A two-phase optimization method for a multiobjective vehicle relocation problem in electric carsharing systems

Maurizio Bruglieri[a,†], Ferdinando Pezzella[b], Ornella Pisacane[b]

[a]Dipartimento di Design, Politecnico di Milano, via Durando 38/a, 20158, Milano, Italy
[b] Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, Via Brecce Bianche 12, 60131 Ancona, Italy

Abstract

The paper focuses on one-way electric carsharing systems, where the fleet of cars is made up of Electric Vehicles (EVs) and the users can pick-up the EV at a station and return it to a different one. Such systems require efficient vehicle relocation for constantly balancing the availability of EVs among stations. In this work, the EVs are relocated by workers, and the issue of finding a trade-off among the customers' satisfaction, the workers' workload balance and the carsharing provider's objective is addressed. This leads to a three-objective optimization problem for which a two-phase solution approach is proposed. In the first phase, feasible routes and schedules for relocating EVs are generated by different randomized search heuristics; in the second phase, non-dominated solutions are found through epsilon-constraint programming. Computational results are performed on benchmark instances and new large size instances based on the city of Milan.

Keywords: one-way carsharing; vehicle operational repositioning; pick-up and delivery problem; operator-based relocation; epsilon-constraint programming; randomized search heuristics.

1. Introduction and literature review

Pay as you drive is nowadays a real possibility thanks to carsharing systems, where users can rent a car even for a short time (usually fractions of an hour) and pay for it according to the time of use. This favors the sharing of cars among different people, leading to a reduction in the number of parked cars, and by consequence in the traffic congestion and air pollution. Carsharing systems are made more flexible in the one-way modality where a vehicle can be picked-up from one station and returned to a different one. However, the one-way carsharing system suffers from possible imbalances between the demand and the availability of cars, leading to the need for vehicle relocation.

The aim of this paper is to address this issue by assuming that the relocation is performed by workers who drive the vehicles from a pick-up station to one of delivery, and then move, using folding bicycles, from a delivery station to one of pick-up. The workers' routes have to respect a maximum duration (i.e., a work shift), due to some forms of contract. According to the features of the user requests, the solution has also to satisfy time windows constraints. In fact, for each request, a time interval usually specifies within which the vehicle can be either picked-up or delivered. In the case of Electric Vehicle (EV) sharing systems, the battery level constraints are also imposed, i.e., an EV cannot be delivered to a station if its battery level does not allow reaching the station or if its residual battery level is lower than that required by the user.

† Corresponding author. Email: maurizio.bruglieri@polimi.it

Literature contributions show that an efficient strategic plan can provide the decision makers with a good compromise between the service level offered to the users and the total costs. However, only a few works directly address the vehicle relocation problem, optimizing more than one objective at the same time. Several existing literature contributions study the factors impacting on the system's performances (e.g., fleet size, relocation policies etc.); model strategic, operational and tactical decisions (Jorge and Correia, 2013; Brandstätter et al., 2016) and investigate also the relationship between the supply and the demand (e.g., Martinez et al., 2017).

In the following, we focus on the main contributions related to the Vehicle Relocation Problem arising in one-way carsharing systems. This problem belongs to the operational decisions and, in particular, it arises from possible unbalances between the demand and the availability of either vehicles at stations or parking lots. Vehicle relocation is traditionally classified (Barth and Todd, 1999) into: *static* where the immediate needs of parking lots is taken into account; *historical*, where the vehicles are relocated according to the estimation of future relocation requests performed on the basis of historical data and finally, *exact predictive*, usually performed in carsharing systems based on reservation since it uses a precise knowledge of the relocation requests. Moreover, the relocation can be performed by either the user who is driving it or by workers of the carsharing system itself (Barth et al., 2004). In the first case, the service provider can adopt pricing strategies to incentivize the users to return the vehicles to specific stations; in the second case, instead, the workers can perform vehicle relocation through either trucks or by driving them. Since the use of trucks may be unsuitable in an urban context (e.g. due to a negative impact on traffic congestion), the workers directly drive the vehicles to relocate them from stations with too many vehicles, requiring the removal of one of them (pick-up relocation requests) to stations needing vehicles (delivery relocation requests). In such a system, once the relocation requests have been estimated (e.g. by a simulator), the most significant problem is efficiently routing and scheduling the workers in order to handle as many relocation requests as possible.

The effectiveness of introducing one-way services in addition to the round-trip ones is investigated in Jorge and Correia (2015) where an Integer Linear Programming model is proposed for properly selecting a set of new one-way carsharing services to use in round-trips systems. The same topic is addressed by Nourinejad and Roorda (2015) through a simulation-optimization approach for both sizing the fleet and finding the optimal relocation hours in one-way and two-way systems.

While a general survey on the problem of efficiently relocating the vehicles is given by Shaheen and Cohen (2013), here we only focus on the contributions related to the operator-based relocations. Jorge et al. (2014), in order to maximize system profitability, compare a solution approach based on mathematical programming with the one defined on a simulation model, considering several real-time relocation policies. Nourinejad et al. (2015) solve the vehicle relocation jointly to the staff rebalance, using two integrated multi-traveling salesman formulations. Halffmann et al. (2015) introduce an Online Min-Wait Relocation Problem for minimizing the total waiting time. Santos and Correia (2015) formulate a Mixed Integer Programming model to minimize the total cost due to the staff employed in the relocation operations, the rejection of requests and the maintenance of requests not fulfilled. Martinez et al. (2017) consider both the relocation along a day and the reservations to evaluate their impact on the level of service offered to the users and the system's profit. Di Febbraro et al. (2012) minimize both the number of required workers and that of vehicles needed to satisfy demand, through a discrete-event simulation.

The use of EVs, as alternative to the one of traditional Internal Combustion Engine Vehicles (ICEVs), especially in urban contexts, is incentivizing the diffusion of electric carsharing systems. On one hand, such systems exploit the advantages of the EVs, which are less noisy than the ICEVs and do not emit harmful CO2 emissions locally (although the electric energy supplied to the EVs during the recharging process may be produced through an energy mix also with harmful emissions). On the other hand, the EVs pose new management problems, mainly due to their poor driving range, that complicate the vehicle relocation by the workers. Several literature contributions address the efficient management of EVs (Brandstätter et al., 2016). For example, Lee and Park (2013) describe a three-step methodology for relocating EVs in carsharing systems where the demands are estimated on the dataset provided by a taxi system. Then, the relocations are performed out of the operation hours and finally, a genetic algorithm is run to solve the staff scheduling problem, minimizing the travelled distance. An event-based simulation is applied in Repoux et al. (2014) to relocate EVs in a one-way system, experimenting several policies to maximize both the users' satisfaction and the sustainability perceived by the workers. Recently, Cao et al. (2016) propose a threshold

triggering method for relocation where the threshold is set solving an optimization model that minimizes the out-of-service rate and the number of moving EVs.

Under the hypothesis that the relocation is performed by workers who drive EVs from a pick-up point to one of delivery and move from a point of delivery to one of pick-up using folding bicycles, a single-objective **E**lectric-**V**ehicle **Re**location **P**roblem (E-VReP) has been already addressed in one-way carsharing systems (Bruglieri et al. (2014a, b)). In particular, in the first work, the authors propose a Mixed Integer Linear Programming (MILP) formulation for the EVReP, to maximize the number of requests served. In the second paper, the authors design a simulator that combines the real data yielded by the Agency for Mobility, Environment and Territory of the Municipality of Milan (AMAT), concerning the users' relocation requests, and the location of the recharging stations in the city, provided by A2A, the main energy supplier in Milan (www.a2a.eu).

To investigate the economic sustainability of this relocation approach, the problem has been extended in Bruglieri et al. (2017), associating a revenue with each relocation request satisfied and a cost with each operator used. The new variant of the E-VReP maximizes the total profit instead of the total number of relocation requests satisfied. After proving the NP-hardness and the APX-hardness of this problem, to overcome the drawback of the high CPU time required by the MILP formulation, a Ruin and Recreate (R&R) metaheuristic is designed. Numerical results show that the R&R finds solutions very close to the ones of MILP but in by far shorter computational times. In the same context, very recently, Bruglieri et al. (2018) propose an Adaptive Large Neighborhood Search (ALNS) that outperforms by far both the R&R and a Tabu Search ad hoc designed in the same paper, even on real like instances with on average 100 relocation requests.

Also Malaguti et al. (2017) consider an operator based relocation problem from the economical point-of-view of a service provider that wants to maximize the profit associated with the trips performed by users. They consider an EV sharing system on reservation and propose both a MILP model and two model-based heuristics to solve the relocation problem on real like large-scale instances (125 requests).

Despite a rich literature in carsharing systems optimization, only few contributions address multiobjective problems for efficiently managing specific issues. For example, Boyaci et al. (2015) propose a MILP model for supporting strategic and tactical planning decisions for car-sharing systems by considering simultaneously decisions associated with the allocation of strategic assets, i.e. stations and vehicles, and the allocation of personnel for relocation operations (tactical decision). Their model maximizes the net revenue of the service provider and the users' benefit expressed as the utility gain (due to each trip satisfied) minus the rental and the accessibility cost (from/to a station to/from a center). Assuming that the EVs are accumulated in a virtual hub before being distributed to the stations, the authors reduce the number of variables and adopt a weighted sum method to compute the efficient frontier (i.e., the Pareto front) for the service provider's revenue and the users' benefit. Experimental results, carried out on a real world electric carsharing system in Nice (France), highlight the trade-off found between the service provider's point of view and that of the users. The authors extended this work in Boyaci et al. (2017) proposing a simulation-optimization approach for the EV relocation problem in carsharing systems with reservation and presenting three mathematical programming models for clustering stations, optimizing operations and the personnel flow.

Table 1 compares the contributions in the literature concerning the operator based carsharing relocation, nearest to our work, with respect to the: 1. relocation type on the basis of the possible information on the user demand, i.e., Static (S), Historical (H) or Exact (E); 2. fleet characteristics i.e., either Internal Combustion Engine Vehicles (ICEV) or Electric Vehicles (EV); 3. parking modality, i.e. Free Floating (FF) or Station Based (SB); 4. presence of Time Windows associated with the relocation requests (Yes or No); 5. time representation: either Continuous (Cont) or Discrete (Discr); 6. kind of objective functions considered, i.e., the minimization of the number of Operators used (O), the maximization of the number (or profit) of relocation Requests satisfied (R), the minimization of the total Distance travelled by the operators (D), the minimization of the duration of the Longest route (LD), (we use the symbol "x" instead of "+" to emphasize that the objectives are not just reduced to only one but are really considered separately to compute the Pareto optimal front); 7. the solution approach adopted where MILP stands for Mixed Integer Linear Programming formulation, while Sim for simulation and Heur for heuristic; 8 the maximum instance size addressed in the experiments in terms of number of relocation requests considered.

