Free Floating Electric Car Sharing Design: Data Driven Optimisation

Michele Cocca[a], Danilo Giordano[b], Marco Mellia[a], Luca Vassio[a]

[a] *Department of Electronics and Telecommunications, Politecnico di Torino, Italy.*
[b] *Department of Control and Computer Engineering, Politecnico di Torino, Italy.*

Abstract

In this work we consider the design of a Free Floating Car Sharing (FFCS) system based on Electric Vehicles. We face the problems of finding the optimal placement of charging stations, and the design of smart car return policies, i.e, how many and where to place charging stations, and whether to ask or not customers to return the car to a charging pole.

We leverage actual data containing rentals performed by Car2Go customers. We obtain information for several months worth of actual trips in the city of Turin, our use case. Via trace driven simulations, we replay the exact same trips while simulating electric car based FFCS, to accurately gauge battery discharging and recharging. With this, we compare different charging station placements, also driven by optimisation algorithms. Moreover, we observe the impact of collaborative or selfish car return policies.

Results are surprisingly: just as few as 13 charging stations (52 poles) guarantee a fleet of 377 vehicles running in a 1 million inhabitant city to work flawlessly, with limited customer's discomfort. We believe our data driven methodology helps researchers and car sharing providers discerning different design solutions. For this, we make available all data and tools to foster further studies in these directions.

Keywords: car sharing, electric vehicle, data driven optimisation, charging station placement, free floating

*This document is an extension of our previous work [1] presented at IEEE SMARTCOMP 2018.

Email addresses: michele.cocca@polito.it (Michele Cocca), danilo.giordano@polito.it (Danilo Giordano), marco.mellia@polito.it (Marco Mellia), luca.vassio@polito.it (Luca Vassio)

1. Introduction

Mobility and pollution are challenging problems in our cities. Private vehicles, still important urban transportation means, are among the major contributors to both congestion and air pollution. Because of this, smart and shared mobility are seen as a key component to reduce emissions and traffic [2]. Given a fleet of cars, Free Floating Car Sharing (FFCS) systems allow customers to pick and drop a shared car everywhere inside an operative area, thus reducing the number of private cars and increasing the number of available parking spots. The conversion from internal combustion cars into Electric Vehicles (EVs) is seen as the next big opportunity to drastically reduce pollution inside urban areas [3]. However, the design of a system based on electric cars entails the deployment of a charging station network [4].

In this work, we tackle the design of an electric FFCS system. This is a challenging problem, given charging constraints which impact car availability, and the cost of the infrastructure setup and maintenance. The design of the charging station infrastructure requires thus ingenuity to maximise customers' comfort, and minimise cost for the operator [5, 6, 7].

Two are the main problems that need to be faced: i) the charging station placement problem, i.e., how many and where to install charging stations; and ii) the return policy customers have to follow at the end of the rental, i.e., in which cases to ask the customer to return the car to a charging station. In this paper we face both the above problems. Notice that the number of charging stations is directly related to system installation costs.

We strongly believe actual usage data is fundamental to answer these questions. While in the past some works have proposed solutions for the design of electric FFCS [3, 8], we are among the first to take a complete data-driven approach [5, 6, 7, 9, 10, 11] in an electric FFCS. Here, we extend our preliminary works [1, 12] by presenting a more extensive evaluation, and introducing two search algorithms.

We start by describing the system we implemented to collect real data from the FFCS systems currently in use in the city of Turin (Italy), which we consider as a test case. Despite being based on internal combustion engine cars, this data perfectly captures the actual usage patterns of regular customers [13]. This naturally factors the desired origin and destination of trips and the time

varying demand, including special events such as sport matches or strikes.

We next leverage the data collected from more than 2 months of rentals in 2017. First, we characterise how customers actually use the FFCS, in terms of rental/parking event duration, and of the origin/destination of hundred of thousand trips. Then, we develop a flexible event driven simulator that replays the exact same events recorded in the traces to accurately mimic actual customers' habits. It simulates the usage of each EV, its battery consumption and charging, while considering different design parameters. With this, we run thorough simulations to understand the implication of the design choices, such as charging station placement algorithms, and car return policies.

Results show that placing the charging stations in those areas where cars stay parked for long periods performs worse than a totally agnostic random placement. Instead, placing charging station in those areas where cars are frequently parked even for short periods guarantees better performance.

Next we gauge the benefits of considering collaborative car return policies, where customers voluntarily or forcibly return the car to a charging station in case the battery level decreases below a threshold like the authors of [14] proposed. This kind of return policies are inspired by the user-relocation model presented in [15, 16, 17, 18], where the relocation is driven by the presence of parking and charging stations and not by the demand areas. We observe that this halves the number of charging stations required to sustain the system. However, this increases customer's discomfort, in terms of number of times customers have to return the car to a charging station, at the cost of additional distance from their desired final destination.

To solve this tension, we further optimise the placement of the charging station by means of global optimisation algorithms. We implement and validate two algorithms: a hill-climb local search and a genetic algorithm, both tuned to minimise system cost and customer's discomfort.

Results are surprising: just equipping 5% of the city area with charging stations guarantees the system to self sustain, with no cars ever running out of battery in two months of trips. This corresponds to install only 13 charging stations in the whole city of Turin, which has 1 million inhabitants. Furthermore, the placement found by the genetic algorithm guarantees only 4% of

re-routing events, with the customers parking the car, on average, within 90 m from their desired final destination.

We believe results presented in this paper, guided by actual usage pattern for FFCS customers, are very important for regulators, policy makers, car sharing providers, as well as for researchers working in this area. Our data driven approach provides novel opportunities to guide the design of electric car sharing system, where the realistic figures provided by data allow investigating solutions that meet both customer requirements and limit system costs.

In an effort to allow reproducibility and extend our results, we make both the data set and the simulator publicly available as open source [19].

After discussing related work in Section 2, we present our methodology to collect data and characterise our data-set in Section 3. In Section 4 we describe the simulation model, its parameters and metrics of interest. Section 5 presents the different placement heuristics and the algorithms we design to optimise the placement. Section 6 discusses the impact of simple charging stations placement policies and return policies, while Section 7 reports the results ofthe optimisation and their validation. Section 8 discusses limitations and future work, before drawing conclusions in Section 9.

2. Related work

The diffusion of the free floating approach to car sharing led to an increasing attention by many researchers, with many analyses of these systems and their extension to electrical vehicles. The studies performed in 2011 by Finkorn and Müller [2, 20] are the first attempts to analyse benefits of FFCS for the population. Their results on customers' characterisation, like travelled distances and rental duration, are similar to ours. Later works [21, 22, 23] also collected data and analysed the mobility pattern of customers and differences among cities. While providing insights on usage patterns, these works do not discuss the implications on Electric Vehicles based FFCSs. We also introduced UMAP [13] - a system to harvest data by crawling FFCS websites. Here we use traces collected with UMAP to drive our system design.

The introduction of EVs for private and public transportation brought the problem of the

design of the electric charging station infrastructure. After a survey among FFCS customers in Ulm (Germany), authors of [3] investigated the positive influence and feasibility of an electric FFCS systems. Authors in [9] show the benefits of placing charging stations with different capacity according to the car parking duration. Authors of [24] presents a simulation study similar to ours, but using random models to generate random trips rather than actual traces. Their algorithms tend to place charging stations along frequently used streets, so to let drivers top up the battery in 10 minutes.

Few data driven studies address the charging station placement, by respectively minimising cost of installation, power loss and maintenance [1, 5, 7], or by minimising the customers' walked distances necessary to reach a charging pole [6]. In [5], authors study the impact on the power distribution grid, with limited focus on FFCS performance. Authors of [7] instead focus on charging station design to minimise customers' anxiety. In our previous work [1], we presented a study of charging station placement based on actual data. Here we build upon this work, and present a more in depth study that includes global optimisation algorithms, never considered before.

Other works focus on station-based car sharing systems. Authors of [10] present algorithms to place the parking stations in a two-way scenario. They consider a combustion engine fleet, and solve the problem by considering real data from operative car sharing systems. The same authors propose a similar methodology considering a one-way scenario and electric vehicles [11]. Here they use synthetic data and other socio-economic information to estimate the demand. Both works are similar to our in spirit, but are limited to station-based car sharing.

Considering return policies, an interesting data driven research is presented in [14]. The authors focus on station-based car sharing system with electric vehicles, finding that the best policy is to charge a car only when its state of charge goes below to a minimum threshold. This is similar to the return policies we consider in this paper.

Other works focus on FFCS with EVs to study the revenue considering a demand-supply scenario for energy [25], introducing policies to free charging stations when occupied by fully charged cars [8], maximising revenue by moving cars in areas of high-demand [18], or providing incentives to customers to balance fleet [15, 16, 17]. This works are orthogonal to our.

To the best of our knowledge, we are the first to take a completely data driven approach for designing an electric FFCS system by optimising different metrics impacting customer experience.

3. Data collection and characterisation

Obtaining mobility data is fundamental in the design of transport systems. Nowadays, the diffusion of ICT technologies makes collecting actual usage data much simpler. Here, we present our approach to collect data from currently running FFCSs. For this, we designed and engineered *UMAP* [13]. It is composed by three modules: a data acquisition, a data normalisation and integration, and a data characterisation and analysis module. *UMAP* is freely accessible as open source at [19].

3.1. Data Acquisition

The first step consists in the data acquisition from the FFCS platform of interest. Each platform exposes information through a web-service that lets customers find positions of cars when available for rental. *UMAP* implements crawler-modules that harvests these web-service to collect, at periodic time instant, which cars are available. For now, we have developed crawlers for two FFCS providers, Car2Go and Enjoy, both offering services in Italy. For Car2Go, we rely on their public APIs [1], while for Enjoy we implemented a custom crawler by reverse engineering API. In both cases, the system returns the currently available cars using a JSON document, for each city in which they are present. *UMAP* takes periodic *snapshots* describing which cars are parked and ready for rental.

