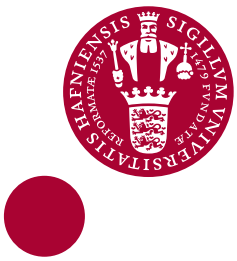


Optimal charging and repositioning of electric vehicles in a free-floating carsharing system

Folkestad, Carl Axel; Hansen, Nora; Fagerholt, Kjetil; Andersson, Henrik; Pantuso, Giovanni

Published in:
Computers and Operations Research

DOI:
10.1016/j.cor.2019.104771

Publication date:
2020

Document version
Publisher's PDF, also known as Version of record

Computers and Operations Research 113 (2020) 104771

Contents lists available at ScienceDirect

Computers and Operations Research

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

Optimal charging and repositioning of electric vehicles in a free-floating carsharing system

Carl Axel Folkestad[a], Nora Hansen[a], Kjetil Fagerholt[a,*], Henrik Andersson[a], Giovanni Pantuso[b]

[a] Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway
[b] Department of Mathematical Sciences, University of Copenhagen, Copenhagen, Denmark

ARTICLE INFO

Article history:
Received 2 August 2018
Revised 28 June 2019
Accepted 17 August 2019
Available online 21 August 2019

Keywords:
One-way carsharing
Free-floating carsharing
Vehicle relocation optimization
Integer programming
Genetic algorithm

ABSTRACT

Carsharing has received increased attention from the Operations Research community in recent years. Currently, many systems are adopting electric vehicles that require charging when battery levels fall below a given level. To do this, staff is often used to move cars to charging stations. Repositioning cars, rather than simply moving them to the closest charging station, might provide a better distribution of cars and in turn generate increased revenue and customer service while only marginally increase the operational costs. We present a mathematical model for the problem of charging and repositioning a fleet of shared electric cars. The model considers the assignment of cars to charging stations and the routing of staff and service vehicles. The complexity of the resulting mixed integer program makes it impossible to solve real world instances using a commercial solver. Therefore, we propose a new Hybrid Genetic Search with Adaptive Diversity Control algorithm. Tests based on data from a real life carsharing organization demonstrate that the proposed method can handle real size instances and that combining repositioning and charging operations can give significant benefits.

1. Introduction

Carsharing systems have existed in various forms for several decades. Lately, due to the enabling power of internet technologies, their popularity has increased, making them a standard means of transport in several urban areas across the globe. Short-term car rental can serve some of the users' transportation needs without the financial commitment of purchase, insurance, parking, and maintenance that comes with private ownership. Carsharing is generally defined as short-term vehicle access among a group of members who share the use of a vehicle fleet that is owned, maintained, managed, and insured by a *Carsharing Organization* (CSO). The rental cars can be both gasoline and electric vehicles, implying different operational challenges. Carsharing services can be divided into two categories; *free-floating systems* and *station-based systems*. Free-floating systems enable users to pick up available cars and deliver them anywhere within a specified business area. In a station-based system, the cars are allocated to dedicated stations. A station-based system is either a *two-way* or a *one-way* system. In a two-way system, the user must pick up and return the car at the same station, while in a one-way system the user can pick up and return the car at different stations.

Charging and *repositioning* of rental cars are among the most important operational challenges in electric vehicle free-floating systems. Charging consists of using staff to relocate a shared car to a charging station when the battery level falls below a given threshold. Repositioning consists of redistributing rental cars in order to improve the CSO's ability to meet customer demand. In fact, demand imbalances may result in rental cars accumulating in certain areas of the city while other areas remain unsupplied. CSOs adopt different strategies for charging electric cars and for preventing or resolving poor distributions of shared cars, such as pricing schemes which penalize parking in certain zones of the city or offering bonuses for customers plugging in shared cars with depleted batteries (see e.g., Hansen and Pantuso, 2018; Jorge et al., 2015). However, despite possible preventive measures, both charging and repositioning require the employment of dedicated staff moving the cars from their current position to charging stations or to other areas for rebalancing purposes.