Feature	Lee & Park (2013)	Boyaci et al. (2015)	Boyaci et al. (2017)	Nourinejad et al. (2015)	Gambella et al. (2017)	Bruglieri et al. (2014a,b)	Bruglieri et al (2017, 2018)	Our work
1. Static, Historical, Exact	H	E	E	E	E	H, E	H, E	H, E
2. Fleet (ICE, EV)	EV	EV	EV	ICE	EV	EV	EV	EV
3. Parking	SB	SB	SB	SB	SB	SB	SB	SB
4. Time Windows	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
5. Time representation	Discr	Discr	Discr	Discr	Discr	Cont	Cont	Cont
6. Objective function	D	(O+D) x R	(O+D) x R	O+D	R+D	R	O+R	OxR xLD
7. Solution approach	Heur	MILP	MILP, Sim	MILP	MILP, Heur	MILP, Heur	MILP, Heur	MILP, Heur
8. Maximum instance size	30	500	300	40	125	30	100	500

Table 1: Summary of comparison over the literature

At the best of our knowledge, no work in the literature models the service provider's objective, the total satisfaction of users' demand and the workers' workload balance in the electric vehicle relocation, at the same time. For this reason, in this paper, we introduce a three-objective E-VReP. In particular, the service provider's objective is taken into account by minimizing the number of workers used to relocate the EVs since it represents the main relocation operational cost. The satisfaction of users' demands is considered by maximizing the number of relocation requests served since this allows users to find the vehicles or deliver them where they want. Indeed, the vehicles are relocated according to the users' needs. Finally, the workers' workload balance is accounted by minimizing the duration of their longest route, since it guarantees at the same time the minimization of the total duration of the routes and the fairness among the workers, avoiding that some of them works longer than others. The main contribution of this paper is a method to approximate the Pareto optimal front based on a two-phase approach. In the first phase, several feasible routes serving different relocation requests are generated by randomized search heuristics; in the second phase, a three-objective non-overlapping model is formulated to select non dominated combinations of feasible routes and it is solved by epsilon-constraint programming (Chankong and Haimes, 1983). A very advantageous property of our solution method is that its computational complexity depends only linearly on the total number of the relocation requests and thus, it can also be used to solve large size instances.

Although the papers of Boyaci et al. (2015 and 2017) can look very close to ours, some significant differences exist. From the modelling perspective, they optimize two objectives, i.e., the total operational cost (to minimize) and the number of requests satisfied (to maximize). While, we also aim to a load balance among operators. Both the two aforementioned papers and ours deal with the trip accept/reject. In particular, maximizing the number of requests satisfied, we may not accept all of them. From the methodology point of view, they use a hierarchical approach for solving the bi-objective problem, assuming to know a priori the importance that a decision maker gives to each of them. Instead, we design a two-phase approach for approximating the Pareto front on large-scale instances while we detect the exact Pareto front, through a MILP-based approach, on some small and medium scale instances. Finally, they deal with a discrete time model while ours is a continuous time model.

Further contributions of our paper are:

- to solve the workers' routing and scheduling for relocating the vehicles and, at the same time, to decide which requests have to be satisfied and which are not;
- the development of two randomized search heuristics suitable for finding several feasible solutions with a small computational effort;
- the definition of a MILP-based method to compute the Pareto front in an exact way;
- the testing of the two-phase approach on real-world-like instances, also with large size (with about 500 requests), specifically generated for this work ;
- the evaluation of the solution distribution and diversification goodness through the *Spacing* and *Hypervolume* indicators, respectively;
- the comparison of the numerical results obtained with the two-phase approach, also with those detected by the MILP based method, when available.

The remainder of this paper is organized as follows: Section 2 introduces the notation and the three-objective formulation of the EVReP; Section 3 describes the heuristic solution approach; Section 4 presents numerical results carried out on two sets of real-world-like instances and finally, Section 5 draws some conclusions and derives useful insights.

2. A Three-objective Mixed Integer Linear Programming Formulation for the E-VReP

This section aims to introduce the notation followed in the paper (summarized in Table 2) and to describe the three objective functions of the multiobjective E-VReP. In the following, the statement of the problem is provided, describing separately the characteristics of the fleet, the relocation requests and the workers.

2.1. Fleet characteristics

A one-way carsharing system is assumed, made up of a homogeneous fleet of EVs, each one able to cover a maximum distance L with a fully recharged battery. The maximum covered distance is assumed to be linearly proportional to the residual charge of the battery. Moreover, the time, Γ, for fully recharging an exhausted battery is known. Although the behavior of the charging time function depends on the battery technology used (Marra et al. 2012), for the sake of simplicity it is assumed to be linear, as done also in several papers (e.g., Schneider et al, 2014).

2.2. Relocation request characteristics

The hypothesis used in Bruglieri et al. (2014a, b) and maintained in this paper is that both sets of delivery and pick-up requests are known *a priori.* This holds not only in carsharing systems on reservation but also in those using historical data to forecast the users' demand (Jorge and Correia, 2013). The delivery requests and those of pick-up are denoted by D and P, respectively. Each parking lot, where a user can pick-up/deliver an EV, is equipped with a recharge dock. In this way, an EV is assumed to be recharged during the time in which it is not required.

For each relocation request $r \in R = P \cup D$, the time window $[\tau_r^{min}, \tau_r^{max}]$ and the residual battery charge ρ_r (measured as a fraction of the total battery capacity) are specified. The values τ_r^{min} and τ_r^{max} represent the earliest and the latest time, respectively, at which r can be carried out. In particular, the former indicates the time before which the EV is not available; the latter, instead, the time after which it is not convenient to move the EV. If $r \in D$, the parameter ρ_r represents the minimum charge level that the EV has to have at τ_r^{max}. Consequently, an EV delivered before τ_r^{max} may also have a battery charge level less than ρ_r. Indeed, to satisfy the battery constraint, it is only necessary that the charge level ρ_r be reached within τ_r^{max}, considering that the EV is immediately recharged after delivery. If $r \in P$, the parameter ρ_r is the available battery charge level of the EV at the time τ_r^{min}.

2.3. Worker characteristics

A team of K workers is assumed to be available to perform the EV relocations. The workers are supposed to leave (even at different times) a common depot (denoted by *0*) and to reach, using a folding bicycle, the parking station associated with their first request of pick-up. They then load the folding bicycle into the trunk of the EV and reach a delivery request with a compatible time window and battery level, by driving the EV. They can continue to serve compatible pairs of pick-up and delivery requests until they go back to the depot within their shift T. In this way each worker can perform at most one route. The operational time, c_{ij}, required to satisfy each pair of requests i and j is assumed to be given. These times include both the traveling time from the location of request i to that of request j using the vehicle, the compatibility of the sequence of requests (i.e., the bicycle if i is a delivery request or the depot, the EV if i is a pick-up request) and the time taken to load/unload the bicycle into/from the trunk and to leave the parking station or to park.

Table 2: Nomenclature of the input data

Parameter	Meaning
P	set of pick-up requests
D	set of delivery requests
R	set of all the requests (i.e., $R = P \cup D$)
0	depot
K	number of available workers
$[\tau_r^{min}, \tau_r^{max}]$	time window of relocation request r
ρ_r	battery level demanded by request r (measured as a fraction of the total battery capacity)
L	distance covered by a fully charged EV
Γ	time to fully recharge an exhausted EV
T	work shift
s'	average speed of an EV
s''	average speed of a bike
q'	average time for parking an EV and removing the bike from the EV trunk
q''	average time for loading the bike in the EV trunk and leaving the parking lot with the EV
d_{ij}	minimum path length between i and j
c_{ij}	operational time for satisfying the pair of requests i and j

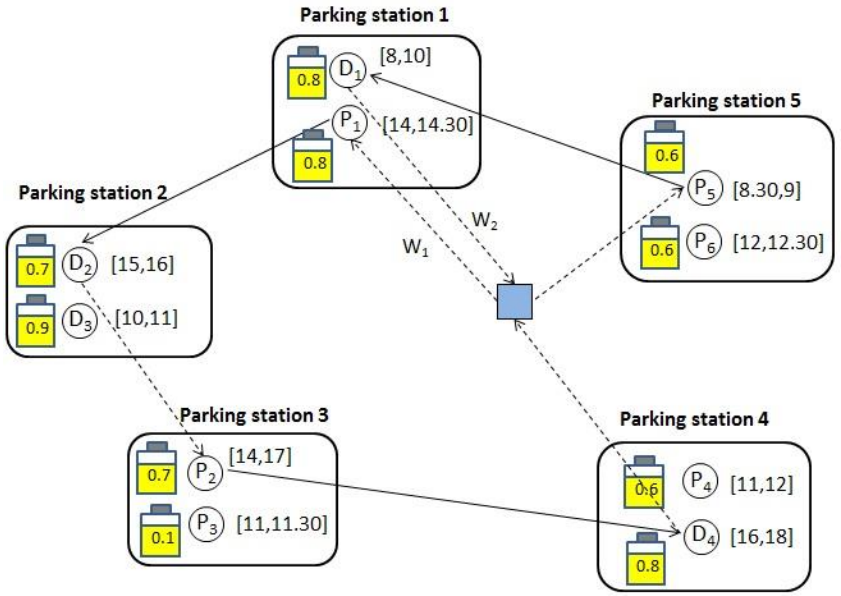

Figure 1: An instance of the E-VReP with six pick-up requests (P_i for i = 1,...,6) and four delivery requests (D_i for i = 1,...,4). For each request, the battery level is indicated on the left of the node, while the time window is shown on the right. Three workers, W_1, W_2, W_3 leaving a common depot (the square node) are available. A feasible solution with two workers is shown. The dashed arcs denote that a worker is moving by bicycle (the travel time of these arcs is 20 minutes) while the solid arcs indicate that he/she is driving an EV (their travel time is 12 minutes).

2.4. *Problem statement*

The aim of the three-objective E-VReP is to find a set of routes in order to: 1) reduce the number of workers used; 2) maximize the number of relocation requests satisfied and 3) minimize the duration of the longest route. Figure 1 shows a feasible solution for an instance of the problem. In this instance, each parking station is assumed to be 5 Km far from the adjacent ones; also the depot is supposed to be distant 5 Km from every parking stations. Moreover we assume L= 150 Km, Γ= 4 hours, T=5 hours, s' =25 Km/h, s'' =15 Km/h and $q' = q'' = 1$ minute. In this way worker W_1 can leave from the depot at 14.10 arriving by bike in P_1 at 14.30; leaves from P_1 at 14.31 driving the EV and arrives at D_2 at 14.43 where the EV is delivered at 15 (left hand side TW of D_2); then arrives by bike at P_2 at 15.21 and driving the EV of P_2 arrives at D_4 at 15.34, where the EV is delivered at 16 (earliest time window of D_4) and then arrives at the depot by bike at 16.21. In this way the total time spent by worker W_1 is 1 hour and 50 minutes, thus not exceeding the work shift T=5 hours. While worker W_2 can leave from the depot at 8.10 arriving by bike in P_5 at 8.30, leaves from P_5 at 8.31 driving the EV and arrives at D_1 at 8.43 where the EV is immediately delivered. Thus W_2 arrives by bike to the depot at 9.04, spending 54 minutes. We notice that sometimes the battery charge level of the picked-up EV is lower than the value indicated for the delivered EV. For instance, for P_5 it is 0.6 while for D_1 is 0.8. Nevertheless, the solution is feasible since the EV is delivered in D_1 at 8.43 with a battery level of 0.57 (since the battery consumption to go from P_5 to D_1 is 0.03) but at 10 (latest time window of D_1) the EV reaches a battery level of 0.89 > 0.8 since in the 77 minutes, from 8.43 to 10, the battery level increases of 0.32, since the EV is always connected to the docking station.