In a nutshell, a car is described as an object annotated by information like plate, vehicle identification number (VIN), fuel level, model, parked location, etc. This is instrumental for the customers, e.g., to choose which car to rent. This object is only present if the car is available, i.e., it is parked and free for a rental. When a customer reserves and rents it, the object disappears, and reappears later when the customer ends the rental, likely in a different location.

[1] Car2Go API, `https://www.car2go.com/api/tou.htm`, service subject to approval by Car2Go. Approval granted in September 2016, service disconnected in January 2018.

UMAP takes a snapshot $S(t)$ every minute, a reasonable time resolution to capture car rental and return events, while avoiding overloading the servers. In particular, we extract the *VIN* or *plate* field to uniquely identify a car, the parking *coordinates* reported with *longitude* and *latitude* by the in-car GPS, and the *fuel* level of the car.[2]

We started collecting data with *UMAP* in December 2016, and we have collected more than a year of data in all the 22 and 5 cities where Car2Go and Enjoy offer service, respectively.

3.2. Data Normalisation and Integration

In this second module we process and consolidate each snapshot $S(t)$ to obtain *booking* and *parking* periods for each car. The intuition is to track the availability of each car, and rebuild the history of parking and booking periods over time: when some customer books a car at time $[t-1, t)$, the latter "disappears" from the system starting from $S(t)$. We identify this events, and record it by creating a new booking, and setting the start time ($t_s = t$) and the start position. When the customer ends the booking, the car "reappears" in the system in $S(t2), t2 > t$. We record this event, by setting the end time ($t_e = t2$) and end position of the booking. After that, for the same car, a new parking period starts, hence we record a new parking event with its initial time ($t_s = t2$) and its position. When the car "disappears" again in $S(t3), t3 > t2$, we record end time ($t_e = t3$) of the parking event, and we start recording a new booking.

Next, we define a procedure to filter bookings and to obtain actual *rentals*. The FFCS allows customers to reserve a car prior the actual rental starts. In case the customer cancels the reservation, the car becomes available, creating a booking in our data-set, for which no rental actually ever happened. Conversely, some bookings last for several days or weeks, e.g., due to a car going offline, or under repair. At last, the back-end system may sometimes fail, generating spurious bookings. *UMAP* filters these artefacts to obtain actual rentals: Initial and final positions must be at least 500 m far apart, booking duration must be greater than 3 minutes, and shorter than 1 hour.

At last, we need to estimate the possible driving path and length from the knowledge of only the initial and final position. Euclidean distance between starting and ending coordinates of the

[2]The GPS coordinates are only available if a car is parked and available. There is no risk for users' privacy during rentals. In addition no user's identifier is exposed. Therefore data is totally anonymized as there is no means to know who booked a car.

trip represents a lower bound of the real driven distance, since cars have to follow the topology of the city and traffic laws. To estimate the real driving distance, we apply a corrective factor, that we obtain again from data. In more details, given a rental, we use the information returned by the Google Directions API related to the best driving path from the initial to the final position of the rental. [3] Then, we compute the ratio between the returned driving distance, and the euclidean distance. We collect this information at time the trip end is observed.[4] We repeat this for about 10 000 trips, observing the distribution of the ratio which ranges between 1 and 2, with most of values around the median value of 1.4. We use the latter value as corrective factor to obtain the driving distance from the euclidean distance.

In this work, we consider 8 weeks between September and November 2017 for the Car2Go provider to optimise the charging station placement. We collected about 190 000 bookings that generated 125 000 actual rentals from the 377 cars of the Car2Go fleet in Turin. To verify the robustness of the solution, we use other two traces collected in June-July 2017 and December 2017-January 2018.

Our data takes into account all the rentals, also in presence of special events generating a higher demand (like strikes, sport events, concerts, etc.). Therefore our data captures both normal daily traffic patterns, and anomalies.

In the following, we provide a quick characterisation of the data, which is instrumental to guide the design of the charging station placement algorithms.

3.3. Data characterisation

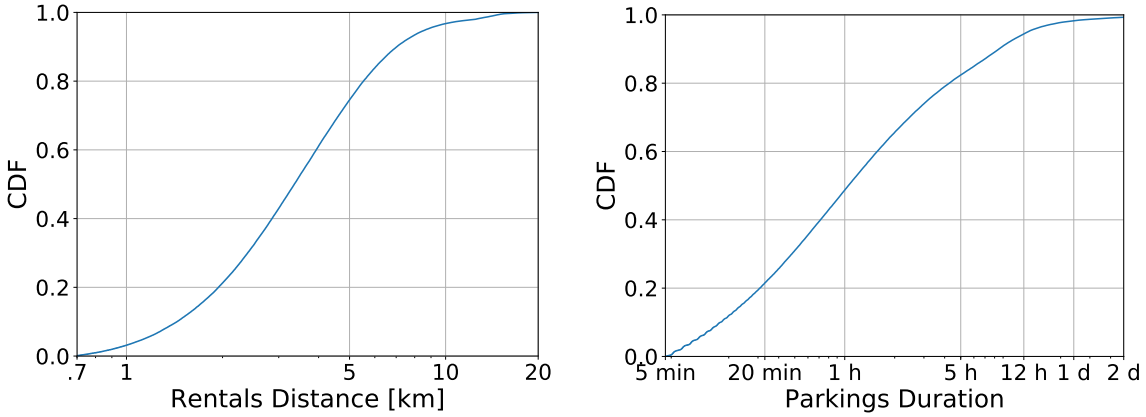
First focus on the characterisation of the (estimated) distance travelled during rentals. This figure is interesting since it is directly related to the minimum amount of charge an electric car shall have to complete a trip. Fig. 1(a) shows the empirical Cumulative Density Function (CDF) of the distances covered over all rentals. 97% of the trips covers less than 10 km, roughly corresponding

[3] `https://developers.google.com/maps/documentation/distance-matrix/`, freely available for a limited number of queries.

[4] Google Directions API factor eventual traffic congestion at different time of the day. As such, we include its impact in our model.

to the operative area diameter in Turin. The remaining 3% are trips to and from the airport, up to 19 km far away.[5]

Next, we investigate the parking duration. This figure is interesting since it is related to the amount of charge a parked car obtains when attached to a charging station. Fig. 1(b) illustrates the empirical CDF of parking duration. Interestingly, more than half of the parking events lasts less than 1 hour. This is due to the high utilisation of cars in FFCS, especially during business hours. Conversely, 10% of the parking events lasts more than 8 hours, with some cars that are left parked for days. The former are probably due to overnight parkings, while the latter hint for some cars parked in areas with few customers.



(a) CDF of travelled distances. X-axis is logarithmic.

(b) CDF of parking durations. X-axis is logarithmic, and limited to 2 days.

Figure 1: Characteristics of the trips in our data-set.

Next, we analyse how parking habits are different in the city area. For this, we divide the service operative area using a grid of squared zones of 500x500 meters, obtaining 261 zones covering Turin. For each zone, we compute the total number of parkings recorded and the average parking time.

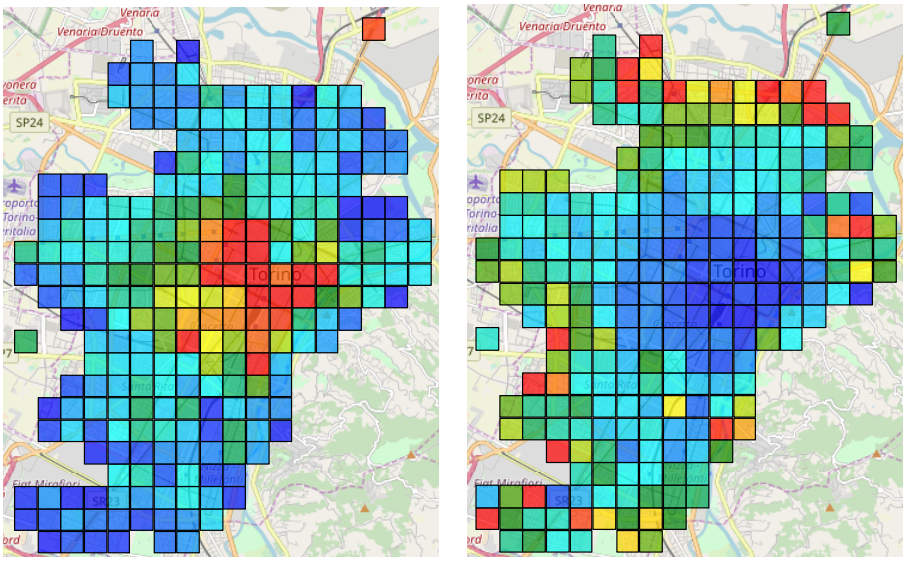
Fig. 2(a) shows the heatmap of the total number of parkings in each city zone. The warmer the colour is, the more frequently cars are parked here. The hot areas correspond to the city centre which exhibits the highest number of parkings. Customers rely on car sharing for travelling and moving downtown, a working area full of shops and restaurants. The zones with more parkings

[5]The Turin airport has a reserved parking for Car2Go vehicles. It is about 15 km far away from the city centre. Distances for trips to the airport, reached by a straight highway, are not corrected.

are close to the train stations, where 47 parking events per day are observed on average. On the contrary, few parkings are observed in the suburbs (down to less than 1 event per day), where people likely return home in the evening [13].

Fig. 2(b) shows the heatmap of the average parking time for each zone. Peaks are on borders of the operative area, where parking events last more than 24 hours. Few cars reach these peripheries and rest unused for long time (see also rightmost part of Fig. 1(b)). The lower values are registered in the downtown, where cars stay parked only for 85 minutes on average.