On the one hand, these operations typically result in substantial costs for the CSO. On the other hand, efficient charging and repo-

* Corresponding author.
E-mail addresses: kjetil.fagerholt@ntnu.no (K. Fagerholt), henrik.andersson@ntnu.no (H. Andersson), gp@math.ku.dk (G. Pantuso).

https://doi.org/10.1016/j.cor.2019.104771

sitioning can significantly increase both resource utilization and customer satisfaction by means of a better supply of rental cars. Ultimately, it is essential for the sustainability of carsharing systems to find good solutions to these problems.

In this paper we study the integrated problem of charging and repositioning a fleet of shared electric vehicles in a one-way free-floating system, an operational planning problem which we have denoted as the Free-Floating Electric Carsharing Charging and Repositioning Problem (FFECCRP). Consistently with the operation of the CSO that inspired this work, we assume that service operators (members of staff) are transported to electric rental cars in need of charging by service vehicles (typically large cars or minivans) with fixed capacity. After being dropped off, the operator drives the rental car to the selected charging station (which is also a decision), where a service vehicle (not necessarily the same one that dropped him/her off) must pick him/her up and transport him/her to his/her next rental car to be handled. This gives a very complex routing problem, where one has to decide i) the routes of the service vehicles that are dropping off and picking up the service operators, ii) routes of the service operators, iii) which rental cars to charge/reposition, and iv) to which charging station (not necessarily the nearest one) to bring each rental car. The time and location at which an operator is dropped off affect the pick up of the operator leading to temporal and spatial interdependencies between the routing of operators and service vehicles.

Repositioning in one-way carsharing systems has recently received increased attention in the literature due to the increased flexibility for the users and the consequent imbalances caused in the distribution of the fleet. Example of studies on the shared vehicle repositioning are (Boyaci et al., 2015; 2017; Bruglieri et al., 2017; Cepolina and Farina, 2012; Correia and Antunes, 2012; Kaspi et al., 2014; Kek et al., 2009; Kuhne et al., 2016; Nourinejad et al., 2015; Repoux et al., 2015; Zhao et al., 2018) and, particularly for free-floating systems (Herbawi et al., 2016; Herrmann et al., 2014; Schulte and Voß, 2015; Weikl and Bogenberger, 2013; 2015). Similarly to Weikl and Bogenberger (2015), we pursue the idea of combining charging of Electric Vehicles (EVs) with depleted batteries with relocation activities. In Weikl and Bogenberger (2015), the authors propose a relocation model for free-floating carsharing services consisting of six consecutive steps. First, the target distribution of vehicles is obtained based on historical data. Second, an optimization model is used to determine the inter-zone relocation of vehicles in order to maximize the profit expressed as the difference between the sales generated by the resulting distribution of vehicles, minus the costs incurred by the relocation, e.g., vehicles movements and personnel wages. This step is followed by two rule-based intra-zone relocation plans at the vehicle level. Finally, service trips are defined for maintenance and charging activities. The authors show that this method is computationally efficient also for large-scale systems. In this paper, we focus on the planning of staff-based charging and relocation activities at a higher level of granularity. The optimization model presented by Weikl and Bogenberger (2015) determines the number of vehicles to move between each pair of macro-zones, such that a number of constraints are satisfied (e.g., bounds to the number of vehicles in each zone enforced by public authorities). For each relocation, the cost of driving the rental vehicle and the cost of the operators are incurred. In order to take into account the movements of the operators, in addition to the relocation time, each relocation consumes an "average approach time" that accounts for the fact that the operator must somehow reach the rental vehicle in order to relocate it. Our optimization model "zooms in" on the relocation activities, in order to define not only which moves should be performed, but also how the operators should optimally move around the city in order to perform relocations. That is, in addition to determining which relocations to perform, our model determines routes and schedules for each operator and service vehicle. Using the terminology of Weikl and Bogenberger (2015) our approach is applicable to both intra-zone and inter-zone relocations, depending on the granularity of the zonification.