We can observe that the solution shown in Figure 1 maximizes the number of relocation requests satisfied. Indeed, both the delivery relocations requests D_1 and D_3 can only be served by the vehicle of the pick-up relocation request P_5 , for time window compatibility reasons. Thus, D_1 and D_3 are mutually exclusive and therefore the maximum number of delivery requests that can be satisfied is three and, consequently, also the number of pick-up requests. Hence, the current solution is optimum with respect to this objective since serves exactly six requests. The same requests cannot be served with one less worker, i.e. by only one worker, since among them there are requests like D_1 and D_3 whose time windows differ more than T (=5 hours). If the priority is given to the minimization of the number of workers used, and we neglect the third objective, then only the route of worker W_1 is chosen since it dominates the route of worker W_2 in terms of number of relocation requests satisfied (and of course, at least one worker must be used in order to have relocation). While, if we consider also the third objective, then the route of worker W_2 is no more dominated since its duration (56 minutes) is lower than that of W_1 (92 minutes). We can also observe that the solution of Figure 1, corresponding to the optimization of only the second objective, is very unbalanced in terms of workload of the two workers, being their route duration difference of 36 minutes. Therefore, if the priority is given to the fairness among the workers we have to optimize the third objective, i.e., the duration of the longest route. In this case a solution consists for instance in changing the route of W_1 in the sequence {depot, P_1, D_2, depot}, thus both W_1 and W_2' routes last 56 minutes and the fairness among the worker is reached. Another solution that also optimizes the third objective consists in adding to the previous solution a third worker, W_3, with route given by {depot, P_2, D_4, depot}. In this way all the routes have again the same duration of 56 minutes, but the second objective (i.e., the maximization of the number of relocation requests satisfied) is improved, while the first objective is worsen, since one more worker is used. Hence, this is also a Pareto optimal solution.

A possible way to make the final decision among all the Pareto optimal solutions found consists in choosing that nearest to the *Ideal solution*, i.e., the unreal solution where all the objectives reach their optimal values (Dinh, 2016). Therefore, in this example, the Ideal solution consists in using only one worker, in satisfying six requests and in having a route duration of only 56 minutes. Hence, the (final) nearest solution to the Ideal solution consists in using only worker W_1 that serves four requests according to the route depicted in Figure 1.

Let us consider, in the following, the MILP model proposed in Bruglieri et al. (2014a) that will be taken into account in Section 3 for solving the three-objective version of the problem. The E-VReP was formulated introducing an auxiliary graph $G =< R \cup \{0\}, A >$, with A, the set of arcs linking each pair of compatible relocation requests.

$$\max \sum_{k=1}^{K} \sum_{(i,j)\in A: i\neq 0} x_{ijk} \quad (1)$$

$$\sum_{j\in\delta^+(0)} x_{0jk} \leq 1 \quad \forall k = 1, \dots, K \quad (2)$$

$$\sum_{k=1}^{K} \sum_{j\in\delta^+(i)} x_{ijk} \leq 1 \quad \forall i \in PUD \quad (3)$$

$$\sum_{j\in\delta^+(i)} x_{ijk} - \sum_{j\in\delta^-(i)} x_{jik} = 0 \quad \forall i \in PUDU\{0\}, \forall k = 1, \dots, K \quad (4)$$

$$t_{ik} + c_{ij}x_{ijk} \leq t_{jk} + M\left(1 - x_{ijk}\right) \ \forall (i,j) \in A: j \neq 0, \forall k = 1, \dots, K \quad (5)$$

$$t_{ik} + c_{i0}x_{i0k} \leq t_{0k} + T + M(1 - x_{i0k}) \ \ \forall i \in \delta^-(0), \forall k = 1, \dots, K \quad (6)$$

$$\tau_i^{min} \leq t_{ik} \leq \tau_i^{max} \ \ \forall i \in PUD, \forall k = 1, \dots, K \quad (7)$$

$$d_{ij}x_{ijk} \leq L\left(\rho_i + \frac{t_{ik} - \tau_i^{min}}{\Gamma}\right) \ \ \forall (i,j) \in A: i \in P, j \in D, \forall k = 1, \dots, K \quad (8)$$

$$\rho_i + \frac{t_{ik} - \tau_i^{min}}{\Gamma} - \frac{d_{ij}}{\mathrm{L}}x_{ijk} \geq \rho_j - \frac{\tau_j^{max} - t_{jk}}{\Gamma} - \left(\rho_j + 1\right)\left(1 - x_{ijk}\right) \ \ \forall (i,j) \in A: i \in P, j \in D, \forall k = 1, \dots, K \quad (9)$$

$$1 - \frac{d_{ij}}{\mathrm{L}}x_{ijk} \geq \rho_j - \frac{\tau_j^{max} - t_{jk}}{\Gamma} - \left(\rho_j + 1\right)\left(1 - x_{ijk}\right) \ \forall (i,j) \in A: i \in P, j \in D, \forall k = 1, \dots, K \quad (10)$$

$$x_{ijk} \in \{0,1\} \quad \forall (i,j) \in A: i \in P, j \in D \ \forall k = 1, \dots, K \quad (11)$$

$$t_{ik} \geq 0 \quad \forall i \in PUDU\{0\}, \forall k = 1, \dots, K \quad (12)$$

where: the binary decision variable x_{ijk} is equal to 1 if and only if the arc *(i,j)* is traversed by the operator *k*, i.e., if node *j* is visited by *k* after node *i*, while the continuous non-negative decision variable t_{ik} denotes the arrival time of the operator *k* to the parking lot of node *i*. Moreover, $\delta^-(i)$ and $\delta^+(i)$ indicate the forward star and the backward

star of node *i*, respectively.
The objective function (1), to maximize, represents the total number of requests served; the constraints of type (2) impose that only one route be assigned to each operator used; constraints (3) assure that each relocation request is handled by only one operator; the constraint guaranteeing that each operator visits a node and leaves it is expressed in (4); the visit time of a node is given in the constraint of type (5). The time duration of each route is guaranteed in the constraint of type (6), while constraints (7) assure that the time windows within each node that has to be visited are respected.

The need to properly manage the limited driving range of the EVs is taken into account by constraints (8)--(10). In particular, constraints (8) enforce the distance travelled by each EV to be proportional to the residual battery level, while constraints (9)-(10) ensure that each EV is delivered at node *j* with a battery level not lower than ρ_j.

A three-objective programming formulation of the E-VReP can be obtained with the previous MILP replacing its objective function with the following three objective functions:

$$\min \sum_{k=1}^{K} \sum_{(0,j)\in A} x_{0jk} \qquad (13)$$

$$\max \sum_{k=1}^{K} \sum_{(i,j)\in A} x_{ijk} \qquad (14)$$

$$\min \max_{k=1,..,K} \sum_{(i,j)\in A} c_{ij} x_{ijk} \qquad (15)$$

where (13) represents the minimization of the total number of workers employed, (14) the maximization of the total number of relocation requests satisfied and finally, (15) the minimization of the duration of the longest route.
It is worth noting that the latter objective is introduced to favor, at the same time, minimization of the total route durations and the balancing of the route durations. In this way, it is possible to provide the decision makers with a set of solutions that take into account the service provider's objective (the minimization of the operational cost), expressed by (13), the satisfaction of users' demand, given by (14), and the workers' workload balance given by (15).

2.5. *Exact method to solve the three-objective E-VReP*

A possible way to solve the three-objective E-VReP presented is by considering all possible values of the workers used, from 1 to *K*, each one at a time, and then solving only a bi-objective problem, given by objectives (14) and (15). The latter problem can be solved using the constraint method, i.e. through a sequence of MILP problems where the objective (14) is transformed into a threshold constraint. The resulting exact method for solving the three-objective E-VReP is summarized in Figure 2. Note that at step 2.1.2, as soon as the MILP becomes infeasible, it is possible to exit from the for-cycle on ε since the problem would certainly be infeasible with greater values of ε, as constraint (17) turns out to be tighter. In step 2.1.3, each element added to the optimal Pareto front is a triplet consisting of the number of workers used, the total number of requests served and the maximum duration of the routes.

Such a method may require solving $K \cdot \min\{|P|, |D|\}$ MILP problems whose size increases in a quadratic way with the number of relocation requests (due to the routing variables x_{ijk}). Thus, it may become prohibitive to face real world instances in reasonable time (with one tenth of the relocation requests). For this reason, in the next section, a method for approximating the Pareto optimal front is proposed, due to the NP-hardness of the problem addressed.

Algorithm Exact_multiobjective_E-VReP

1. Let $F = \emptyset$;
2. **For each** $\bar{K} = 1, \dots, K$

2.1 **For each** $\varepsilon = 2, \dots, \ 2\ min\{|P|, |D|\}$ increased by a step equal to 2

2.1.1 Solve the following MILP:

min z

s.t.

$$z \geq \sum_{(i,j)\in A} c_{ij} x_{ijk} \quad \forall\, k = 1, \dots, \bar{K} \quad (16)$$

$$\sum_{k=1}^{\bar{K}} \sum_{(i,j)\in A} x_{ijk} \geq \varepsilon \quad (17)$$

and constraints (2)-(12) where all occurrences of K are substituted by $\bar{K}$;

2.1.2 **If** the problem is infeasible break;

2.1.3 **Else** let $F = F \cup \left\{\left(\bar{K},\ \sum_{k=1}^{\bar{K}} \sum_{(i,j)\in A} x_{ijk}^{*} - 1,\ z^{*}\right)\right\}$ being (x^{*}, z^{*}) the optimal solution found;

3. **Return** F.

Figure 2 Pseudo-code of the exact method for computing the Pareto front of the multiobjective E-VReP

Note that the maximum cardinality value of the Pareto optimal front, |F|, can be easily estimated in the following way

$$|F| \leq \sum_{k=1}^{\tilde{K}} (\mu - k + 1) = \tilde{K}(\mu + 1) - \sum_{k=1}^{\tilde{K}} k = \tilde{K}(\mu + 1) - \frac{\tilde{K}\left(\tilde{K} + 1\right)}{2} = \tilde{K}\left(\frac{2\mu - \tilde{K} + 1}{2}\right) \qquad (18)$$

where $\mu = \min\{|P|, |D|\}$ and $\tilde{K} = \min\{\mu, K\}$, since a worker can serve at most $\mu - k + 1$ pairs of requests when k workers are employed, and their number can vary from 1 to $\tilde{K}$ (since they have to serve at least one pair of requests).