The large spread of parking density and duration challenges the decision on where to place charging stations. Indeed, if placed in areas where cars are frequently parked but for short time (e.g., city centre), batteries would get little charge. If placed in areas where few cars stay parked for long time (e.g., suburbs), cars will be fully charged cars but occupying the station for long time.



(a) Number of Parkings. (b) Average Parking Time.

Figure 2: Heatmaps showing (a) number of parkings per zone and (b) average parking time per zone. Warmer areas have larger values.

4. Electric car sharing simulator

Our goal is to study different design choices for electric car sharing systems, based on collected data. For this, we developed a flexible event-based simulator that allows us to compare different

algorithms and tune parameters while collecting metrics of interests.

We simulate a fixed fleet of electric cars. Each car is characterised by its parking location, and the current status of battery charge. The simulator takes as input the pre-recorded data-set of rentals, i.e., the trace, characterised by the start and end time, and initial and final geographic coordinates. For simplicity, space is divided into 261 zones of 500 x 500 m each (as explained in Section 3).

4.1. Trace event processing

Each recorded rental reflects a mobility interest of a customer, i.e., a desired trip. In more details, each trip $i \in \mathcal{I}$ is characterised by its start and end time, $t_s(i)$ and $t_e(i)$, and origin and destination coordinates, $o(i)$ and $d(i)$. We associate each position to the zone $O(i) = zone(o(i))$ and $D(i) = zone(d(i))$. We assume a charging station cs, composed of k poles, can be placed at the centre of a given zone $z \in \mathcal{Z}$, so either $cs(z) = 1$ if the station is present, or $cs(z) = 0$ otherwise. $N = \sum_{z \in \mathcal{Z}} cs(z)$ is the total number of zones equipped with charging stations, with $K = N \cdot k$ the total number of poles.

We have a set $\mathcal{A}$ of cars, where each car $a \in \mathcal{A}$ at time t is characterised by its position $p(a,t)$, its zone $P(a,t) = zone(p(a,t))$, and the residual battery capacity $c(a,t) \in [0, C]$, with C being the maximum nominal capacity.

The simulator processes each rental event i in temporal order. When a car rental start event i at time $t = t_s(i)$ is processed, the (simulated) customer looks for a car in the initial position zone $O(i)$. If cars are present, the customer rents the most charged one, independently whether the car is at a pole being charged or not.[6] In formulas, we get a car $\bar{a} \in \mathcal{A}$ such that:

$$c(\bar{a}, t) \geq c(\hat{a}, t) \ \forall \hat{a} \in \arg\min_{a \in A} dist(O(i), P(a,t)).$$

If no car is present, the customer walks to the closest zone containing an available car, mimicking the normal behaviour of FFCS customers that look for the closest car to rent on their smartphones. A car rental end event is then scheduled using the trace final time

[6] We choose this policy because people are worried about vehicle range [26].

$t_e(i)$ and desired destination location $d(i)$. When car a rental end event at time $t_e(i)$ is processed, the customer returns the car in $p(a, t_e(i))$, chosen according to the behaviour described in the next paragraph. The simulator updates the battery charge status by consuming an amount of power proportional to the trip distance:

$$c(a, t_e(i)) = \max\left(c(a, t_s(i)) - Energy(p(a, t_s(i)), p(a, t_e(i)))\right)$$

with $Energy(\cdot)$ that models the energy necessary to go from the car origin $p(a, t_s(i))$ to the car destination $p(a, t_e(i))$. Here we consider $Energy(\cdot)$ to be dependent only on the two positions and proportional to their distance, but more complicated functions can be easily implemented.

In case the battery level drops below 0 ($c(a, t_e(i)) = 0$), the trip i is declared *infeasible*. The discharged car still performs further trips, all marked as infeasible, until it reaches a charging station.[7]

Depending from the return policy, the customer may connect the car to a charging pole. We investigate the following return policies:

- *Free Floating*: the customer opportunistically connects the car to a charging pole if and only if it is available (present and free) in the final desired zone $D(i)$;
- *Needed*: cars are connected to a pole only when the battery level at the end of the rental goes below a minimum percentage threshold α, i.e., $(c(a, t_s(i)) - Energy(p(a, t_s(i)), d(i)))/C \leq \alpha$. This implies the customer can be *rerouted* to the closest zone with an available free charging pole, if none exists in the desired final zone $d(i)$;
- *Hybrid*: the customers follow the Needed policy, but voluntarily connect to a charging pole – if available – in the desired ending zone, whatever car charge status is;

The *Free Floating* policy never obliges the customer to bring the car far from the desired ending location, even in case battery charge is close to exhaustion. *Needed* mandates to connect cars to a charge station only if battery runs low, thus trying to protect from battery exhaustion. *Hybrid*

[7] This is instrumental to give an exhausted car the chance to recover energy.

mixes the two policies letting customers opportunistically recharge the battery whenever they park close to a charging station.

Notice that policies similar to *Needed* have been introduced in [14], where the system make the users charge the car considering the battery state of charge, the instantaneous electricity cost, and the user's range anxiety.

4.2. Performance metrics and parameters

We measure the following metrics, that we identify having influence in the customers' quality of experience:

- *InfeasibleTrips%*: percentage of infeasible trips due to completely discharged battery observed during the whole simulation;
- *Charges%*: percentage of trips where the customer connects the car to a charging pole, implying the burden to plug the car;
- *Reroutings%*: percentage of trips where the customers are rerouted to a zone different from their original destination because they are forced to charge the car;
- *WalkedDistance*: walked distance from the desired destination. This is considered non-zero both when the car is charged or rerouted. The walk distance when returning the car to a pole in the desired final destination is considered to be 150 m, i.e., the average distance from any point to the centre of a square of 500 m side;

Infeasible trips are critical, and the system shall be engineered so that they never happen. Other performance metrics shall be minimised. In addition to the above metrics, the simulator collects statistics about car battery charge level, and fraction of time a battery stays under charge.

The key design parameters that we focus on are (i) number of zones Z which are equipped with a charging station; (ii) the locations of charging stations within the city; (iii) adopted return policies.

We consider the following scenario: the fleet has a constant number of cars equal to 377 (the same as observed in the trace). Electric cars have the same nominal characteristics as the Smart

ForTwo Electric Drive, i.e., 17.6 kWh battery, for 135 km of range, with a discharge curve that is proportional to the travelled distance (12.9 $kWh/100\,km$).[8] Charging stations have 4 low power (2 kW) poles each. These are cheap to install and a good compromise between costs, power requested, and occupied road section. We model a simple linear charge profile (complete charge in 8 hours and 50 minutes in our case). For *Hybrid* and *Needed* policy we set the minimum battery charge threshold, α, equal to 25%. This is a precautionary approach, since the maximum travel distance is 19 km (Fig. 1(a)), corresponding to about 14% of the battery capacity. At last, the initial position of the cars, only affecting the initial transient, is randomly chosen.

Our Python simulator completes a single simulation including 125 000 rentals in less than 5 seconds.[9] To post-process generated results and extract aggregated data, we use PySpark[10] on a Big Data cluster of 30 nodes.

5. Strategies for charging station placement

The main objective of this paper is to assess what is the best charging station placement. Assuming that we have Z zones and N charging stations, there are $\binom{N}{Z}$ possible placement solutions, which makes it prohibitive to find exhaustively the optimal solution. For this reason we evaluate different approaches. The first one uses domain knowledge acquired by characterising the data about current usage to propose heuristics. The second instead, uses two different data-driven simulation-based optimisers.

5.1. Heuristic placements

Each zone z is assigned a likelihood l_z. We greedily choose the top Z zones, according to three likelihood definitions:

- *Average parking time*: l_z is the average parking duration in z as measured in the trace; the stations will be located in the areas where cars stay parked the longest time, hence with more probability to fill-up the battery during a charging event;

[8] `https://www.smart.com/uk/en/index/smart-electric-drive.html`

[9] We are able to run up to 40 simulations in parallel on a 40-core Intel Xeon processor with 128 GB of RAM, running Ubuntu 16.04 OS.

[10] `http://spark.apache.org/docs/latest/api/python/#`

- *Total number of parkings*: l_z is the total number of parking events recorded in i in the trace; hence the stations will be located in the areas with more parking events. Likely, more cars will be charged;

- *Random placement*: l_z is an independent and identical distributed random uniform variable. As such, recharging stations result placed at random over the city area;

The first two heuristics are driven by the intuition to place charging stations in those zones where cars are likely to be parked for long time, or frequently. The latter is presented as a baseline for comparison. Referring again to Fig. 2(a) and 2(b), the charging stations will be located in the zones with warmer colours, respectively for total number of parkings and average parking time policy.

5.2. Simulation based advanced optimisation strategies

Given the complexity of the optimisation problem and the humongous space of possible solution, we investigate the adoption of meta-heuristics, a class of global optimisation algorithms [27]. These algorithms explore the space of possible solutions in smart ways, looking for better solutions while avoiding getting trapped into local minima.

In our case, the evaluation of a solution requires the simulation of two months of rentals, which is performed in approximately 5 seconds on a high-end machine. It is then important to consider that we have limited resources in the choice of meta-heuristics to consider. The class of optimisation problems where the number of solutions (i.e., fitness evaluations) have to be limited as much as possible lies in the so called expensive-optimisation [27, 28]. In the literature, there are several architectures and algorithms suitable for global optimisation in tough numerical problems [28], with direct-search class algorithms that explicitly target expensive-optimisation problems.

In our work, we consider a simple local-search algorithm based on a hill-climb method, and a more complex and powerful genetic algorithm. We explicitly design both algorithms with the perspective of reducing the number of simulations. Both are implemented in our open-source tools [19].