Staff-based repositioning is also the focus of Zhao et al. (2018), who present a mathematical model for the integrated EV rebalancing and staff relocation for one-way station-based systems. With respect to this study: a) we consider a free-floating system, allowing the users to drop off cars at any common parking place, and b) we consider a system with real-time reservations rather than a priori reservation. The first point entails that staff might have to relocate cars with depleted batteries to charging stations as this is not necessarily done by users. This task is additional to car relocations pursuing a better distribution of the fleet. The second point entails that in the study of Zhao et al. (2018), staff has to relocate EVs in order to fulfill binding reservations by customers. That is, staff-based relocation is used as a strategy to reduce the fleet size. In our study, staff is not responsible to fulfill reservations but works to ensure a more profitable distribution of cars.

The routing of rental cars and operators is also included in several other studies. Boyaci et al. (2015) and Boyaci et al. (2017) route rental cars and operators separately, but do not consider the trade off between the costs and additional revenue of repositioning. Nourinejad et al. (2015) address the trade-off between repositioning costs and gains by including both routing of rental cars and operators in the same problem. However, only repositioning under the assumption that the CSO must fulfill all demand is performed, with no attention to daily operations like charging. Similarly, Herbawi et al. (2016) propose an evolutionary algorithm to determine the routing of both operators and a single service vehicle transporting the operators which severely restricts the routing possibilities. Finally, Bruglieri et al. (2017) propose heuristic methods for solving staff-based relocation problems with electric vehicles. The authors assume that users always drop off and pick up cars at charging stations. This entails that cars are automatically charged when not used. In this paper we relax this assumption, and consider the case of a purely free-floating service. This means that operators are responsible for moving cars with depleted batteries to charging stations.

One important aspect that distinguishes the problem that we study in this paper with all previous studies is that we integrate routing of the service vehicles and operators in the case where the service operators are transported by service vehicles to the rental EVs, dropped off and picked up (possibly by another service vehicle) after they have taken the EVs to their recharging station. This, combined with that the destinations for EVs to be charged (and possibly repositioned) are also determined within the optimization makes the FFECCRP an extremely complex routing problem. This also makes it very different to the electric vehicle routing problem, see for example Schneider et al. (2014).

In this paper we present a mathematical model for the FFECCRP. Since the proposed mixed integer programming model is unable to solve real world instances when using a commercial solver, we design a solution algorithm for solving real-life problem instances. The solution method consists of a Hybrid Genetic Search with Adaptive Diversity Control algorithm. The contributions of this paper are therefore:

1. A detailed description of a new problem emerging in carsharing systems.
2. A novel mathematical model for the problem of charging and repositioning electric vehicles in a free-floating carsharing system.
3. A Hybrid Genetic Search with Adaptive Diversity Control algorithm (HGSADC) capable of solving instances of size compatible with real-life problems.