3. A two-phase heuristic algorithm for the three-objective E-VReP

In this section, a heuristic solution approach is designed for solving the three-objective E-VReP presented in the pr search he se, the fe

Figure 2 Pseudo-code of the exact method for computing the Pareto front of the multiobjective E-VReP

It is worth noting that the term *route* here refers to a sequence of alternate pick-up and delivery requests, performed by a worker who leaves from the depot and returns to it. Moreover, two relocation requests are considered *compatible* if both can be satisfied, respecting, at the same time, their time windows, the remaining work

shift of the worker and the current battery level of the EV. This last feasibility condition is related to the use of a fleet of EVs which, as known, present a short driving range.

The first constructive heuristic, called *RandHeur1*, specifically designed and implemented in this work, randomly selects a compatible pair of pick-up and delivery relocation requests and adds it to the route if compatible with all the requests previously inserted in the route. Thanks to the random nature of this heuristic, it can be run more than once in order to generate different feasible solutions and thus to populate the set Π with a large number of alternatives.

In addition, a second random heuristic algorithm, called *RandHeur2*, is implemented. *RandHeur2* differs from *RandHeur1* , in the way the next request is selected. Indeed, instead of choosing the next request in a completely random way, it is randomly selected from among the p requests at a minimum distance from the previous one.

Finally, a modified version of *RandHeur1*, called *MRandHeur1*, is obtained from the former, saving in Π only the routes that are not dominated, i.e., those with least duration for the same number of requests served. This makes possible to obtain a smaller set of routes but of higher quality compared to those generated by *RandHeur1* at the same values of max_iter and max_routes. In the numerical result section (Section 5), the impact of *RandHeur1*, *RandHeur2* and *MRandHeur1*, in the approximation of the Pareto front, is analyzed in detail showing that the best results are obtained considering Π as the union of the routes generated by all the heuristics.

The general steps of those heuristics are outlined in detail in Figure 3. At step 2.4.1.2, two necessary conditions for the compatibility of requests r_1 and r_2 are tested: the first one concerns their time windows, the second one the battery charge level of the picked-up EV compared to the battery charge level required. If both of them do not fail, the procedure *Insert*(r_1, r_2, π) is called: it tries to insert, in a feasible way, a pair of requests (r_1, r_2) into the route π, returning TRUE if possible, FALSE if otherwise. Such a procedure is explained in Figure 4. Note that in the pseudo-code of *Insert*(r_1, r_2, π), a_r and w_r represent the arrival time at the location of the request r and the waiting time necessary to satisfy it, respectively. The steps from 1.1 to 1.4 guarantee that $w_{r_1} = 0$ for the first pick-up request r_1. Step 1.5 sets the current duration, Λ, of the route. Step 1.6 tests both if the duration is less than the work shift and if the battery level of the EV, when picked-up, is sufficient to arrive at the location of the delivery request r_2. In this case the pair (r_1, r_2) is inserted. In an analogous way, the rest of the code sets both the arrival and the waiting time of a pair (r_1, r_2) when other requests are already served in the route and checks if this can be done in a feasible way between a delivery request r_3 and a pick-up request r_4.

In addition, in *RandHeur1*, the procedure *Exist_request* $(\tilde{D}, r_1, \tilde{F})$ returns a random request r_2 in $\tilde{D}$ such that $(r_1, r_2) \in \tilde{F}$, while, in *RandHeur2*, a random request r_2 in $\tilde{D}$ among the first p at minimum distance from r_1 such that $(r_1, r_2) \in \tilde{F}$. Finally, in *RandHeur1*, the procedure *New_request*$(\tilde{P})$ returns a random request in $\tilde{P}$, while, in *RandHeur2*, a random request in $\tilde{P}$ among the first p at minimum distance from r_2.

Heuristics Outline

1. Let $\Pi = \emptyset, iter = 0$;
2. **While** ($iter \leq max\ \_iter$ and $|\Pi| \leq max\ \_routes$)
   - 2.1. Let $\tilde{P} = P$, $\tilde{D} = D$, $\tilde{F} = P \times D$;
   - 2.2. Let route $\pi = (0,0)$;
   - 2.3. Let r_1a random request in $\tilde{P}$;
   - 2.4. **While** ($\tilde{F} \neq \emptyset$)
     - 2.4.1. **While** (r_2=Exist_request ($\tilde{D}, r_1, \tilde{F}$));
       - 2.4.1.1. Let $\tilde{F} = \tilde{F} \backslash \{(r_1, r_2)\}$;
       - 2.4.1.2. **If** ($\tau_{r_1}^{min} + \frac{d_{r_1 r_2}}{s'} + q' + q'' > \tau_{r_2}^{max}$ or $\rho_{r_1} - \frac{d_{r_1 r_2}}{L} + \frac{\tau_{r_2}^{max} - \tau_{r_1}^{min}}{\Gamma} < \rho_{r_2}$)
       - 2.4.1.3. **then** go to 2.4.1
       - 2.4.1.4. **Else** Insert(r_1, r_2, π);
       - 2.4.1.5. **End If**
       - 2.4.1.6. **If** (Insert(r_1, r_2, π)=TRUE) **then** let $\tilde{P} = \tilde{P} \backslash \{r_1\}$, $\tilde{D} = \tilde{D} \backslash \{r_2\}$ and r_1=New_request($\tilde{P}$));
       - 2.4.1.7. **End If**
     - 2.4.2. **End While**
   - 2.5. **End While**
   - 2.6. Let $\Pi = \Pi \cup \{\pi\}$;
   - 2.7. $iter++$;
3. **End While**
4. **Return** Π;

Figure 3 The general pseudo-code of the constructive heuristics

Procedure Insert(r_1,r_2,π)

1. **If** (π =(0,0)) **then**

1.1. Let a_{r_2}=$max\{\tau_{r2}^{min}, \tau_{r1}^{min} + d_{r1r2}/s' + q' + q''\}$;

1.2. Let a_{r_1}=$min\{\tau_{r1}^{max}, a_{r_2} - d_{r1r2}/s' - q' - q''\}$;

1.3. Let $a_{r_2} = a_{r_1} + d_{r1r2}/s' + q' + q''$;

1.4. Let $w_{r_2} = \max\{0, \tau_{r_2}^{min} - a_{r_2}\}$;

1.5. Let $\Lambda = d_{0r_1}/s'' + d_{r_1r_2}/s' + q' + q'' + w_{r_2} + d_{r_20}/s''$;

1.6. **If** ($\Lambda \leq T$ and $\min\left\{\rho_{r_1} + \frac{a_{r_1} - \tau_{r_1}^{min}}{\Gamma}, 1\right\} \geq \frac{d_{r_1r_2}}{L}$)

1.7. **then** let $\pi = (0, r_1, r_2, 0)$ and **Return** TRUE;

1.8. **Else Return** FALSE;

2. Else

2.1 **If** $\exists\ (r_3, r_4) \in \pi$: $r_3 \in D \cup \{0\}$ and $r_4 \in P \cup \{0\}$ satisfying

2.1.1 $a_{r_3} + w_{r_3} + \frac{d_{r_3r_1}}{s''} \leq \tau_{r_1}^{max}$

2.1.2 $\max\left\{a_{r_3} + w_{r_3} + \frac{d_{r_3r_1}}{s''}, \tau_{r_1}^{min}\right\} + \frac{d_{r_1r_2}}{s'} + q' + q'' \leq \tau_{r_2}^{max}$

2.1.3 $\min\left\{\rho_{r_1} + (\max\left\{a_{r_3} + w_{r_3} + \frac{d_{r_3r_1}}{s''}, \tau_{r_1}^{min}\right\} - \tau_{r_1}^{min})/\Gamma, 1\right\} \geq \frac{d_{r_1r_2}}{L}$.

2.2 **then** the pair (r_1, r_2) is inserted between (r_3, r_4)

2.3 **Else Return** FALSE;

Figure 4 The pseudo-code of the procedure Insert(r_1,r_2,π)

In the second phase, in order to find a set of non-dominated solutions, the following MILP model is formulated to select the routes from Π, guaranteeing that no relocation request is served more than once:

$$\min \xi = \sum_{\pi \in \Pi} y_\pi \qquad (19)$$

$$\max \sigma = \sum_{r \in R} \sum_{\pi \in \Pi} b_{r\pi} y_\pi \qquad (20)$$

$$\min \Phi \qquad (21)$$

$$\sum_{\pi \in \Pi} b_{r\pi} y_\pi \leq 1 \quad \forall r \in R \qquad (22)$$

$$\Phi \geq \tau_\pi y_\pi \qquad \forall \pi \in \Pi \qquad (23)$$

$$y_\pi \in \{0,1\} \qquad \forall \pi \in \Pi \qquad (24)$$

where the binary decision variable y_π is equal to 1 if the feasible route $\pi \in \Pi$ is selected, and 0 if otherwise. As input data, the model receives $b_{r\pi}$ equal to 1 if the relocation request $r \in R$ is handled in the feasible route $\pi \in \Pi$, 0 if otherwise. The objective function (19), namely to minimize, represents the total number of routes selected, i.e., the total number of workers employed for EVs relocation; the objective function (20), which is to maximize, denotes the total number of relocation requests satisfied, while (21) represents the time duration of the longest route to minimize. The group of constraints (22) assures that there is no-overlapping, i.e., each relocation request belongs to at most one feasible route. It is worth noting that these constraints are not equal, since a relocation request may not be served and it also may not belong to any selected route. Finally, the group of constraints (23) assures that Φ is the duration of the longest route, being τ_π the duration of each route $\pi \in \Pi$.

The three-objective MILP formulation (19)--(24) is solved by adopting the ϵ-constraint programming, as detailed in Figure 5, where F denotes the Pareto efficient front. In this outline, $\sigma^I, \sigma^N, \Phi^I, \Phi^N$ represent the Ideal and the Nadir point (Dinh , 2016) of the number of relocation requests satisfied and of the longest route duration, respectively.

It is worth noting that the computational complexity of our solution approach depends only linearly on the total number of relocation requests, since the latter only makes an impact on the number of iterations in the for-cycle at step 3.3. Unlike the exact method presented in Section 3.5, the size of the MILPs, solved at steps 3.1.1 and 3.3.1, does not depend on the number of the relocation requests, but only on the number of the routes considered in set Π. In this way our method is also able to address large size instances.