We consider the single-objective case, i.e., algorithms have to minimise a single fitness function f defined as follow:

$$f = M \cdot InfeasibileTrips\% + WalkedDistance$$

As commonly done in linear programming, M is a number big enough to make the first addend always larger than the second. In this way, *InfeasibleTrips%* has to be minimised first. Secondly, the algorithms starts minimising the *WalkedDistance%*. In a nutshell, we look for solutions that make all trips feasible, and only then we target the customers discomfort, i.e., reducing the walked distance. As we will better show in the next section, reducing *WalkedDistance* will naturally help in reducing also *Charges%* and *Reroutings%*. Indeed, in its definition, the walked distance weights both these metrics.

5.2.1. Hill-climbing local search

Hill-climbing methods belong to the family of local search algorithms and are a popular choice because they are fast, simple to implement, and requires limited computational resources. They are iterative algorithms that start with an arbitrary solution, then attempt to find a better solution by making incremental changes to the solution. If the change produces a better solution, it is selected as current solution. Incremental changes are then made to the latter, until no further improvements can be found. For non-convex problems, like the one here faced, these methods will find only local optima, from which it would be impossible to escape. Local optima are not necessarily the globally best possible solution.

Our hill-climbing algorithm is very similar to the coordinate descent version [29]. We start from the best configuration found among the three heuristics. At each iteration, the algorithm randomly picks a charging station, and moves it in a empty neighbouring zone, i.e., north, south, east and west adjacent zones. All other charging stations are left untouched. If there is a direction of improvement, we perform a line search along the best direction, i.e., we keep moving the same station along the same direction. When no improvement is possible, we take another charging station at random, and try to move it as before. When no improvement is found after a complete cycle of all charging station, a local minimum is reached and the algorithm exits. We also stop the

local search after a maximum number of visited solutions.

In our implementation, we check multiple neighbours in parallel. Moreover, the algorithm keeps memory of all tested configurations, hence avoiding useless and expensive simulations. For details see [19].

In our experiments, on average the optimisation reaches convergence within 1 500 maximum tested solutions.

5.2.2. Genetic optimiser

Genetic algorithms are a particular class of evolutionary algorithms inspired by the natural evolution [30]. They are known to work well when dealing with discrete variable functions, as in our case.

Genetic algorithms start creating a random population of a given number of individuals. In our case, each individual corresponds to a random placement of charging stations. A mating pool is created from the initial population, then the offspring is generated by crossover and mutation operations. Crossover mixes the genes from different individuals picked at random, i.e., a child is created from the union of the genes of both parents. To keep the number of genes (i.e., charging stations) constant, random genes are removed so that at the end Z are left. Mutation instead moves a single charging station in a random empty zone. During crossover operation, some genes may mutate with low probability ($P\{mutation\} = 0.02$ in our case).

The presence of clones is avoided by the algorithm, which discards the copies, thus encouraging the exploration of the search space and saving precious resources. The algorithm estimates the quality of each new individual computing the fitness function f, i.e., by running an entire simulation.

As by natural selection, the best individuals survive to the next generation, while the worst individuals are suppressed. The optimisation loop continues until the maximum number of generations is reached (200 in our case).

If the diversity of available genes, i.e., the total number of distinct charging stations in the whole population, decreases too much without improvements, we increase the population genetic diversity by increasing the mutation probability ($P\{mutation\} = 0.2$ in this case). This randomises the

evolution. More details about the algorithm can be found in our open-source implementation [19].

The algorithm is amenably suitable for parallel implementation since the fitness of each individual belonging to a specific generation can be analysed separately from the others. Another advantage of genetic algorithms is the widespread exploration of the solution space, while local search algorithms tend to explore only limited portion of the space.

In our experiments, we set the initial population to 100, with 50 new individuals created at each generation by crossover. Most of the optimisations ended within 100 generations, i.e., 5 000 overall solutions are evaluated through simulation.

6. Impact of heuristic placements and return policies

6.1. Impact of heuristic charging station placements

In this first set of experiments we evaluate which heuristic placement performs better. Firstly, for each heuristic we consider the simple opportunistic *Free Floating* car return policy, i.e., customers always return the car in the desired zone, and connect it to a charging pole if available. The goal is to check if this simple return mechanism is sustainable with electric cars, and gauge the impact of the three different heuristics for the charging station placement.

Fig. 3 shows the performance in terms of infeasible trip percentage versus an increasing percentage of charging zones, ranging from 1% to 30%. In Turin, this corresponds to install from 2 to 78 charging stations overall (reported in the top x-axis). We observe notably different performances. First, the average parking time placement policy (*Avg time*) performs the worst, with still 6% of infeasible trips even when more than 25% of zones are equipped with charging stations. Even a simple random placement performs better (*Mean rnd*, obtained as the average of 10 independent instances). The total number of parkings placement (*Num parking*) reaches about 2% of infeasible trips at 10% coverage, reaching self-sustainability when more than 20% of zones are equipped (no infeasible trips).

The intuition of why such a striking difference is given by the different properties of areas where the heuristics place charging stations. *Avg time* placement favours peripheral zones where few trips ends, and where cars stay parked for long time (see Fig. 2(b)). On the contrary, *Num parking*

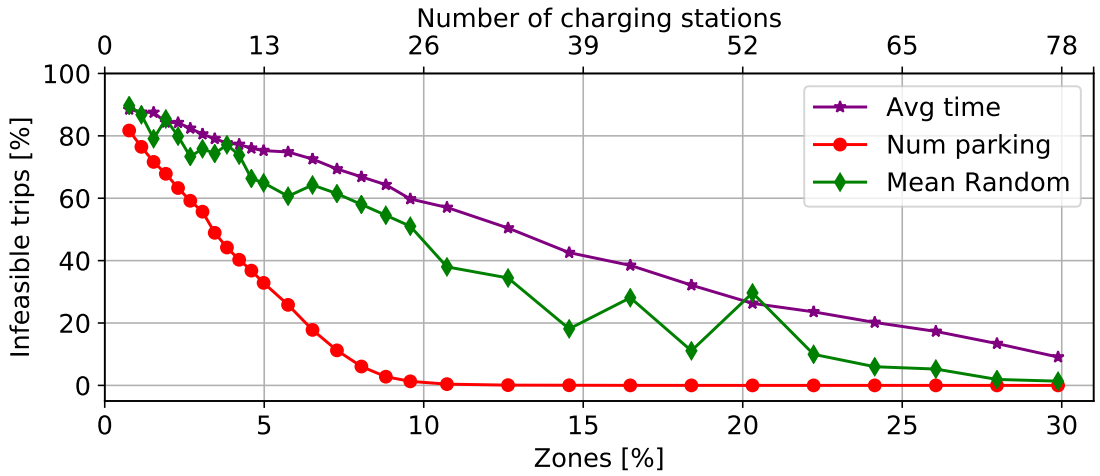


Figure 3: Percentage of unfeasible trips as function of charging stations (percentage and number of city zones), for the different heuristic placements.

favours city centre areas, where cars frequently are parked for short time (see Fig. 2(a)). Indeed, in the whole simulation, for $Z = 40$ ($\approx 15\%$ of zones), only 7 430 charges have been recorded for *Avg time*, compared with 47 628 charges of the *Num parking*. Even if plugged time is shorter, the *Num parking* policy allows the cars to charge the (little) energy consumed during the (short) trips. Moreover, as shown in Fig. 4 the *Avg time* placement generates much longer plugged times, often much longer than the time needed for a full charge. Therefore, many cars occupy the charging poles when they are already charged, preventing other cars to use those poles and increasing the number of infeasible trips.

In a nutshell the best approach among these three heuristics is to place charging stations in the central areas, in which the parkings last less but are more frequent. For this reason, we will use the *Num parking* placement algorithm as best heuristic for the remaining of the paper.

6.2. Impact of return policy

We now investigate the impact of the three different return policies. We quantify the implications of forcing customers to return the car to a different zone than the desired one, when the battery is below a critical level (i.e., below the percentage threshold $\alpha = 25\%$ of full battery capacity).

We focus again on the infeasible trip percentage with respect to the charging station coverage.

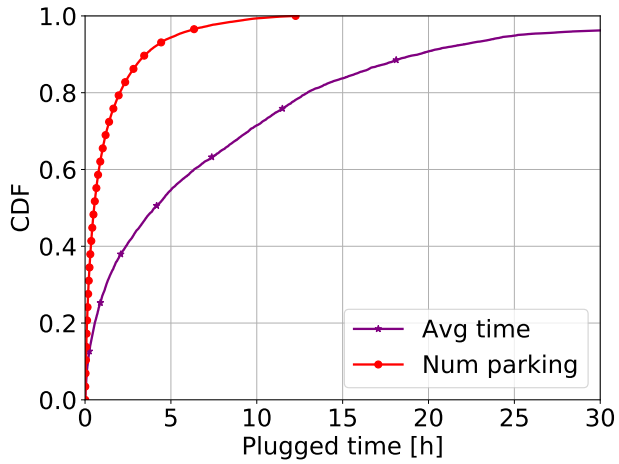


Figure 4: CDF of the time spent by a car at a charging station (Z=40), for *Num parking* and *Avg time* placement algorithms.

Fig. 5 shows results, with *Num Parking* placement. *Needed* and *Hybrid* policies perform much better than the opportunistic *Free Floating*. In details, *Hybrid* and *Needed* guarantee to successfully conclude all trips with just $Z = 11$ and $Z = 15$ charging zones, respectively (4.2% and 5% of the zones), while *Free Floating* reaches this goal only at $Z = 60$ (23% of zones). In a nutshell, adopting a policy which mandates customers to charge the cars when battery level gets low drastically reduces the number of infeasible trips, even with a handful of charging stations.