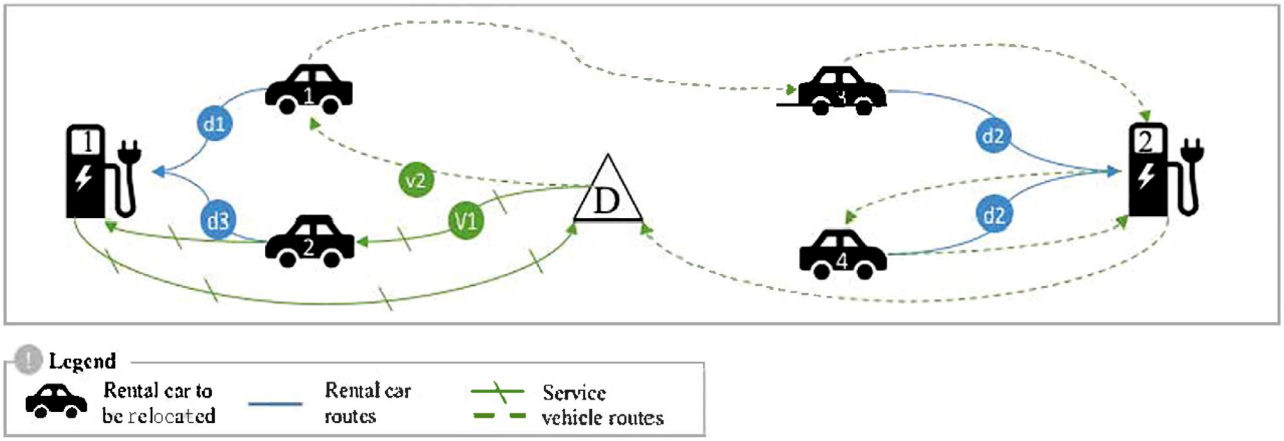


Fig. 1. Illustration of an example problem with four cars (1,2,3,4) in need of charging, two charging stations (1,2), two service vehicles (v1,v2), and three operators (d1,d2,d3).

4. Results showing the effectiveness of the proposed algorithm and that combining charging and repositioning yields advantages for the CSO.

The paper is organized as follows. First, the FFECCRP is described in Section 2. Then, a mathematical model of the problem is proposed in Section 3 before the HGSADC is described in Section 4. A computational study is presented in Section 5, while Section 6 discusses the economical implications of the model. Finally, we conclude in Section 7.

2. Problem description

In this section we provide a more thorough description of the problem. Most features of the problem are based on discussions with a CSO operating in a major European city, which, for the sake of confidentiality, will remain unnamed. We define the Free-Floating Electric Carsharing Charging and Repositioning Problem (FFECCRP) as the problem of repositioning electric rental cars to charging stations in a free-floating carsharing system when their battery level falls below a predefined threshold. When a rental car needs charging, a member of staff (here and in what follows named *operator*) *relocates* the rental car from its original position to a charging station. Only cars with battery level below the fixed level are relocated. The charging station for each car is chosen, among those that can be reached with the available charge, taking into account the current distribution of cars in the business area. This way charging and repositioning operations are combined. This implies that the CSO in some cases might want to reposition a car to a far away charging station if this improves the distribution of the fleet. In addition, a relocation to a far away charging station might also be used to move an operator from one area of the city to another. Each charging station has a given number of available charging slots and must have free capacity if a car is relocated to that station. We assume that users do not compete for charging slots with the CSOs operator.

Consistently with the operation of the CSO that inspired this work, operators are transported to rental cars by service vehicles (typically large cars or mini-vans) with fixed capacity. Subsequently, the operator drives the rental car to the selected charging station. After an operator has relocated a car to a charging station, a service vehicle will pick him/her up at the charging station and transport him/her to his/her next rental car. Operators are not necessarily picked up by the same service vehicle that dropped them off, but service vehicles are the only available means of transport. Though alternative means of movement for the operators are used in some cities (e.g., foldable bikes or public transport), service vehicles are adopted by several CSOs, like the one this study is based on. A given number of operators and service vehicles are available at the depot in the beginning of the planning period.

The *planning period*, i.e., the total time available for charging and repositioning the fleet of rental cars, may vary from one up to several hours depending on the hour of the day, the area to cover (different groups of operators might cover different areas of the city), and on the number of cars to charge and reposition. Typically, a shorter planning period is adopted during the day in order to respond to the likely changes in the system while the fleet is being used by customers. Cars in need of charging are made unavailable in the booking system and remain unavailable until they are charged to a sufficient battery level. In practice, additional rental cars might have their battery depleted during the planning horizon and thus need to be charged. However, these cars will be relocated in the subsequent planning period or by a separated group of operators starting their planning period at a later time. When a rental car with depleted battery has been moved to a charging station, it is made unavailable to the customers until the battery has been fully charged. Once the battery is fully charged the car is again made available for customers and the charging slot it occupied is made available (in practice either a customer or the first operator visiting the charging station will unplug it).