Algorithm ϵ-constraint programming

1. Let $F = \emptyset$, let $\pi_1, \pi_2, \dots, \pi_h$ the routes in Π ordered by decreasing duration and let $\sigma^N = \sum_{r \in R} b_{r\bar{\pi}}$ where $\bar{\pi} = \underset{\pi \in \Pi}{\operatorname{argmin}} \tau_\pi$;
2. Let σ^I the number of relocation requests satisfied, found by solving the MILP formulation (8), (10)-(12).
3. **For each** worker set size $k = 1, \dots, K$
   - 3.1 **For each** $\epsilon_1 = \tau_{\pi_1}, \tau_{\pi_2}, \dots, \tau_{\pi_h}$
     - 3.1.1 Solve the MILP formulation (20), (22)--(24) with these additional constraints:
       - 3.1.1.1 $\sum_{\pi \in \Pi} y_\pi \leq k$, i.e., maximum number of workers to use;
       - 3.1.1.2 $\Phi \leq \epsilon_1$
     - 3.1.2 **If** the problem is infeasible **then** go to step 3.2;
     - 3.1.3 **Else** let $F = F \cup \{(k,\ \sigma_{\epsilon_1},\ \Phi_{\epsilon_1})\}$ being $(k,\ \sigma_{\epsilon_1},\ \Phi_{\epsilon_1})$ the optimal solution found;
     - 3.1.4 **End If**
   - 3.2 **End For**
   - 3.3 **For each** $\epsilon_2 = \sigma^N, \dots, \sigma^I$ (increased by a step equal to two)
     - 3.3.1 Solve the MILP (21)--(24) with these additional constraints:
       - 3.3.1.1 $\sum_{\pi \in \Pi} y_\pi \leq k$
       - 3.3.1.2 $\sigma \geq \epsilon_2$
     - 3.3.2 **If** the problem is infeasible **then** go to step 3.;
     - 3.3.3 **Else** let $F = F \cup \{(k,\ \sigma_{\epsilon_2},\ \Phi_{\epsilon_2})\}$ being $(k,\ \sigma_{\epsilon_2},\ \Phi_{\epsilon_2})$ the optimal solution found;
     - 3.3.4 **End If**
   - 3.4 **End For**
4. **End For**
5. Remove from *F* all the dominated triplets.

Return *F*.

Figure 5 The pseudo-code of the ϵ-constraint programming used in phase 2

4. Numerical results

The computational results shown in this section were obtained by implementing the two-phase approach, proposed in Section 4, in AMPL language (Fourer et al., 2002) and solving the MILP formulations through the state of the art solver CPLEX12.5 on a PC Intel Xeon 2:80 GHz with 2GB RAM. In all the random heuristics proposed, the max_iter and max_routes parameters were set to 1000 and 200, respectively. Moreover, in *RandHeur2*, p is equal to 4.

4.1. Performance indicators

The performances of the proposed approach were evaluated through three indicators: 1) the number of non-dominated solutions (η); 2) the distribution of the solutions along the Pareto front through the *Spacing* metric (*S*); 3) both the convergence and the diversity using the *Hypervolume* metric (*HV*).

It is worth noting that, since the objective functions take values of different order (e.g., the time duration is an order of magnitude more than of the two other objectives) and measure different aspects, the metrics are computed on the normalized values of the results. For instance, if a non-dominated solution *i* is characterized by the number of workers employed ξ_i, the normalized value ξ_i' is found as:

$$\xi_i' = \frac{\xi_i - \xi^{min}}{\xi^{max} - \xi^{min}} \qquad (25)$$

where $\xi^{min}$and ξ^{max} are the minimum and the maximum number of workers employed, respectively. In an analogous way, the normalized values of the other objectives are computed.

The *Spacing* metric, introduced to the literature by Schott (1995), aims to measure the range variance of neighboring solutions in the approximated Pareto front (Coello and Cortés, 2005). In other words, it measures the distribution of the solutions along the Pareto front and it is mathematically defined as:

$$S = \sqrt{\frac{1}{\eta}\sum_{i=1}^{\eta}\left(d_i - \bar{d}\right)^2} \qquad (26)$$

where $\bar{d}$ is the average of all the distances $d_i, \forall i = 1, \dots, \eta$ and the *i*-th distance d_i is defined as:

$$d_i = \min_{k \in F | k \neq i} \{|\xi_i' - \xi_k'| + |\sigma_i' - \sigma_k'| + |\Phi_i' - \Phi_k'|\}$$

The smaller the *S* value, the higher the diversification of *F* is.

Instead, the *Hypervolume* (*HV*), introduced in the literature by Zitzler and Thiele (1999), aims to measure both convergence and diversity (Jiang et al., 2014). In other words, it measures how far the points of the approximated Pareto front are from the *worst point* (*WP*) found with reference to the normalized values (i.e., (1,0,1), being the point whose coordinates are the worst values of the three objectives to optimize).

It is mathematically defined as:

$$HV = volume\left(\bigcup_{i}^{\eta} H_i\right) \qquad (27)$$

where the hypercube H_i is built taking *WP* and the *i*-th solution as the diagonal corners of the hypercube.

When the optimal Pareto front is known, the *Inverted Generational Distance* (*IGD*) can be used, since it is shown to be the most common metric adopted in the literature in that case (Riquelme et al., 2015). Introduced by Van Veldhuizen and Lamont (2000), it estimates the distance of the approximated Pareto front from the points belonging to the true front, and it is mathematically expressed as:

$$IGD = \frac{\sqrt{\sum_{i=1}^{\eta} d_i^2}}{\eta} \qquad (28)$$

Obviously, having a value of IGD equal to zero means that the found Pareto front exactly approximates the true one.

4.2. Results on benchmark instances

The solution approach was tested on the benchmark set of instances (AMAT), described in Bruglieri et al. (2014b). The AMAT set consists of 30 different instances, with 22 relocation requests on average.

Such a set was built exploiting the real data on private car movement provided by the AMAT agency. Those movements are specified through an Origin-Destination (O-D) matrix from/to several zones of Milan and the following time ranges: 7.00 a.m.-10.00 a.m.; 10.00 a.m.-4.00 p.m. and finally, 4.00 p.m.-8.00 p.m. Although the data referred to trips with different aims, the authors considered those related to occasional trips, as it is the typical reason for using carsharing services.

Moreover, five stations with four parking lots and 21 stations with two parking lots were considered. In order to estimate the movements performed through carsharing services, the authors intersected the O-D zones with a circular boundary of 500 meters around each charging station. In fact, this distance represents the area that is easily reachable by walking from a station. Then, the movements between each pair of stations were estimated considering 0.5% of the AMAT movements between the OD zones that fell in the catchment areas of the stations (to obtain a number of carsharing movements consistent with the current usage of carsharing in Milan). In addition, a fraction proportional to the overlap between the OD zones and the catchment areas was considered. Moreover, if the catchment areas of two stations overlapped, the movements concerning the OD zones that fell in the intersection were equally distributed between the two stations. The imbalances due to such estimated carsharing travel demand were computed through a Matlab simulator developed in Bruglieri et al. 2014(b).

The simulator had a time-step feed with data regarding the station capacities, the travel times between pairs of stations, and the travel demands. In particular, the simulator randomly generated the vehicle movements between pairs of stations following a probability distribution according to the carsharing travel demand estimated from the AMAT data. At each simulated minute, the inventory of each parking station and both the position and the charge level of the EVs were updated according to their movements.

Moreover, K was set to 16, since the maximum value over all the instances of min{|P|,|D|}=16; as a result, no more than 16 workers could be employed on this data set. The other parameters of Table 2, common to all the AMAT instances, were set as in the following: L= 150 Km, Γ= 4 hours, T=300 minutes, s' =25 Km/h, s'' =15 Km/h and $q' = q'' = 1$ minute.

The numerical results found by the two-phase approach described in Section 4, considering Π as the union of the solutions given by RandHeur1, RandHeur2 and MRandHeur1, are shown in Table 3. The results are given in detail for each instance of the AMAT set. The columns report the instance name, the number of pick-up requests (|P|), delivery requests (|D|), routes found through the heuristics (|Π|)), the number of Pareto front solutions generated (η), the upper bound on the cardinality of the optimal Pareto front (max |F|) given by (18), the maximum number of workers employed (ξ^{max}), the maximum percentage of relocation requests satisfied (σ^{max} %), the minimum and maximum duration of the longest route (Φ^{min} and Φ^{max}), the *Spacing* (S), the *Hypervolume* (HV), the CPU time (in seconds) required to generate Π, and the total CPU time (in seconds) required by the two-phase approach, respectively. The last row of Table 3 reports the average values or the median if the values considered are integer. The minimum number of workers employed and the minimum number of relocation requests served are not reported since they are always equal to one and two, respectively.

It is worth noting that our approach finds a number of non-dominated solutions very close to the maximum possible one (on average, only 5 fewer non dominated solutions). Concerning the *Spacing* metric, its average value is equal to 0.06, with a standard deviation of 0.03, and a minimum and maximum value equal to 0.03 and 0.12, respectively. This very small average value of S implies that our non-dominated solutions are well distributed along the Efficient Pareto fronts. Finally, the average *Hypervolume* was 9.66, with a standard deviation of 5.49, a minimum and a maximum value of 1.9 and 23.11, respectively. The high average *HV* value obtained proves that our solution approach generated a well-diversified non-dominated set. This can be mainly justified through the use of different heuristics for finding Π required in the first phase. Concerning the CPU time effort, our solution approach on average requires 16.92 seconds in the first phase, and 243.96 seconds in the second one.

In Table 4, the impact of the different heuristics used to build Π, both on the number of non-dominated solutions

found in phase 2 and on their quality, is analyzed more in depth. In particular, the columns denote the instance name, the percentage gap between the duration of the longest route found by *RandHeur1* and the one by *RandHeur1* and *RandHeur2* ($\Delta\Phi\%$), and their percentage gap on the number of non-dominated solutions found ($\Delta\eta$), respectively. Analogously, the last two columns report the same gaps between the solutions found by RandHeur1and RandHeur2, and those found by *RandHeur1*, *RandHeur2* and *MRandHeur1*. The last row of the table reports the average values.

Concerning the number of non-dominated solutions, in about more than half the instances (56%), there is no improvement when the solutions of *RandHeur2* are added to those of *RandHeur1*. By adding also those of *MRandHeur1*, an improvement occurs in only one third of the instances (33%). Moreover, in more than one fourth of the instances (27%), there is a decrease of η when the solutions of *RandHeur1* are added to those of *RandHeur2*, something that occurs in almost half of the instances (47%) when those of *MRandHeur1* are also added. This behavior can be justified by the fact that some of the new solutions added dominate some of the previous ones since the successive heuristics are of higher quality than *RandHeur1*, which is completely random. This improvement of the quality can be noted by the fact that the average duration of the longest route decreases by 0.97% when the solutions of *RandHeur2* are added to those found by *RandHeur1*, and by 6.57% when those of *MRandHeur1* are added.

The approximated Pareto front is also compared with the optimal one found through the procedure described in Figure 2. This comparison is possible only in a few instances due to the prohibitive CPU time required by the latter. In fact, all the instances for which $\min\{|P|, |D|\} \leq 7$ were tested, yielding a CPU time limit of 3,600 seconds to Cplex for solving each MILP in step 2.1.1 of Figure 2. However, only on three of these (AMAT6, AMAT12 and AMAT15), Cplex was able to solve all the MILPs to optimality. Such comparisons, considering the non-dominated solutions where both the approaches (the two-phase and the exact one) serve the same number of relocation requests with the same number of workers, are reported in Tables 4, 5 and 6. In all these tables, the columns represent the number of workers employed (ξ), the number of relocation requests served (σ), the duration of the longest route found by the two-phase approach (Φ^{Heur}) and by the exact one (Φ^{*}), respectively, and the relative percentage gap between them ($\Delta\Phi\%$). It is worth noting that on AMAT6, through the two-phase approach, 11 Pareto optimal solutions are found out of 22 (they are emphasized in boldface in the last column); on AMAT12, 11 out of 21 and finally, on AMAT15, 9 out of 14. Moreover, the high quality of the approximation of the optimal Pareto front is also confirmed by the very small values of the IGD metric obtained, on these instances being equal to 0.023, 0.025 and 0.039, respectively. In addition, the CPU time effort required by Cplex12.5 to compute the optimal Pareto front is by far higher than that required by the two-phase approach, being 296.85 seconds against the 111.50 seconds of the latter, on AMAT6; it is 103.45 seconds against the 17.35 seconds on AMAT12 and finally, 137.17 seconds against 3.59 seconds on AMAT15.