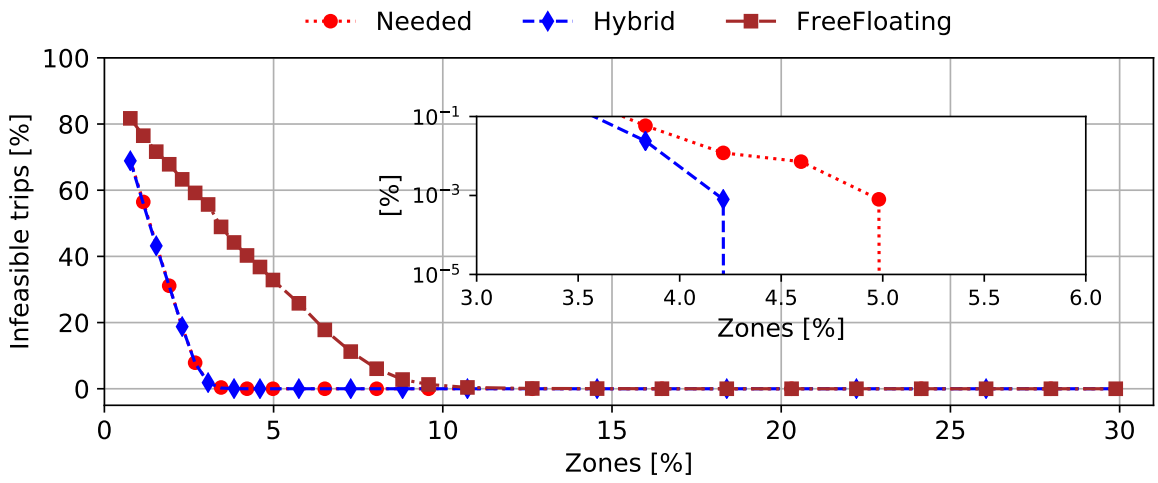
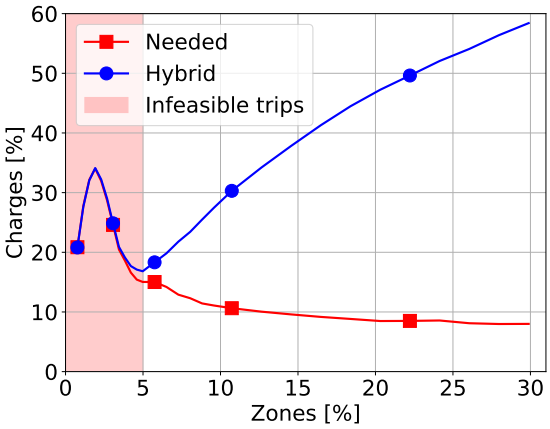
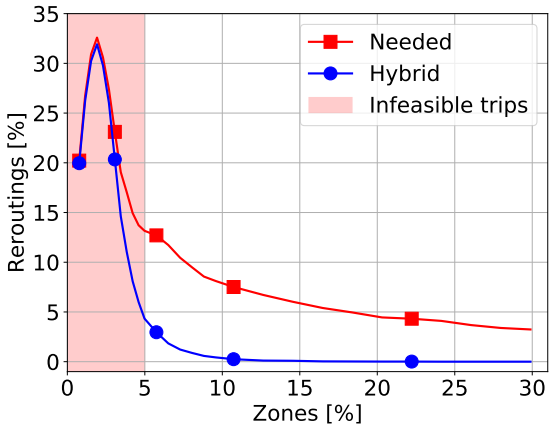


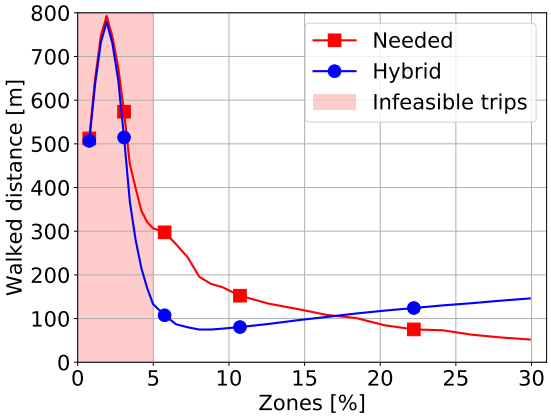
Figure 5: Percentage of infeasible trips for different zone coverage percentage analysing the return policies. The inset highlights where the infeasible trips go to 0.



(a) Percentage of trips ending with a charge.



(b) Rerouted trips percentage.



(c) Walked distance, averaged over all trips.

Figure 6: Metrics of interests when comparing Hybrid and Needed return policy (Max parking placement adopted).

6.2.1. Customers' discomfort

Forcing customers to charge batteries, even at cost or rerouting them has clearly an impact on their comfort. Indeed, forcing a customer to park in a charging station can be annoying, because of the burden of reach the charging station, and losing time to plug to and unplug the car from the pole. Even worse, rerouting customers to the closest zone for charging increases the distances they have to walk to reach the desired destination.

Fig. 6(a) reports the percentage of trips that end by charging the car. Shaded area highlights the infeasible region, where the lack of charging zones create artefacts. Focusing on the feasible region instead, the two curves start with similar values, but then they diverge. Interestingly, the

percentage of charges decreases for the *Needed* policy, getting as low as 8%. This suggests a better usage of each single charge, i.e., the battery fills up and does not need frequent charges. Conversely, the *Hybrid* policy increases steadily the fraction of trips where customers have to plug to a pole. This because the higher Z, the higher the probability of ending the trip in a zone with a free charging pole.

Fig. 6(b) represents the rerouting percentage in function of charging stations coverage. Rerouting probability decreases as expected: the more the stations are, the more likely customers find a charging station at their desired final zone. Yet, the two policies have different performance. The *Hybrid* policy is less likely to reroute customers. In fact, by opportunistically connecting the car to a charging pole if available, the average battery charge is high, thus decreasing the rerouting probability. With more than 7% of charging zones, the percentage of rerouting is already lower than 1% for *Hybrid* policy. In a nutshell, *Hybrid* policy significantly reduces the number of times the customer has to drive to a charging station in a different zone than the desired one. However, it increases the number of times the customer parks at a charging station and has to plug the car to the pole. Therefore, one must be cautious when weighting these results and designing return policies which impact the customers' comfort.

Whenever the system forces a customer to park the car in a charging station, customers may be routed far from their desired destination. If the charging station is not in the final destination zone, the customer has to walk by at least 500 m to reach their final destination. Even in case the charging station is found inside the final zone, the customer will have to park at the charging station, instead of the desired destination. For this reason, we evaluate the average distance the customers have to walk to reach their actual final destination. In details, when the customers suffer rerouting, we evaluate the actual distance between the charging station and the final destination. When they end in a charging zone and plug the car, we count an average distance of 150 m. Finally, if the customers do not charge the car, we assume they arrived at their final destination directly.

Results are shown in Fig. 6(c). Consider the feasible region. The *Needed* policy exhibits a decreasing trend (from 280 m to 60 m). On the contrary, the *Hybrid* policy first exhibits a decrease (minimum of 50 m at 8% of charging zones), but then it slowly increases till it overtakes

the *Needed* policy. This is due to the fact that with few charging stations (6-15%) the number of charges is limited ($< 30\%$, see Fig. 6(a)) by the availability of charging stations. Instead, when this number grow, the opportunistic *Hybrid* policy forces the customer to walk more times within the ending zone. To this extent, the *Needed* policy performs better.

Even if the values of walked distances seem very small, remember that they are averaged over all trips. Restricting to the few long walks due to rerouting (not reported for the sake of brevity), the walked distance that we observe ranges from 2 200 m to 1 500 m, with similar behaviours for the two policies. Recall that the charging station placement algorithm likely places stations mainly in the city centre. Therefore, the charging stations are concentrated in a small area, so that rerouting from the suburbs significantly affects the walked distance when rerouted. This gives hope for further optimisation, as we will see in the next section.

Given the very few rerouting of the *Hybrid policy*, one can envision a system which directly takes care of those very few cars that need a battery charge, i.e. by relocating vehicles. For instance less than 3 cars per day would need to be relocated with 15% coverage or higher.

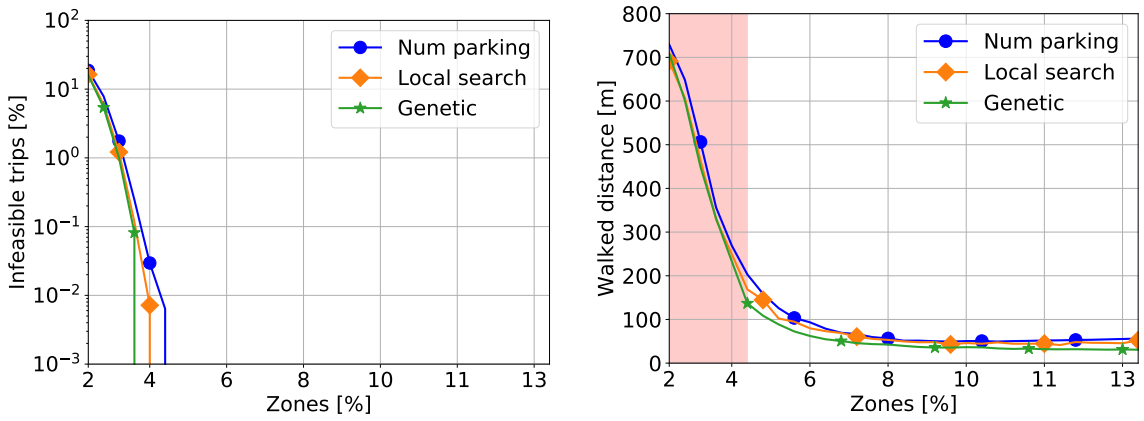
In summary, with *Hybrid* policy, less than 15% of zones guarantees all trips to be feasible, reduces the walked distance, asks for few rerouting events, at the cost of moderately high percentage of times (40%) customers are asked to charge the battery.