Fig. 1 illustrates a small example problem with four rental cars in need of charging, two charging stations, two service vehicles, and three operators. Service vehicle 1 transports operator 3 to rental car 2, which is relocated to charging station 1. Service vehicle 2 transports operators 1 and 2 to rental cars 1 and 3, respectively. Operator 1 relocates the rental car to charging station 1, where service vehicle 1 picks up both operators 1 and 3 and returns them to the depot. Operator 2 relocates rental car 3 to charging station 2, and is picked up and transported to rental car 4 by service vehicle 2. The operator relocates the rental car to charging station 2 where he is again picked up by service vehicle 2 and transported back to the depot.

To quantify the distribution of rental cars in the system, the concept of states is introduced for each charging station. Each charging station is assigned a *surrounding area* and from here on the surrounding area is included when discussing charging stations. The *initial state* describes the number of rental cars available for customers at the charging station when the planning period starts, i.e. all rental cars at the charging station and in the surrounding area with a battery level above a given threshold. The *ideal state* gives the ideal number of rental cars around each charging station. The ideal state is typically dependent on the mobility demand and addressed in a separated planning problem. The mobility demand for each time period is assumed to be known in advance and, consequently, the ideal state is known at the time

of planning. After solving the FFECCRP, the *final state* is reached. The final state equals the initial state at a charging station plus the number of rental cars relocated to the station.

The costs a CSO incurs are due to the direct costs of relocating of rental cars, and to the opportunity cost due to the postponement of charging and to imbalances in the distribution of cars. Particularly, the cost of relocating cars is the cost of moving operators with service vehicles to and from rental cars in need of charging as well as the fixed cost for using each individual operator and service vehicle. If charging is postponed, the rental car is unavailable for customers for longer time or until the next planning period. Furthermore, demand might be lost when there is a deviation between the ideal and final state at a charging station at the end of the planning period. The CSO is in general able to quantify the opportunity cost for the unavailability of cars. In practice, the actual state of the system might be different from the final state due to customers using the available cars. Therefore, the opportunity cost for imbalances and postponement of charging is to be understood as a driver towards a better usage of the system rather than a cost to include in balance sheets.

The FFECCRP includes several connected decisions: i) The routes of the service vehicles that are dropping off and picking up the service operators, ii) the routes of the service operators, iii) to which rental cars to charge/reposition, and iv) which charging station (not necessarily the nearest one) to bring each rental car. The time and location at which an operator is dropped off affect the pick up of the operator leading to temporal and spatial interdependencies between the routing of operators and service vehicles. The objective is to minimize the costs of relocating, postponement, and deviations. Central to the problem is the trade off between the cost of not meeting demand due to a disadvantageous distribution of cars in the system and the cost of repositioning.

3. Mixed integer programming model

In this section we propose a mathematical formulation of the FFECCRP. The underlying network consists of nodes representing visits by service vehicles and operators to charging stations, rental cars, and the depot and arcs between these visits. The mth visit to i, called (i, m), by service vehicle v is the node (i, m, v), where i being either a charging station, rental car, or the depot. Similarly, the ath visit to i, called (i, a), by operator d is the node (i, a, d). When creating the network, only rental cars in need of charging are considered. Charging stations can be visited multiple times by both operators and service vehicles while rental cars can only be visited once. Furthermore, if a rental car is visited it must be relocated. All service vehicles and operators start and end at the depot. Service vehicles drive directly between nodes and cannot make intermediate stops at the depot. Operators can only be dropped off in nodes representing rental cars or the depot and picked up in nodes representing charging stations or the depot.