Table 3: Numerical results found through the two-phase approach on the AMAT instances

Instance		|P|	|D|	$\|\Pi\|$	η	max |F|	ξ^{max}	σ^{max}%	Φ^{min}	Φ^{max}	S	HV	H-CPU time	Total CPU time
AMAT	1	15	12	265	62	78	12	88.89	28.47	276.85	0.09	16.67	26.88	374.97
AMAT	2	15	7	243	26	28	6	63.64	26.83	290.01	0.07	5.33	11.67	235.79
AMAT	3	16	6	248	20	21	6	54.55	22.46	198.81	0.07	3.79	12.76	225.87
AMAT	4	15	14	254	90	105	12	96.55	29.61	291.04	0.06	21.31	23.58	442.66
AMAT	5	13	11	246	61	66	11	91.67	9.57	288.3	0.04	12.38	19.28	362.15
AMAT	6	7	8	229	26	28	7	93.33	21.73	166.91	0.08	5.72	13.17	310.02
AMAT	7	9	8	233	26	36	7	82.35	28.37	268.7	0.07	5.73	19.26	305.43
AMAT	8	11	10	232	41	55	9	85.71	26.85	209.37	0.05	8.61	12.44	303.58
AMAT	9	15	7	243	23	28	6	63.64	20.36	174.83	0.06	4.12	17.89	262.11
AMAT	10	11	11	237	59	66	11	100.00	36.34	281.84	0.05	13.15	14.01	312.24
AMAT	11	10	8	240	33	36	7	88.89	20.36	264.03	0.12	10.79	12.88	290.16
AMAT	12	12	6	231	20	21	6	66.67	43.15	219.68	0.08	4.28	10.88	114.33
AMAT	13	13	12	241	60	78	11	88.00	9.57	228.43	0.04	11.37	19.04	392.69
AMAT	14	6	12	228	17	21	5	66.67	20.92	113.93	0.03	2.34	8.64	278.42
AMAT	15	11	5	228	13	15	4	62.50	24.79	263.9	0.12	1.9	16.23	153.4
AMAT	16	16	11	255	63	66	10	81.48	13.19	291.79	0.08	15.86	20.02	244.16
AMAT	17	15	16	264	105	120	14	96.77	9.84	253.36	0.03	23.11	40.24	367.79
AMAT	18	10	12	238	45	55	10	90.91	9.84	282.79	0.11	8.35	14.93	226.39
AMAT	19	15	7	238	27	28	7	63.64	38.55	174.08	0.07	6.06	11.56	166.69
AMAT	20	17	8	249	35	36	8	64.00	20.92	291.61	0.04	7.88	20.48	193.62
AMAT	21	12	16	255	70	78	12	85.71	20.92	280.24	0.04	15.23	26.36	245.47
AMAT	22	13	8	238	27	36	7	66.67	27.93	279.48	0.05	6.46	14.9	183.96
AMAT	23	9	9	235	43	45	8	100.00	28.37	293.37	0.11	14.07	11.78	212.79
AMAT	24	12	7	243	26	28	6	73.68	20.36	191.62	0.04	4.66	14.61	260.86
AMAT	25	7	9	224	24	28	7	87.50	22.78	122.23	0.1	3.11	10.81	160.17
AMAT	26	11	11	245	55	66	10	100.00	9.84	289.85	0.04	11.9	17.89	249.3
AMAT	27	14	12	262	71	78	11	92.31	26.83	269.62	0.04	15.2	23.23	344.68
AMAT	28	13	11	238	50	66	10	83.33	9.84	265.49	0.04	11.34	14.78	202.92
AMAT	29	13	10	229	46	55	10	86.96	39.03	251.06	0.04	10.67	11.62	214.74
AMAT	30	9	15	221	40	45	8	75.00	36.34	255.08	0.07	8.5	15.91	189.03
Average		**13**	**10**	**239**	**40**	**45**	**8**	**81.37**	**23.25**	**246.37**	**0.06**	**9.66**	**16.92**	**260.88**

Table 4: Impact of the heuristics on the generation of Π

		RandHeur1 vs RandHeur1+2		RandHeur1+2 vs all RandHeur	
Instance		**ΔΦ %**	**Δη**	**ΔΦ %**	**Δη**
AMAT	1	1.42	0	5.21	-8.82
AMAT	2	0	0	1.64	0
AMAT	3	0	0	2.64	0
AMAT	4	1.64	-5.21	4.52	-1.1
AMAT	5	0.69	0	10.88	-3.17
AMAT	6	0.89	0	3.85	0
AMAT	7	0.02	0	6.65	-3.7
AMAT	8	0.09	5	6.86	-2.38
AMAT	9	3.36	-8	7.49	0
AMAT	10	1.37	-1.67	1.98	0
AMAT	11	0	0	9.98	0
AMAT	12	0	0	3.87	0
AMAT	13	1.41	-4.48	10.01	-6.25
AMAT	14	1.34	-5.26	7.69	-5.56
AMAT	15	0	0	9.34	-7.14
AMAT	16	1.03	0	4.63	6.78
AMAT	17	3.02	7.84	6.56	-4.55
AMAT	18	1.31	-2.08	10.41	-4.26
AMAT	19	0	0	3.95	3.85
AMAT	20	0.69	0	6.33	2.94
AMAT	21	0.29	-2.82	5.52	1.45
AMAT	22	1.06	0	2.5	0
AMAT	23	0.51	0	4.13	7.5
AMAT	24	1.4	0	10.66	0
AMAT	25	0.56	4.17	1.79	-4
AMAT	26	4.89	3.92	17.45	3.77
AMAT	27	0.88	-2.74	3.09	0
AMAT	28	0.76	4	8.72	-3.85
AMAT	29	0.36	0	7.29	-11.54
AMAT	30	0.24	0	11.48	-2.44
Average		**0.97**	**-0.2**	**6.57**	**-1.42**

Table 5: Comparisons between the approximated and the optimal Pareto front on instance AMAT 6

ξ	σ	Φ^{Heur}	Φ^*	$\Delta\Phi\%$
1	2	21.73	21.73	**0.00**
1	4	37.76	37.76	**0.00**
1	6	64.92	64.92	**0.00**
1	8	81.93	81.94	**0.00**
1	10	109.14	109.14	**0.00**
1	12	146.5	145.91	0.40
2	4	21.73	21.73	**0.00**
2	6	37.76	37.76	**0.00**
2	8	57.29	48.89	17.18
2	10	64.92	64.92	**0.00**
2	12	105.52	72.17	46.21
2	14	166.91	115.91	44.00
3	6	29.99	29.99	**0.00**
3	8	37.76	37.76	**0.00**
3	10	57.29	48.89	17.18
3	12	64.92	52.61	23.40
3	14	105.52	72.17	46.21
4	8	29.99	29.99	**0.00**
4	12	57.29	51.73	10.75
4	14	64.92	56.78	0.90
5	12	55.49	50.51	28.53
5	14	58.66	52.61	11.50
Average				**11.19**

Table 6: Comparisons between the approximated and the optimal Pareto front on instance AMAT 12

ξ	σ	Φ^{Heur}	Φ^*	$\Delta\Phi\%$
1	2	43.15	43.15	**0.00**
1	4	63.88	63.88	**0.00**
1	6	106.52	80.04	33.08
1	8	144.71	114.08	26.85
1	10	188.16	154.07	22.12
1	12	219.68	194.07	13.20
2	4	43.15	43.15	**0.00**
2	6	63.88	63.88	**0.00**
2	8	83.14	69.72	19.25
2	10	106.52	83.15	28.11
2	12	158.44	110.10	43.91
3	6	44.56	44.56	**0.00**
3	8	63.88	63.88	**0.00**
3	10	83.14	69.72	19.25
3	12	94.21	83.15	13.30
4	8	47.31	47.31	**0.00**
4	10	63.88	63.88	**0.00**
4	12	83.14	69.72	19.25
5	10	49.82	49.82	**0.00**
5	12	63.88	63.88	**0.00**
6	12	59.41	59.41	**0.00**
Average				**11.35**

Table 7: Comparisons between the approximated and the optimal Pareto front on instance AMAT 15

ξ	σ	Φ^{Heur}	Φ^*	$\Delta\Phi\%$
1	2	24.79	24.79	**0.00**
1	4	41.04	41.04	**0.00**
1	6	76.22	73.38	3.87
1	8	116.68	108.92	7.12
1	10	263.9	160.57	**0.00**
2	4	35.90	35.90	**0.00**
2	6	41.04	41.04	**0.00**
2	8	68.23	67.93	0.44
2	10	79.9	77.51	3.08
3	6	38.55	38.55	**0.00**
3	8	41.04	41.04	**0.00**
3	10	68.23	67.93	0.44
4	8	38.55	38.55	**0.00**
4	10	47.77	47.77	**0.00**
Average				**1.07**

4.3. Results on large size instances

For the aim of testing the performances of the two-phase approach also on large size instances, a second benchmark set of instances, called L-AMAT, was considered. This set was composed of 20 instances of the E-VReP with a number of relocation requests of between 83 and 100 and an average value of 89. The general parameters of Table 2 were set as for the AMAT instances (Section 4.2).

The numerical results found by the two-phase approach, considering Π again as the union of the solutions given by *RandHeur1*, *RandHeur2* and *MRandHeur1*, are shown in Table 8. The columns of Table 8 have the same meaning as those in Table 3. It is worth noting that even though the average number of relocation requests considered is four times that of the AMAT set, the total CPU time required is less than double. Therefore, in practice, our solution approach is able to address large size instances in reasonable time, in accordance with its linear computational time dependency on the total number of relocation requests (as observed in Section 4). The small average value of the *Spacing* indicator obtained (S = 0.02) proves that the detected non-dominated solutions were well distributed. The high average *Hypervolume* value obtained (HV =24.60) shows that the solution approach also generates a well-diversified non-dominated set.