7. Meta-heuristic optimisation of the charging station placement

In the previous section, we saw how the *Num Parking* placement heuristic works better than the other two. Weighting also charges and rerouting, the *Hybrid* policy shows better performances than *Needed*. For this reason, in this section, we focus on the *Hybrid* return policy with charging stations covering less than 15% of the zones. We further optimise this scenario by running the meta-heuristic placement algorithms, and comparing the results with the *Num Parking* placement. Optimisation with *Needed* return policies are briefly discussed in Appendix A, where very similar results are obtained.

The hill-climbing local search, here abbreviated in *Local Search*, uses *Num Parking* placement as initial solution. The *Genetic* algorithm creates a totally new solution without exploiting any

previous knowledge. Recall that both algorithms are designed to find the best charging stations placement that guarantee 0 infeasibile trips, and to minimise the overall distance the customer has to walk to reach the final destination.



(a) Percentage of infeasible trips. Y-Axis is logarithmic.

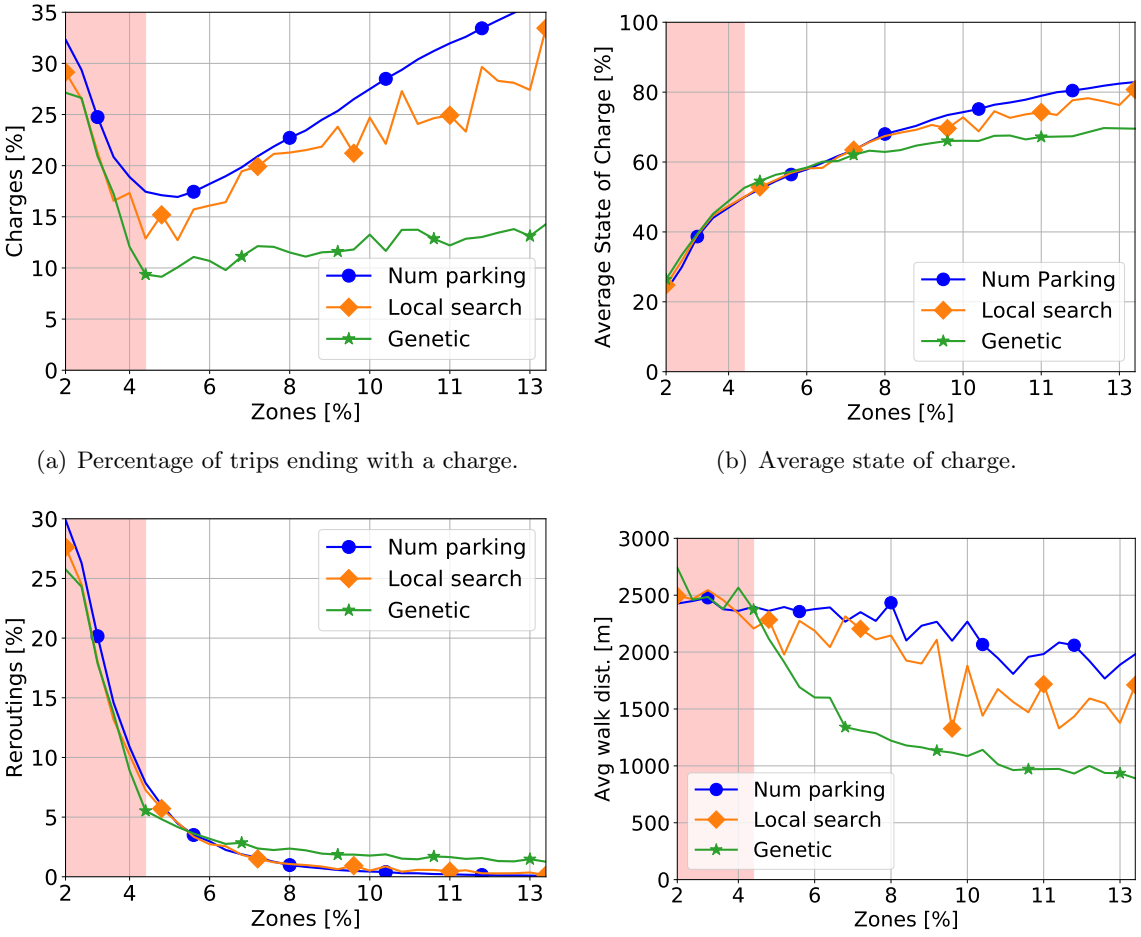
(b) Average walked distance.

Figure 7: Objective metrics to minimise in the optimisation - with *Hybrid* return policy.

Fig. 7 reports the two target metrics, for all the optimised configurations. Firstly, in Fig. 7(a) we compare the infeasible trip percentage. *Num Parking* solution (blue line) has already good performance, reaching 0 with 4.2% of the equipped zones. The *Local Search* (orange line) and the *Genetic* (green line) algorithms are able to further reduce the minimum percentage of zone to equip to guarantee no infeasible trips: 3.8% by the *Local Search*, and to 3.5% with the *Genetic* algorithm, $Z = 10$ and $Z = 9$ zones respectively.

Fig. 7(b) reports the walked distance. Focusing in the feasible region, the *Genetic* algorithm confirms the best performance, reducing the distance from more than 200 m to 136 m when 4.2% of the zones are equipped with charging stations, and reaching just 30 m at 13%.

In Fig. 8 we further study the new solutions on other metrics. Fig. 8(a) reports the percentage of trips ending in a charging station. The more charges are performed, the more time the customer has to spend time plugging/unplugging the car. By minimising the walked distance, we also reduce this metric, since a trip ending with a charge corresponds to 150 m of penalty. Here, the *Local Search* follows the same trend as the *Num parking* with a strong rise. The *Genetic* algorithm shows much better results, from 9% to 14% of trips ending with a charge – half of those found with other



(a) Percentage of trips ending with a charge.

(b) Average state of charge.

(c) Rerouted trips percentage.

(d) Walked distance when rerouted.

Figure 8: *Genetic* and *Local Search* optimisation results for metrics of interests (*Hybrid* return policy adopted).

solutions. This improvement highlights the better placement of the charging stations. Focus now on Fig. 8(b), which reports the average state of charge of the car battery. No major difference are observed here, with all curves almost overlapping up to 7% of the zones.

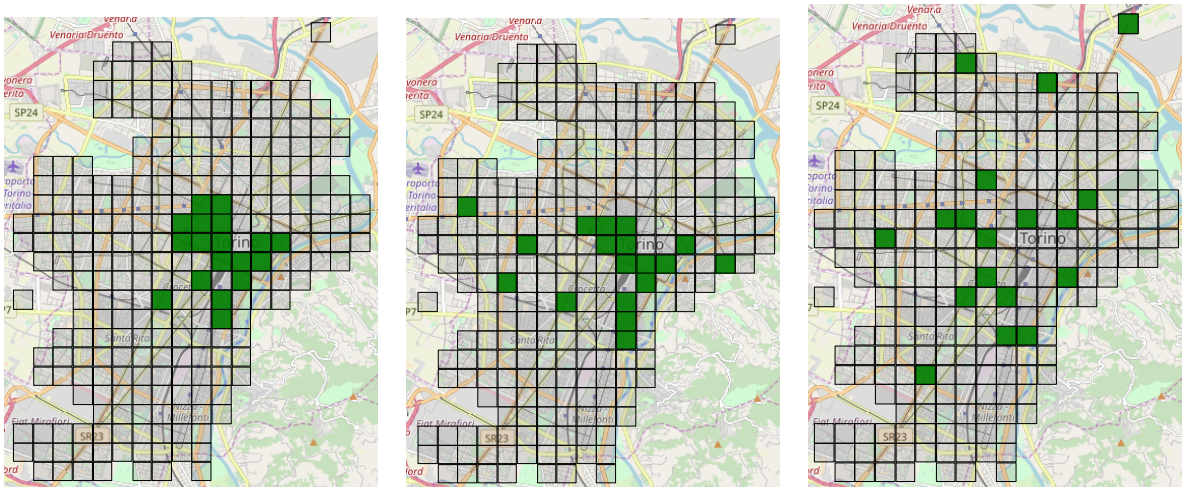
In a nutshell, the solution found by the *Genetic* algorithm lets customers charge much less frequently, while keeping the average state of charge very similar.

Consider next Fig. 8(c), which details the percentage of reroutes. We can see how the trend is the opposite with respect to the previous ones. Here, the *Genetic* optimised solution show a little higher rerouting percentage than the *Num parking*. In particular, the *Genetic* algorithm reaches 1.3% of reroutes, while *Num parking* decreases down to 0.2% of reroutes. Indeed, the *Genetic*

algorithm places charging stations not only where most rentals ends, but also so to decrease the customers' average walked distance, i.e., in those places where likely cars are not so frequently parked but that can be quickly reached in case of rerouting. To understand the importance of this difference, in Fig. 8(d) we evaluate the walking distance a customer has to walk because she suffered a reroute. The *Genetic* algorithm is able to push the walked distance below 1 km, while the *Local Search* generates marginal improvements. In a nutshell, despite customers are rerouted more frequently, on average, they walk for a shorter, and more bearable, distance.

In conclusion, a smart placement of the charging stations is better under different perspectives. The *Genetic* solution, tailored on the data of the usage behaviour, allows us to improve both the system performance, and customers' discomfort, in particular by greatly reducing the number of times they have to charge, and the distance they have to walk.