In the following, we present the applied notation and model formulation. Where applicable, the depot is given index $i = 0$ and the first visit to a node (i, m, v) is indexed with $m = 1$, the second with $m = 2$ and so on. Similarly, the first visit by an operator to a node (i, a, d) is indexed $a = 1$ and so on.

3.1. Sets

$\mathcal{N}$	Set of all nodes
$\mathcal{N}^{CS}$	Set of all charging stations, $\mathcal{N}^{CS} \subset \mathcal{N}$
$\mathcal{N}^{EV}$	Set of rental cars in need of charging, $\mathcal{N}^{EV} \subset \mathcal{N}$
$\mathcal{M}_i$	Set of all possible visits to node i by each service vehicle/operator
$\mathcal{V}$	Set of all service vehicles
$\mathcal{D}$	Set of all operators

3.2. Parameters

Q_j^{CSP}	Number of available charging slots at charging station j
C_j^E	Cost for each car in excess or surplus of the ideal state at charging station j
C_{ij}^T	Travel cost for the service vehicles between node i and j
C_i^{PH}	Cost of postponed charging of rental car i
C^V	Fixed service vehicle cost
C^D	Fixed operator cost
T_{ij}	Travel time between node i and j
T_i^{EV}	Maximum travel time for rental car at node i
T	Length of the planning period
Q	Service vehicle capacity
S_j^0	Initial state at charging station j
S_j^I	Ideal state at charging station j

3.3. Variables

x_{imjnv}	1 if service vehicle v drives directly from visit m at node i to visit n at node j, 0 otherwise
$f_{imajnbvd}$	1 if operator d is transported from visit a at node i to visit b at node j by service vehicle v driving from visit m at node i to visit n at node j, 0 otherwise
q_{ivd}	1 if operator d is dropped off at rental car i by service vehicle v, 0 otherwise
g_{jnbvd}	1 if operator d is picked up at visit (j, b) by service vehicle v at visit (j, n), 0 otherwise
h_{ijbd}	1 if operator d relocates rental car i to charging station visit (j, b), 0 otherwise
t_{imv}^V	Time of arrival to visit (i, m) for service vehicle v
t_{iad}^D	Time of arrival to visit (i, a) for operator d
z_i^H	1 if rental car i is not charged, 0 otherwise
y_j	Number of cars in excess or deficit of the ideal state at charging station j
s_v	1 if service vehicle v is used, 0 otherwise
w_d	1 if operator d is used, 0 otherwise

3.4. Formulation

$$\min \sum_{j\in\mathcal{N}^{CS}} C_j^E\, y_j + \sum_{i\in\mathcal{N}} \sum_{m\in\mathcal{M}_i} \sum_{j\in\mathcal{N}} \sum_{n\in\mathcal{M}_j} \sum_{v\in\mathcal{V}} C_{ij}^T\, x_{imjnv} + \sum_{i\in\mathcal{N}^{EV}} C_i^{PH}\, z_i^H + \sum_{v\in\mathcal{V}} C^V\, s_v + \sum_{d\in\mathcal{D}} C^D\, w_d \quad (1)$$

$$\text{s.t.} \sum_{j\in\mathcal{N}\setminus\{0\}} x_{01j1v} = s_v \qquad v \in \mathcal{V} \quad (2)$$

$$\sum_{j\in\mathcal{N}\setminus\{0\}} \sum_{m\in\mathcal{M}_j} x_{jm02v} = s_v \qquad v \in \mathcal{V} \quad (3)$$

$$\sum_{j\in\mathcal{N}\setminus\{0\}} \sum_{n\in\mathcal{M}_j} x_{imjnv} \le s_v \qquad i \in N\setminus\{0\}, m \in \mathcal{M}_i, v \in \mathcal{V} \quad (4)$$