Table 8: Numerical results found through the two-phase approach on the L-AMAT instances, with K=16

Instance	|P|	|D|	$\|\Pi\|$	η	max |F|	ξ^{max}	σ^{max}%	Φ^{min}	Φ^{max}	S	HV	H-CPU time	Total CPU time
L-AMAT 1	37	50	262	333	472	16	85.06	9.84	296.04	0.02	25.08	226.77	680.77
L-AMAT 2	49	47	261	328	632	16	83.33	9.57	294.48	0.03	25.7	207.58	637.57
L-AMAT 3	51	38	245	311	488	16	83.15	2	288.31	0.02	25.26	158.77	583.76
L-AMAT 4	46	44	250	308	584	16	86.67	9.57	296.12	0.02	21.4	335.33	697.33
L-AMAT 5	45	40	242	313	520	16	91.76	9.84	296.06	0.02	24.61	145.41	472.41
L-AMAT 6	42	44	251	327	552	16	93.02	20.89	298.17	0.02	23.03	101.84	455.83
L-AMAT 7	47	41	264	359	536	16	93.18	9.57	299.32	0.02	26.17	124.74	594.73
L-AMAT 8	45	49	253	304	600	16	82.98	20.36	299.19	0.02	23.66	60.05	376.05
L-AMAT 9	45	38	268	336	488	16	91.57	20.09	296.48	0.02	27.9	93.01	520.01
L-AMAT 10	55	28	247	254	328	16	67.47	9.57	293.54	0.02	26.94	38.16	381.16
L-AMAT 11	44	43	260	318	568	16	91.95	9.84	299.15	0.02	24.74	75.76	390.76
L-AMAT 12	54	41	268	291	536	16	84.21	13.32	289.93	0.02	19.79	63.79	451.79
L-AMAT 13	50	39	242	346	504	16	87.64	9.57	284.86	0.02	25.93	72.50	515.50
L-AMAT 14	42	44	241	317	552	16	90.70	9.84	294.38	0.02	24.11	71.11	506.11
L-AMAT 15	44	47	269	366	584	16	96.70	9.57	296.63	0.02	26.24	136.79	531.79
L-AMAT 16	60	34	239	265	424	16	72.34	27.79	292.47	0.02	23.34	28.44	366.44
L-AMAT 17	46	39	264	337	504	16	91.76	9.84	296.55	0.02	25.38	74.09	445.09
L-AMAT 18	49	43	250	366	568	16	91.30	9.84	295.09	0.02	23.44	90.66	573.66
L-AMAT 19	45	55	248	423	600	16	90.00	9.84	298.26	0.01	26.44	157.50	689.50
L-AMAT 20	46	39	245	303	504	16	82.35	20.36	298.43	0.02	22.8	72.68	345.68
Average	**46**	**42**	**251**	**323**	**536**	**16**	**86.86**	**12.55**	**295.17**	**0.02**	**24.60**	**116.75**	**510.80**

The numerical results shown in Table 8 are found by setting the maximum number of workers K equal to 16 while, the ones reported in Table 9 are obtained with K equal to 30. It is worth noting that using more workers, the number of relocation requests increases of about 12.57% as well as the number of Pareto front solutions generated of about 54.49%. Moreover, the average S is equal to 0.01, i.e., the Pareto front solutions are distributed better than the case with K=16. Finally, the average HV is equal to 64.67, i.e., the Pareto front solutions are diversified better than the case with K=16. However, the average total CPU time increases of about 487.45 seconds.

Table 9: Numerical results found through the two-phase approach on the L-AMAT instances, with K=30

Instance	|P|	|D|	|Π|	η	max |F|	ξ^{max}	σ^{max}%	Φ^{min}	Φ^{max}	S	HV	H-CPU time	Total CPU time
L-AMAT 1	37	50	262	492	675	30	100.00	9.84	296.04	0.01	64.68	226.77	1210.77
L-AMAT 2	49	47	261	581	975	30	93.62	9.57	294.48	0.02	70.61	207.58	1234.58
L-AMAT 3	51	38	245	479	705	30	97.37	2	288.31	0.01	70.48	158.77	1083.77
L-AMAT 4	46	44	250	524	885	30	95.45	9.57	296.12	0.01	59.05	335.33	1114.33
L-AMAT 5	45	40	242	490	765	30	100.00	9.84	296.06	0.01	65.97	145.41	736.41
L-AMAT 6	42	44	251	503	825	30	100.00	20.89	298.17	0.01	58.53	101.84	498.84
L-AMAT 7	47	41	264	576	795	30	100.00	9.57	299.32	0.01	71.69	124.74	1203.74
L-AMAT 8	45	49	253	489	915	30	91.11	20.36	299.19	0.02	60.64	60.05	812.05
L-AMAT 9	45	38	268	494	705	30	100.00	20.09	296.48	0.01	74.86	93.01	1057.01
L-AMAT 10	55	28	247	311	406	30	100.00	9.57	293.54	0.02	50.41	38.16	611.16
L-AMAT 11	44	43	260	490	855	30	95.35	9.84	299.15	0.02	64.9	75.76	858.76
L-AMAT 12	54	41	268	473	795	30	97.56	13.32	289.93	0.01	56.14	63.79	954.79
L-AMAT 13	50	39	242	520	735	30	100.00	9.57	284.86	0.01	68.89	72.5	857.5
L-AMAT 14	42	44	241	506	825	30	95.24	9.84	294.38	0.01	63.24	71.11	937.11
L-AMAT 15	44	47	269	544	885	30	100.00	9.57	296.63	0.01	66.98	136.79	1672.79
L-AMAT 16	60	34	239	400	585	30	100.00	27.79	292.47	0.02	65.83	28.44	770.44
L-AMAT 17	46	39	264	538	735	30	100.00	9.84	296.55	0.01	71.94	74.09	999.09
L-AMAT 18	49	43	250	550	855	30	97.67	9.84	295.09	0.01	64.81	90.66	1046.66
L-AMAT 19	45	55	248	644	915	30	100.00	9.84	298.26	0.01	69.53	157.5	1441.5
L-AMAT 20	46	39	245	425	735	30	92.31	20.36	298.43	0.01	54.21	72.68	863.68
Average	**46**	**42**	**251**	**499**	**795**	**30**	**97.78**	**12.56**	**295.17**	**0.01**	**64.67**	**116.75**	**998.25**

4.4. Results on very large size instances

Finally, the performances of the two-phase approach were tested also on a set of very large size instances, called VL-AMAT, generated through the Matlab simulator (described in Section 4.2). This set was composed of 20 instances of the E-VReP with a number of relocation requests of between 445 and 536 and an average value of 489. The general parameters of Table 2 were set as for the AMAT instances (Section 4.2).

The numerical results found by the two-phase approach, considering Π again as the union of the solutions given by the three heuristics, are shown in Table 10. The columns of Table 10 have the same meaning as those in Table 3. It is worth noting that even though the average number of relocation requests considered is about five times that of the L-AMAT set, the total CPU time required is less four times. This again proves that our solution approach addresses also the very large size instances in a reasonable amount of time, in accordance with its linear computational time dependency on the total number of relocation requests (as observed in Section 4). The small

average value of the *Spacing* indicator obtained (S = 0.01) proves that the detected non-dominated solutions were well distributed. The high average *Hypervolume* value obtained (HV =51.56) shows that the solution approach also generates a well-diversified non-dominated set. Figure 6 plots the total CPU time required by the proposed two-phase approach, varying the instance size.

Table 10: Numerical results found through the two-phase approach on the VL-AMAT instances

Instance	|P|	|D|	$\|\Pi\|$	η	max |F|	ξ^{max}	σ^{max}%	Φ^{min}	Φ^{max}	S	HV	H-CPU time	Total CPU time
VL-AMAT 1	169	314	262	1131	4635	30	56.71	6.00	291.94	0.01	43.38	565.589	2015.589
VL-AMAT 2	206	330	250	1261	5745	30	46.53	9.57	294.67	0.01	59.02	875.269	2686.269
VL-AMAT 3	150	317	261	1281	4065	30	59.73	11.64	291.12	0.01	58.71	505.949	2004.949
VL-AMAT 4	192	253	261	1250	5325	30	49.20	10.16	290.2	0.01	53.7	461.121	1990.121
VL-AMAT 5	175	313	259	1270	4815	30	54.76	9.57	295.06	0.01	55.53	493.339	2343.339
VL-AMAT 6	198	328	248	1233	5505	30	50.76	9.57	293.96	0.01	44.19	642.745	3222.745
VL-AMAT 7	180	301	247	1238	4965	30	53.11	9.57	293.71	0.01	50.76	449.234	1745.234
VL-AMAT 8	167	314	247	1197	4575	30	59.64	9.57	294.55	0.01	45.89	580.758	2476.758
VL-AMAT 9	140	310	246	1160	3765	30	66.91	9.57	297.73	0.01	49.1	409.798	1753.798
VL-AMAT 10	102	390	250	1185	2625	30	86.27	9.57	291.13	0.01	55.21	1099.222	2527.222
VL-AMAT 11	155	335	248	1214	4215	30	58.17	20.09	294.75	0.01	56.59	150.561	1830.561
VL-AMAT 12	150	342	248	1165	4065	30	59.46	2.00	299.14	0.01	53.81	577.396	2187.396
VL-AMAT 13	164	356	248	1321	4485	30	59.26	20.09	296.28	0.01	57.52	499.699	2143.699
VL-AMAT 14	157	337	250	1186	4275	30	58.71	2.00	297.5	0.01	49.86	694.681	2163.681
VL-AMAT 15	128	342	253	1091	3405	30	67.20	9.84	295.31	0.01	48.82	210.237	2159.237
VL-AMAT 16	161	304	251	1235	4395	30	54.43	9.57	299.56	0.01	60.01	374.872	2043.872
VL-AMAT 17	156	332	252	1218	4245	30	60.53	2.00	291.43	0.01	46.05	255.107	1614.107
VL-AMAT 18	202	308	252	1208	5625	30	48.22	2.00	293.33	0.01	43.99	1288.757	2895.757
VL-AMAT 19	170	333	243	1119	4665	30	54.76	9.57	298.72	0.01	45.19	511.831	1832.831
VL-AMAT 20	186	314	238	1275	5145	30	53.80	9.57	294.9	0.01	53.94	358.118	1771.118
Average	**166**	**323**	**250**	**1216**	**4530**	**30**	**57.91**	**9.08**	**294.75**	**0.01**	**51.56**	**550.21**	**2170.41**