7.1. Charging station placement visualisation

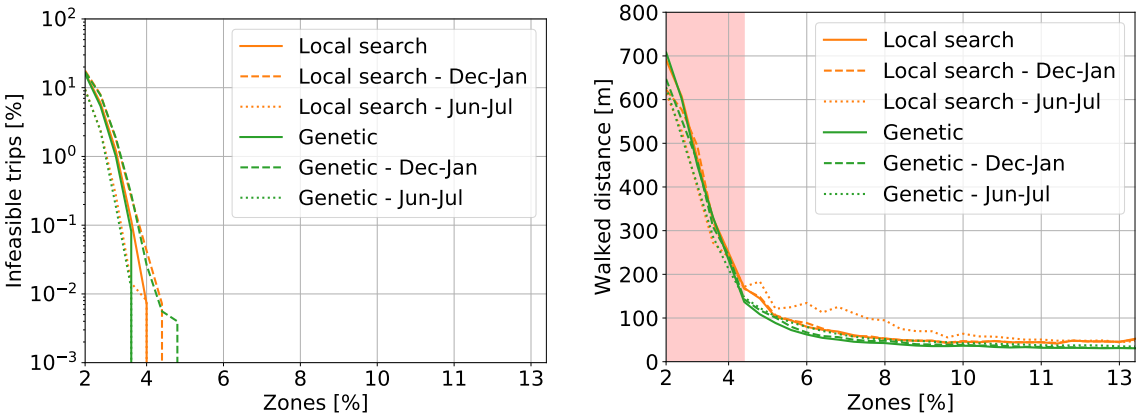


(a) Number of Parkings. (b) *Local Search*. (c) *Genetic* algorithm.

Figure 9: Different placement of 18 zones (7% of the total) for (a) Number of parkings per zone, (b) *Local Search* and (c) *Genetic* solution. Darker areas have larger values.

To give a feeling about the differences in the solutions found by different algorithms, Fig. 9 reports the solutions obtained with 7% of the zones equipped with charging stations (i.e., $Z = 18$ zones).

Num parking solution, Fig. 9(a), places most of the charging stations in downtown area and near



(a) Percentage of infeasible trips. Y-Axis is logarithmic.

(b) Walked distance, averaged over all trips.

Figure 10: Performance of the optimised configuration, tested on other 2 months long data-sets.

the main train stations. *Local Search*, Fig. 9(b), still has many zones in common with *Num parking*, the solution it started from. It just spreads some charging station to cover also some remote zones. The *Genetic* algorithm, Fig. 9(c), shows very few zones in common with *Num parking*. Charging stations are spread all over the city, still with more density in the city centre.

7.2. Validation of optimised configurations

The optimised solutions presented in the previous section are built through data-driven simulations. Hence they might over-fit the data of the specific considered period and not be robust to customers' habit changes. To validate our findings, we test the output placement configurations by using independent test traces, different from the one used to run the optimisation. We rely on two traces collected in Turin in two different periods of the year: one in summer, from June to July 2017; the other in winter, from December 2017 to January 2018. We focus on these two periods since in summer and near Christmas holidays the users may exhibit different habits (e.g., customers may rent the cars to go to parks and swimming pools during summer). These anomalies may represent a challenge for the optimised configurations. In the summer trace we record about 100 000 rentals, while in the winter one 128 000 rentals (respectively 8% less and 3% more with respect to the September/October trace). We compute the best station placement considering the September and October 2017 trace, and test system performance using the summer and winter

traces.

Fig. 10 compares results. We consider both *Local Search* and *Genetic* algorithms. In almost all cases, differences are negligible, showing that the solution is robust. For example, for 13% of zones and considering *Genetic* algorithm, the walked distance on the tests are just 2 m above those in the trace used for optimisation. Notice how in June and July the *Local Search* behaves worse than other cases: this is possibly due to the different mobility patterns in summer, while the Local Search solution could still be too related to the number of parkings in September-October. On the other hand, the solution found by the global genetic algorithm is robust.

8. Discussion and Implication

The mobility model we use for optimising the placement of charging station reflects customers habits. However, there are other aspects of car sharing system design that one could consider. In the following, we briefly discuss two of them.

*8.1. **Scalability***

We built the events trace from real rentals in the analysed city. By using this data we use simulations to study placement solutions which cope with the current amount of traffic and usage of the FFCS. While in a short term this solution is optimal, we have no guarantee whether it will be valid in the future in case of a strong increase in the car sharing usage, e.g., when popularity increases by orders of magnitude. To tackle this problem, we can still leverage rental data to infer a model about car sharing usage patterns in time and space, where the demand can be easily controlled by increasing the frequency of rentals. This model can then be used to create synthetic traces with an increasing car sharing demand, and use simulations to assess overall system performance. As such, the methodology we designed is generic and could be used to study different *What-If* cases.

Directly linked to the scalability problem, another important aspect is the possibility to add new charging stations when car sharing demand changes. By analysing the data collected by UMAP, our methodology can be used to consider the greedy placement of new charging stations on the top of those already present.

8.2. *Economical Aspects*

Economical aspects play too a key role in the placement decision process. In this work, we decided to study only the feasibility of the EV based FFCS system design by considering as few charging stations as possible, leaving for a next step the detailed cost estimation. The cost of the infrastructure creation and management can largely vary depending on different variables such as country, city, incentives. Related to this first class of costs, authors in [31] give a first estimation of installation cost, which can be of up to 5 500 USD per pole in the USA. While analysing these costs and developing a business plan, it is also important to evaluate which parties could be interested on running a business around it, e.g., either the municipality or a third party company could offer the infrastructure as a service to FFCS providers and other customers with electric vehicles.

The second group of variables consider the earnings models and the variable costs of the FFCS provider, like energy and cars. This kind of data requires a careful analysis to get a reliable estimation. Authors in [18] suppose a marginal profit of 75% of the fare considering a FFCS provider in the city of Vancouver. However, to correctly estimate the net profit several aspects need to be considered, such as the fee per minute, the actual duration of the rental costs and incentives for reroutes.

Given that, we are currently working on a more complete model to better study the economical impact of electrification by including in the simulator a precise cost model.

9. Conclusions

Designing an electric free floating car sharing systems leads to many interesting problems and trade-off between usability, costs and benefits for the customers. In this work, we built on actual rental traces to study via accurate simulations the impact of different charging stations placement and charging policies. We considered Turin as a case study, using 2 months of rentals recorded from a currently operational FFCS that we use to run trace driven simulations.

We have shown that few charging stations are enough to make the system self-sustainable. Important is the customers collaborations, so that they voluntarily returns to the cars to charging stations when available

Our data driven results show that just 5% of the city zones that are equipped with charging stations (13 in total, 52 poles) make all trips feasible with an electric car fleet. Moreover, through a charging station placement based on a genetic optimisation algorithm, it is possible to minimise the discomfort for the customers that would be (rarely) asked to bring the car for charging. For example, with 18 charging stations (72 poles in total), on average a customer would walk only 40 m to reach its desired destination. While these numbers will change in different cities, the data driven approach we propose naturally fits the global optimisation algorithm that is able to optimise placement while considering complex customers habits.

We leave for future work the simulations of scenarios with new technologies, such as deployment of faster charging poles and larger batteries, and the scalability in terms of number of customers and fleet size. We believe that our approach, based on data and accurate simulations is very promising to design and understand electric FFCS systems in future smart cities, provided actual data is available.

References

[1] M. Cocca, D. Giordano, L. Vassio, M. Mellia, Free floating electric car sharing in smart cities: Data driven system dimensioning, in: 4th IEEE International Conference on Smart Computing, 2018.

[2] J. Firnkorn, M. Müller, What will be the environmental effects of new free-floating car-sharing systems? the case of car2go in Ulm, Ecological Economics 70 (8) (2011) 1519–1528.

[3] J. Firnkorn, M. Muller, Free-floating electric carsharing-fleets in smart cities: The dawning of a post-private car era in urban environments?, Environmental Science & Policy 45 (2015) 30 – 40.

[4] M. C. Falvo, D. Sbordone, S. Bayram, M. Devetsikiotis, EV charging stations and modes: International standards, in: 2014 Symposium on Power Electronics, Electrical Drives, Automation and Motion, 2014.

[5] P. Phonrattanasak, N. Leeprechanon, Optimal placement of ev fast charging stations considering the impact on electrical distribution and traffic condition, in: 2014 Conference and Utility Exhibition on Green Energy for Sustainable Development (ICUE), 2014.

[6] T. Chen, K. Kockelman, M. Khan, The electric vehicle charging station location problem: A parking-based assignment method for seattle, in: 92nd Transportation Research Board Meeting, 2013, pp. 13–1254.

[7] Z. Jiao, L. Ran, J. Chen, H. Meng, C. Li, Data-driven approach to operation and location considering range anxiety of one-way electric vehicles sharing system, in: 8th International Conference on Applied Energy, 2016.

[8] S. Weikl, K. Bogenberger, A practice-ready relocation model for free-floating carsharing systems with electric

vehicles mesoscopic approach and field trial results, Transportation Research Part C: Emerging Technologies 57 (2015) 206 – 223.

[9] A. Hess, F. Malandrino, M. B. Reinhardt, C. Casetti, K. A. Hummel, J. M. Barceló-Ordinas, Optimal deployment of charging stations for electric vehicular networks, in: Proceedings of the First Workshop on Urban Networking, 2012, pp. 1–6.

[10] T. Rickenberg, A. Gebhardt, M. Breitner, A decision support system for the optimization of car sharing stations, ECIS 2013 - Proceedings of the 21st European Conference on Information Systems.

[11] S. Marc, K. Kathrin, M. Breitner, A decision support system for the optimization of electric car sharing stations, 2015.

[12] M. Cocca, D. Giordano, M. Mellia, L. Vassio, Data driven optimization of charging station placement for ev free floating car sharing, in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018, pp. 2490–2495. doi:10.1109/ITSC.2018.8569256.

[13] A. Ciociola, M. Cocca, D. Giordano, M. Mellia, A. Morichetta, A. Putina, F. Salutari, UMAP: Urban mobility analysis platform to harvest car sharing data, in: Proceedings of the IEEE Smart City Innovations 2017, 2017.

[14] C. Flath, J. Ilg, C. Weinhardt, Decision support for electric vehicle charging, in: Proceedings of the 18th Americas Conference on Information Systems (AMCIS 2012), Seattle, Washington, USA, 9 - 12 August 2012. Association for Information Systems, 2012.