$$\sum_{i\in\mathcal{N}} \sum_{m\in\mathcal{M}_i} x_{imjnv} = \sum_{i\in\mathcal{N}} \sum_{m\in\mathcal{M}_i} x_{jnimv} \qquad j \in \mathcal{N}\setminus\{0\}, n \in \mathcal{M}_j, v \in \mathcal{V} \quad (5)$$

$$\sum_{i\in\mathcal{N}^{EV}} \sum_{b\in\mathcal{M}_j} \sum_{d\in\mathcal{D}} h_{ijbd} \le Q_j^{CSP} \qquad j \in \mathcal{N}^{CS} \quad (6)$$

$$\sum_{j\in\mathcal{N}^{CS}} \sum_{b\in\mathcal{M}_j} \sum_{d\in\mathcal{D}} h_{ijbd} + z_i^H = 1 \qquad i \in \mathcal{N}^{EV} \quad (7)$$

$$\sum_{j\in\mathcal{N}^{CS}} \sum_{b\in\mathcal{M}_j} h_{ijbd} = \sum_{v\in\mathcal{V}} q_{ivd} \qquad i \in \mathcal{N}^{EV}, d \in \mathcal{D} \quad (8)$$

$$\sum_{i\in\mathcal{N}^{EV}} h_{ijbd} = \sum_{v\in\mathcal{V}} \sum_{n\in\mathcal{M}_j} g_{jnbvd} \qquad j \in \mathcal{N}^{CS}, b \in \mathcal{M}_j, d \in \mathcal{D} \quad (9)$$

$$\sum_{i\in\mathcal{N}}\sum_{m\in\mathcal{M}_i}\sum_{a\in\mathcal{M}_i}\sum_{n\in\mathcal{M}_j}\sum_{v\in\mathcal{V}} f_{imajnbvd} + \sum_{i\in\mathcal{N}^{EV}} h_{ijbd} \le w_d \quad j\in\mathcal{N}^{CS}, b\in\mathcal{M}_j, d\in\mathcal{D} \quad (10)$$

$$\sum_{j\in\mathcal{N}\setminus\{0\}}\sum_{v\in\mathcal{V}} f_{011j11vd} = w_d \quad d\in\mathcal{D} \quad (11)$$

$$\sum_{i\in\mathcal{N}}\sum_{m\in\mathcal{M}_i}\sum_{a\in\mathcal{M}_i}\sum_{v\in\mathcal{V}} f_{ima022vd} = w_d \quad d\in\mathcal{D} \quad (12)$$

$$\sum_{k\in\mathcal{N}}\sum_{o\in\mathcal{M}_k}\sum_{c\in\mathcal{M}_k} f_{jnbkocvd} = \sum_{i\in\mathcal{N}}\sum_{m\in\mathcal{M}_i}\sum_{a\in\mathcal{M}_i} f_{imajnbvd} + g_{jnbvd} - q_{jvd}$$
$$j\in\mathcal{N}\setminus\{0\}, n\in\mathcal{M}_j, b\in\mathcal{M}_j, v\in\mathcal{V}, d\in\mathcal{D} \quad (13)$$

$$\sum_{a\in\mathcal{M}_i}\sum_{b\in\mathcal{M}_j}\sum_{d\in\mathcal{D}} f_{imajnbvd} \le Q\ x_{imjnv} \quad i\in\mathcal{N}, m\in\mathcal{M}_i, j\in\mathcal{N}, n\in\mathcal{M}_j, v\in\mathcal{V} \quad (14)$$

$$t^V_{imv} \le \overline{T}\ s_v \quad i\in\mathcal{N}, m\in\mathcal{M}_i, v\in\mathcal{V} \quad (15)$$

$$(t^V_{imv} + T_{ij})\ x_{imjnv} \le t^V_{jnv} \quad i\in\mathcal{N}, m\in\mathcal{M}_i, j\in\mathcal{N}, n\in\mathcal{M}_j, v\in\mathcal{V} \quad (16)$$

