing problem in which the transportation fleet is composed of electric vehicles with limited autonomy. This problem extends the recently introduced GVRP by considering the possibility of performing partial recharges and using several recharge technologies, which have a wide potential to produce additional cost and energy savings. A mathematical programming model was presented, but since this model can only be solved to optimality for very small instances, several heuristic algorithms were also proposed with the aim of solving real size instances. An extensive computational experience has been performed, focused on evaluating the performance of the proposed algorithms and analyzing the main elements of the problem approached.

For this purpose, we first created a benchmark for the GVRP-MTPR, including instances with different geographical configurations for customer locations and recharge stations, and solved them with our heuristics. We concluded that Algorithm 48A, based on exhaustive local search, is best when dealing with medium sized instances with up to 200 customers; however, the algorithm based on Simulated Annealing, SA, performs better for larger instances, because its computational complexity grows more slowly with the instance size. Besides, we also concluded that SA is faster when applied to instances with many feasible solutions, while 48A is faster on more constrained instances.

Since a GVRP-MTPR benchmark is not available in the literature, we also solved the GVRP instances of Erdogan and Miller-Hooks (2012), showing that our heuristics are competitive with the state of the art. We also solved the E-VRPTW instances of Schneider et al. (2014) after adapting them to the GVRP-MTPR by dropping the time windows. These experiments allowed us to confirm that important cost savings could be achieved if the vehicles had a higher autonomy.

Furthermore, some additional tests on instances with different recharge technology combinations and allowing partial or full recharges illustrated the benefit of having multiple technologies available and allowing partial recharges. First, none of the technologies considered individually performed better than any other in all cases, highlighting the importance of having several technologies available to choose the most appropriate one for each recharge. Second, partial recharges were shown to provide significant cost and energy savings, and also to be helpful to ensure feasibility in some complicated instances.

Well known additional elements of standard vehicle routing problems, such as considering several depots, time windows, heterogeneous fleets, etc. could be added to the GVRP-MTPR, as well as some other extensions related to the specific characteristics of electric vehicles, as suggested by Lin et al. (2014). The design of efficient exact approaches, though limited to solving very small sized instances, is another possible line of future research, together with the consideration of other classical metaheuristics.

Acknowledgments

This work was partially supported by projects TIN2012–32482 and MTM2012-33740 from the Government of Spain and by the programme "Visitantes Distinguidos 2011–12" of Universidad Complutense de Madrid. We would also like to thank Elise Miller-Hooks and Sevgi Erdogan for letting us use their GVRP instances and for the clarifications about their interpretation, Simona Mancini for the fruitful discussions about the problem and 3 anonymous referees for their valuable suggestions that helped us to improve the paper significantly.

References

Aarts, E.H., Lenstra, J.K., 2003. Local Search in Combinatorial Optimization. Princeton University Press.

Barco J, Guerra A, Muñoz L, Quijano N, 2013. Optimal Routing and Scheduling of Charge for Electric Vehicles: Case Study. arXiv preprint arXiv:1310.0145.

Cavadas, J., Correia, G., Gouveia, J., 2014. Electric vehicles charging network planning. In: Computer-based Modelling and Optimization in Transportation. Springer International Publishing, pp. 85–100.

Chen, T.D., Kockelman, K.M., Khan, M., 2013. Electric vehicle charging station location problem: A parking-based assignment method for Seattle. In: Transportation Research Board 92nd Annual Meeting (No. 13-1254).

Conrad, R.G., Figliozzi, M.A., 2011. The recharging vehicle routing problem. In: Doolen, T., Van Aken, E. (Eds.), Proceedings of the 2011 Industrial Engineering Research Conference.

Energy Lab. 2011. Sviluppare la mobilità elettrica - Tecnologie, ambiente, infrastrutture, mercato e regole. GIE edizioni, Roma. ISBN 978-88-97342-07-6.

Erdogan, S., Miller-Hooks, E., 2012. A green vehicle routing problem. Transp. Res. Part E 48, 100–114.

Frade, I., Ribeiro, A., Gonçalves, G., Antunes, A.P., 2011. Optimal location of charging stations for electric vehicles in a neighborhood in Lisbon, Portugal. Transp. Res. Rec.: J. Transp. Res. Board 2252 (1), 91–98.

Glover, F., Laguna, M., 1997. Tabu Search. Kluwer Academic Publishers, Boston.

Golden, B.L., Raghavan, S., Wasil, E.A., 2008. The Vehicle Routing Problem: Latest Advances and New Challenges. Springer.

Hansen, P., Mladenovic, N., 2001. Variable neighborhood search: principles and applications. Eur. J. Oper. Res. 130, 449–467.

Kara, I., Kara, B., Yetis, M., 2007. Energy minimizing vehicle routing problem. Lect. Notes Comput. Sci. 4616, 62–71.

Kuo, Y., 2010. Using Simulated Annealing to minimize fuel consumption for the time-dependent vehicle routing problem. Comput. Ind. Eng. 59 (1), 157–165.

Küçükoolu, Y., Ene, S., Aksoy, A., Öztürk, N., 2013. A green capacitated vehicle routing problem with fuel consumption optimization model. Int. J. Comput. Eng. Res. 3 (7), 16–23.

Lin, C., Choy, K.L., Ho, G.T.S., Chung, S.H., Lam, H.Y., 2014. Survey of green vehicle routing problem: past and future trends. Expert Syst. Appl. 41 (4), 1118–1138.

Nie, Y.M., Ghamami, M.A., 2013. corridor-centric approach to planning electric vehicle charging infrastructure. Transp. Res. Part B: Methodol. 57, 172–190.

Paschero, M., Anniballi, L., Del Vescovo, G., Fabbri, G., Mascioli, F.M.F., 2013. Design and implementation of a fast recharge station for electric vehicles. In: IEEE International Symposium on Industrial Electronics 2013: 1-6. 978-1-4673-5194-2.

Pisinger, D., Ropke, S., 2007. A general heuristic for vehicle routing problems. Comput. Oper. Res. 34, 2403–2435.

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

Schneider, M., Stenger, A., Goeke, D., 2014. The Electric Vehicle Routing Problem with Time Windows and Recharging Stations. Transportation Science. Published online in Articles in Advance, 06 Mar 2014. <http://dx.doi.org/10.1287/trsc.2013.0490>.

Suman, B., Kumar, P., 2006. A survey of Simulated Annealing as a tool for single and multiobjective optimization. J. Oper. Res. Soc. 57, 1143–1160.

Syslo, M.M., Deo, N., Kowalik, J.S., 1983. Discrete Optimization Algorithms with Pascal Programs. Prentice-Hall.

Toth, P., Vigo, D., 2002. The vehicle routing problem. SIAM.

Touati-Moungla, V., Jost, V., 2010. Combinatorial optimization for electric vehicles management. In: International Conference on Renewable Energies and Power Quality (ICREPQ11), Las Palmas de Gran Canaria (Spain).

Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2012. Heuristics for Multi-Attribute Vehicle Routing Problems: A Survey and Synthesis. Tech. Rep., CIRRELT 2012-05.

Xiao, Y., Zhao, Q., Kaku, I., Xu, Y., 2012. Development of a fuel consumption optimization model for the capacitated vehicle routing problem. Comput. Oper. Res. 39 (7), 1419–1431.