[15] A. Brendel, J. Tim Brennecke, P. Zapadka, L. M Kolbe, A decision support system for computation of carsharing pricing areas and its influence on vehicle distribution.

[16] A. Brendel, S. Lichtenberg, I. Nastjuk, L. M Kolbe, Adapting carsharing vehicle relocation strategies for shared autonomous electric vehicle services, 2017.

[17] A. B. Brendel, C. Rockenkamm, L. M. Kolbe, Generating rental data for car sharing relocation simulations on the example of station-based one-way car sharing, in: HICSS, 2017.

[18] S. Wagner, C. Willing, T. Brandt, D. Neumann, Data analytics for location-based services: Enabling user-based relocation of carsharing vehicles, in: ICIS, 2015.

[19] UMAP and electric car sharing simulator. Anonymized datasaset of 2 months of trips of car sharing users in the city of Turin, https://github.com/michelelt/sim3.0.

[20] J. Firnkorn, M. Müller, Selling mobility instead of cars: New business strategies of automakers and the impact on private vehicle holding, Business Strategy and the Environment 21 (4) (2012) 264–280.

[21] S. Habibi, F. Sprei, C. Englundn, S. Pettersson, A. Voronov, J. Wedlin, H. Engdahl, Comparison of free-floating car sharing services in cities, in: European Council of Energy Efficient Economy Summer Study, 2016, pp. 771–778.

[22] K. Kortum, R. Schnduwe, B. Stolte, B. Bock, Free-floating carsharing: City-specific growth rates and success factors, Transportation Research Procedia 19 (2016) 328 – 340.

[23] S. Schmller, S. Weikl, J. Mller, K. Bogenberger, Empirical analysis of free-floating carsharing usage: The Munich

and Berlin case, Transportation Research Part C: Emerging Technologies 56 (2015) 34 – 51.

[24] R. Bi, J. Xiao, D. Pelzer, D. Ciechanowicz, D. Eckhoff, A. Knoll, A simulation-based heuristic for city-scale electric vehicle charging station placement, in: Intelligent Transportation Systems (ITSC), 2017 IEEE 20th International Conference on, 2017, pp. 1–7.

[25] M. Eisel, B. Hildebrandt, L. Kolbe, J. Schmidt, Applying demand response programs for electric vehicle fleets, in: AMCIS, 2015.

[26] M. F. Jung, D. Sirkin, T. M. Gür, M. Steinert, Displayed uncertainty improves driving experience and behavior: The case of range anxiety in an electric car, in: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, 2015.

[27] S. Rao, Engineering Optimization: Theory and Practice: Fourth Edition, John Wiley and Sons, 2009.

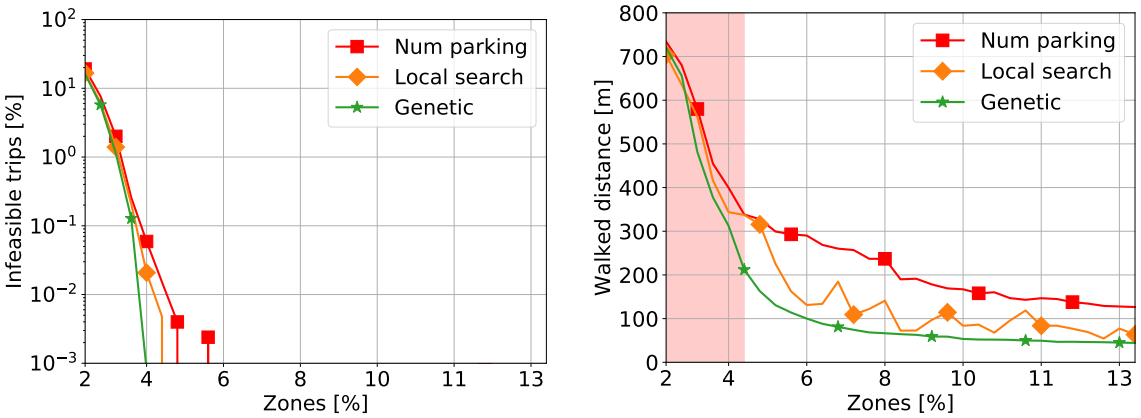
[28] C. C. A. Floudas, P. M. Pardalos, Encyclopedia of Optimization, Springer-Verlag, 2006.

[29] S. J. Wright, Coordinate descent algorithms, Math. Program. 151 (1) (2015) 3–34.

[30] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, 1st Edition, Addison-Wesley Longman Publishing, 1989.

[31] M. Smith, J. Castellano, Costs associated with non-residential electric vehicle supply equipment: Factors to consider in the implementation of electric vehicle charging stations, Tech. rep. (2015).

Appendix A. Placement optimisation for *Needed* return policy

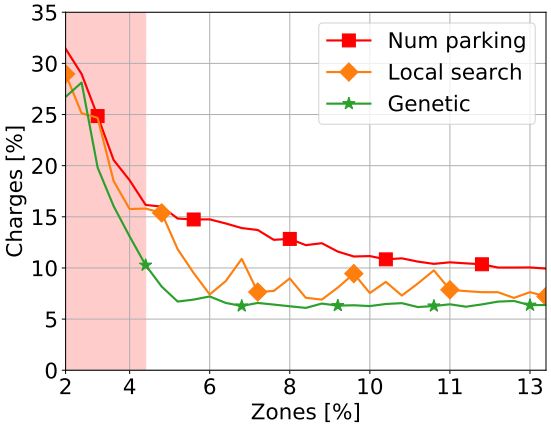


(a) Percentage of infeasible trips. Y-Axis is logarithmic.

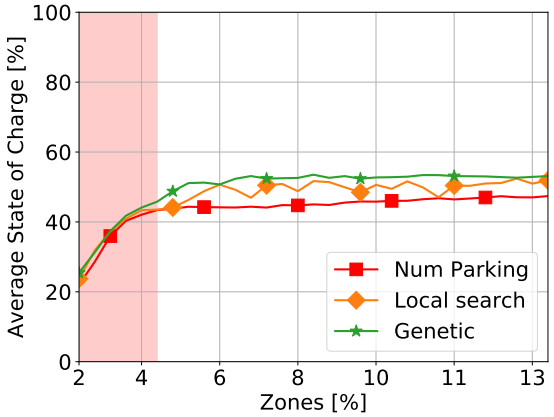
(b) Walked distance, averaged over all trips.

Figure A.11: Objective metrics to minimise in the optimisation - with *Needed* return policy

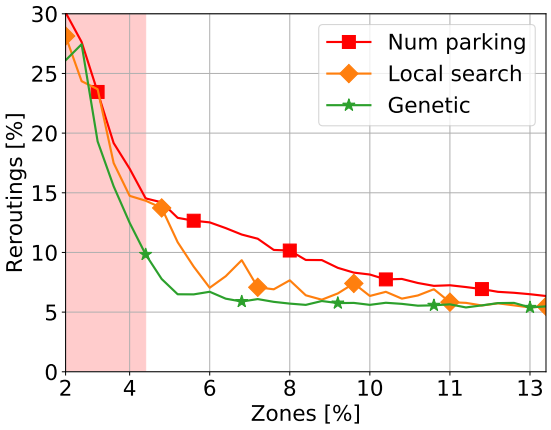
Here we briefly report the results for the optimisation experiments of the *Needed* policy. We followed the same procedure explained in Sec. 7 for the *Hybrid* return policy. As in that case,



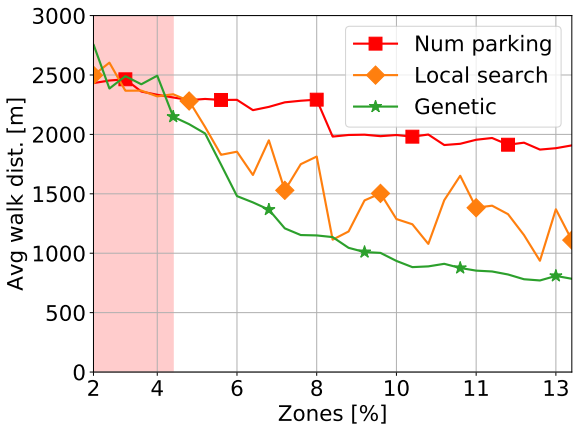
(a) Charges percentage.



(b) Average state of charge.



(c) Rerouted trips percentage.



(d) Walked distance when rerouted.

Figure A.12: *Genetic* and *local search* optimization results for metrics of interests (*Needed* return policy adopted).

the genetic algorithm is able to largely optimise the solution, as reported in Fig. A.11, with *local searches* stuck in local minima. In particular, for the walked distance (Fig. 11(b)), the genetic algorithm is able lower it from 136 m to 45 m at 13% of zones. Still, it doesn't reach the performance of the *Hybrid* policy, i.e., 30 m at 13% of zones.

Fig A.12 reports the other user discomfort metrics. The two optimisation algorithms reduce the charge events (Fig 12(a)). However, the average state of charge (Fig. 12(b)) is always higher in the optimised solutions than in the *Num parking* configuration. This further demonstrates that a smarter placement allows the car to get more energy for each charge. Fig. 12(c) reports the rerouting percentage for the different algorithms. As expected, the values are larger than with the

Hybrid policy, since no opportunistic charge is performed. However, the genetic algorithm is able to quickly reach a small value of reroutings, hence better exploiting every charge possibility. Finally, we analyse the distance a user has to walk when rerouted (Fig. 12(d)). Here, with respect to the *Hybrid* policy case, the *local search* shows larger deviations from the *Num parking*. The genetic algorithm reaches values of about 800 m, even below the ones reached with the *Hybrid* policy.

In conclusion, within this range of zones equipped with charging stations, a smart placement with the *Needed* policy approaches the *Hybrid* policy results.