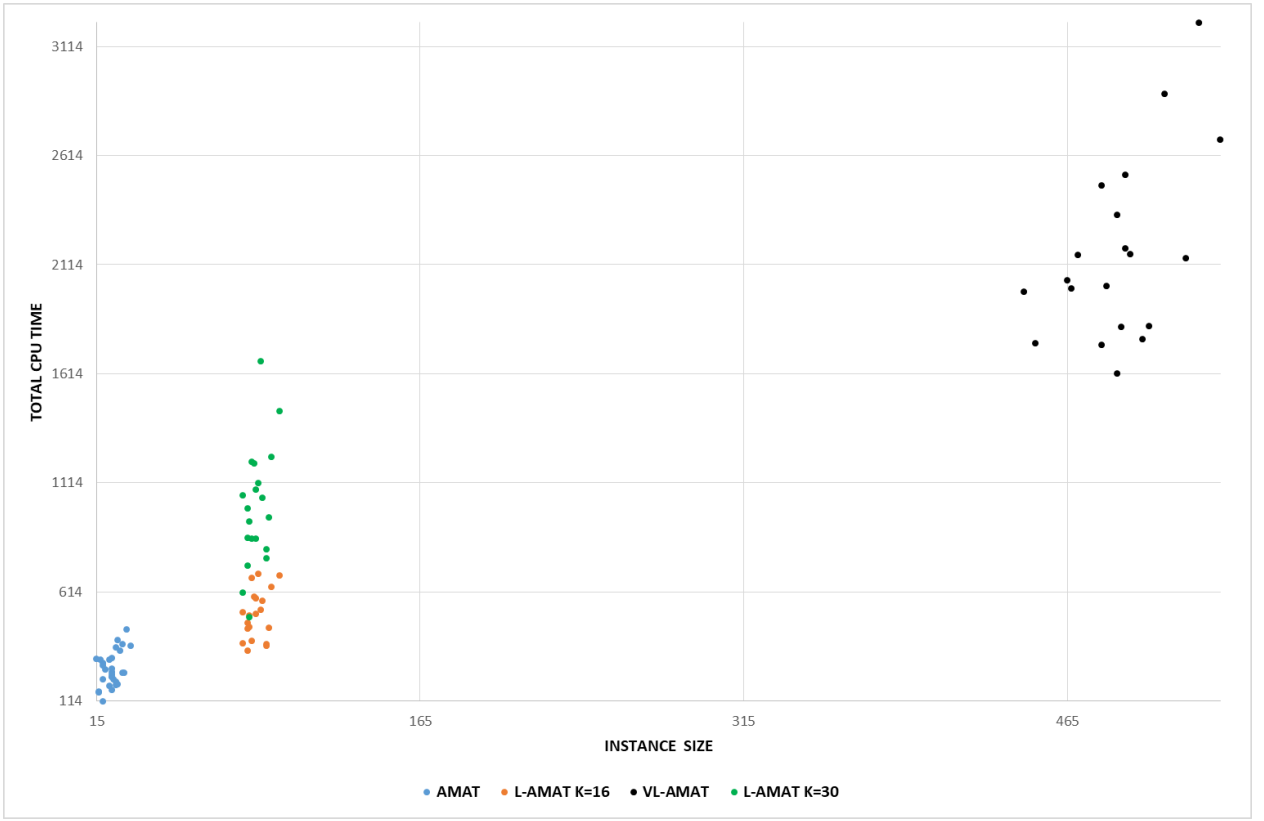

Figure 6 Plot of the total CPU time varying the instance size

5. Conclusions and future research

A three-objective relocation problem was addressed in one-way electric carsharing systems where the workers move from a delivery request to one of pick-up using a folding bicycle. This guarantees, from the carsharing provider point of view, a minimization of the total costs due to the number of workers used; from the workers' perspective a fair distribution of the relocation requests among them; from the users' point of view, the maximization of the service level (i.e., the number of relocation requests satisfied).

This problem was modelled through a three-objective mixed integer linear programming formulation and solved in an approximated way by means of a two-phase approach. In the first phase, different heuristic algorithms, suitably designed for the problem, were used to populate the set Π of feasible solutions. In the second phase, Π was used to generate a set of non-dominated solutions through ϵ-constraint programming.

Our solution approach to approximate the optimal Pareto front has been tested on three real like sets of instances (AMAT, L-AMAT and VL-AMAT) with size about 30, 100 and 500 relocation requests, respectively. It was able to detect a high number of non-dominated solutions (on average, 40 solutions on the AMAT instances, 323 on the L-AMAT with the maximum number of workers K equal to 16, 499 on the L-AMAT with K equal to 30 and finally, 1216 on the VL-AMAT), all well distributed as proven by the small *Spacing* value (on average, 0.06, 0.02, 0.01 and 0.01 on AMAT set, L-AMAT set with K=16, L-AMAT set with K=30 and VL-AMAT set, respectively), and well diversified as shown by the high *Hypervolume* value (on average, 9.66, 24.60, 64.67 and 51.56, on AMAT set, L-

AMAT set with K=16, L-AMAT with K=30 and VL-AMAT set, respectively).

Moreover, on three instances of the AMAT benchmark set, it was possible to compare the solutions of the two-phase method with the optimal Pareto front, observing that half of them coincided. In the other half of the solutions, the two-phase method had an average worsening of the duration of the longest route of less than 8%. However, the two-phase approach is by far faster, requiring an average CPU time of about 260 seconds against the 44,148 seconds of the exact one. Therefore, the two-phase approach reveals itself to be much faster than the exact method with, on average, a small deterioration in the solution quality, only concerning the longest route duration.

Future research work concerns investigating other possible methods to compute the Pareto optimal front in a more efficient way, although all exact methods, unlike our heuristic solution approach, cannot require a polynomial computational time due to the NP-hardness of the problem.

References

Barth, M.,Todd, M., 1999. Simulation model performance analysis of a multiple station shared vehicle system. Transportation Research Part C: Emerging Technologies 7, no. 4, 237-259.

Barth, M., Todd, M., Xue, L., 2004. User-based vehicle relocation techniques for multiple-station shared-use vehicle systems. Transportation Research Record 1887, 137-144.

Boyacı, B., Zografos, K. G., Geroliminis, N., 2015. An optimization framework for the development of efficient one-way car-sharing systems. European Journal of Operational Research, 240 (3), 718-733.

Boyacı, B., Zografos, K. G., & Geroliminis, N., 2017. An integrated optimization-simulation framework for vehicle and personnel relocations of electric carsharing systems with reservations. Transportation Research Part B: Methodological, 95, 214-237.

Brandstätter, G., Gambella, C., Leitner, M., Malaguti, E., Masini, F., Puchinger, J., Vigo, D. (2016). Overview of Optimization Problems in Electric Car-Sharing System Design and Management. In Dynamic Perspectives on Managerial Decision Making (pp. 441-471). Springer International Publishing.

Bruglieri, M., Colorni, A., Luè, A., 2014(a). The vehicle relocation problem for the one-way electric vehicle sharing. Networks, 64(4), 292-305.

Bruglieri, M., Colorni, A., Luè, A., 2014(b). The vehicle relocation problem for the one-way electric vehicle sharing: an application to the Milan case. Procedia Social and Behavioral Sciences 111, 18-27.

Bruglieri, M., Pezzella, F., Pisacane, O., 2017. Heuristic algorithms for the operator-based relocation problem in one-way electric carsharing systems. Discrete Optimization, 23, 56-80.

Bruglieri, M., Pezzella, F., Pisacane, O., 2018. An Adaptive Large Neighborhood Search for Relocating Vehicles in Electric Carsharing Services. Discrete Applied Mathematics (to appear), DOI: 10.1016/j.dam.2018.03.067.

Cao, G., Wang, L., Jin, Y., Yu, J., Ma, W., Liu, Q., ... & Fu, T., 2016. Determination of the Vehicle Relocation Triggering Threshold in Electric Car-Sharing System. In Proceedings of 2016 Chinese Intelligent Systems Conference (pp. 11-22). Springer Singapore.

Chankong, V., Haimes, Y.Y.,1983. Multiobjective decision making.Theory and methodology. North-Holland.

Coello, C. A., Cortés, N. C, 2005. Solving Multiobjective Optimization Problems Using an Artificial Immune System. Genetic Programming and Evolvable Machines, 6, 163–190.

Di Febbraro, A., Sacco, N., Saeednia, M., 2012. One-way carsharing: Solving the relocation problem. Transportation Research Board 91st Annual Meeting.

Dinh T. L., 2016. Multiobjective Linear Programming- An Introduction. Springer International Publishing.

Fourer, R., Gay, D., Kernighan, B. W., 2002. The ampl book. Duxbury Press, Pacific Grove.

Halffmann, P., Krumke, S. O., Quilliot, A., Wagler, A. K., & Wegener, J. T., 2015. On the online min-wait relocation problem. Electronic Notes in Discrete Mathematics, 50, 281-286.

Jiang, S., Ong, Y. S., Zhang, J., Feng, L., 2014. Consistencies and contradictions of performance metrics in multiobjective optimization. Cybernetics, IEEE Transactions on, 44(12), 2391-2404.

Jorge, D., Correia, G. H. A., 2013. Carsharing systems demand estimation and defined operations: a literature review. European Journal of Transport and Infrastructure Research, 13, 201-220.

Jorge, D., Correia, G. H. A.,Barnhart, C., 2014. Comparing Optimal Relocation Operations With Simulated Relocation Policies in One-Way Carsharing Systems. IEEE Transactions on Intelligent Transportation Systems, 15(4), 1667--1675.
Jorge, D., Barnhart, C., & de Almeida Correia, G. H., 2015. Assessing the viability of enabling a round-trip carsharing system to accept one-way trips: Application to Logan Airport in Boston. Transportation Research Part C: Emerging Technologies, 56, 359-372.
Lee, J. and Park, G. L., 2013. Planning of relocation staff operations in electric vehicle sharing systems. In Lecture Notes in Computer Science, 7803, 256-265. Springer Science + Business Media.
Gambella, C., Malaguti, E., Masini, F., Vigo, D., 2017. Optimizing relocation operations in electric car-sharing. Omega, 1-12. DOI: 10.1016/j.omega.2017.11.007.
Marra, F., Yang, G. Y., Træholt, C., Larsen, E., Rasmussen, C. N., You S., 2012. Demand profile study of battery electric vehicle under different charging options. In Power and Energy Society General Meeting, 2012 IEEE, pp.1-7.
Nourinejad, M., & Roorda, M. J., 2015. Carsharing operations policies: a comparison between one-way and two-way systems. Transportation, 42(3), 497-518.
Nourinejad, M., Zhu, S., Bahrami, S., & Roorda, M. J. (2015). Vehicle relocation and staff rebalancing in one-way carsharing systems. Transportation Research Part E: Logistics and Transportation Review, 81, 98-113.
Repoux, M., Boyaci, B., & Geroliminis, N.,2014. An event-based simulation for optimising one-way carsharing systems. In Presentation at the 14th Swiss Transport Research Conference, Monte Verità/Ascona, Switzerland.
Riquelme, N., von Lucken, C., Baran, B., 2015.Performance metrics in multi-objective optimization. Computing Conference (CLEI), 2015 Latin American. IEEE.
Santos, G., Correia, G. (2015). A MIP model to optimize real time maintenance and relocation operations in one-way carsharing systems. Transportation Research Procedia, 10, 384-392
Schneider, M., Stenger, A., Goeke, D., 2014. The Electric Vehicle Routing Problem with Time Windows and Recharging Stations. Transportation Science, Published on-line in Articles in Advance 06, pp.500-520.
Schott, JR., 1995. Fault tolerant design using single and multicriteria genetic algorithm optimization.Thesis(MS)—Massachusetts InstituteofTechnology, Department of Aeronautics and Astronautics.
Van Veldhuizen, D. A., Lamont, G. B., 2000. On measuring multiobjective evolutionary algorithm performance, in 2000 Congress on Evolutionary Computation, IEEE Service Center: Piscataway, New Jersey, 1, 204–211.
Zitzler, E., Thiele, L. 1999. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. Evolutionary computation, IEEE transactions on, 3 (4), 257–271.