$$t^D_{jbd}\ g_{jnbvd} \le t^V_{jnv} \quad j\in\mathcal{N}^{CS}, n\in\mathcal{M}_j, b\in\mathcal{M}_j, v\in\mathcal{V}, d\in\mathcal{D} \quad (17)$$

$$t^D_{iad} \le \overline{T}\ w_d \quad i\in\mathcal{N}, a\in\mathcal{M}_i, d\in\mathcal{D} \quad (18)$$

$$(t^D_{iad} + T_{ij})h_{ijbd} \le t^D_{jbd} \quad i\in\mathcal{N}^{EV}, a\in\mathcal{M}_i, j\in\mathcal{N}^{CS}, b\in\mathcal{M}_j, d\in\mathcal{D} \quad (19)$$

$$(t^V_{imv} + T_{ij})f_{imajnbvd} \le t^D_{jbd} \quad i\in\mathcal{N}, m\in\mathcal{M}_i, a\in\mathcal{M}_i, j\in\mathcal{N}, n\in\mathcal{M}_j, b\in\mathcal{M}_j, v\in\mathcal{V}, d\in\mathcal{D} \quad (20)$$

$$y_j \ge S^0_j + \sum_{i\in\mathcal{N}^{EV}}\sum_{b\in\mathcal{M}_j}\sum_{d\in\mathcal{D}} h_{ijbd} - S^I_j \quad j\in\mathcal{N}^{CS} \quad (21)$$

$$y_j \ge -S^0_j - \sum_{i\in\mathcal{N}^{EV}}\sum_{b\in\mathcal{M}_j}\sum_{d\in\mathcal{D}} h_{ijbd} + S^I_j \quad j\in\mathcal{N}^{CS} \quad (22)$$

$$x_{imjnv}\in\{0,1\} \quad i\in\mathcal{N}, m\in\mathcal{M}_i, j\in\mathcal{N}, n\in\mathcal{M}_j, v\in\mathcal{V}, d\in\mathcal{D} \quad (23)$$

$$f_{imajnbvd}\in\{0,1\} \quad i\in\mathcal{N}, m\in\mathcal{M}_i, a\in\mathcal{M}_i, j\in\mathcal{N}, n\in\mathcal{M}_j, b\in\mathcal{M}_j, v\in\mathcal{V}, d\in\mathcal{D} \quad (24)$$

$$q_{ivd}\in\{0,1\} \quad i\in\mathcal{N}^{EV}, v\in\mathcal{V}, d\in\mathcal{D} \quad (25)$$

$$g_{jnbvd}\in\{0,1\} \quad j\in\mathcal{N}^{CS}, n\in\mathcal{M}_j, b\in\mathcal{M}_j, v\in\mathcal{V}, d\in\mathcal{D} \quad (26)$$

$$h_{ijbd}\in\{0,1\} \quad i\in\mathcal{N}^{EV}, j\in\mathcal{N}^{CS}, b\in\mathcal{M}_j, d\in\mathcal{D}|T_{ij}\le T^{EV}_i \quad (27)$$

$$t^V_{imv}\ge 0 \quad i\in\mathcal{N}, m\in\mathcal{M}_i, v\in\mathcal{V} \quad (28)$$

$$t^D_{iad}\ge 0 \quad i\in\mathcal{N}, a\in\mathcal{M}_i, d\in\mathcal{D} \quad (29)$$

$$z^H_i\in\{0,1\} \quad i\in\mathcal{N}^{EV} \quad (30)$$

$$y_j\in\mathbb{Z}^+ \quad j\in\mathcal{N}^{CS} \quad (31)$$

$$s_v\in\{0,1\} \quad v\in\mathcal{V} \quad (32)$$

$$w_d\in\{0,1\} \quad d\in\mathcal{D} \quad (33)$$

The objective function (1) minimizes the cost of relocating and the costs of deviating from the ideal state at each charging station and for postponed charging. Constraints (2) and (3) state that if a service vehicle is used, it must leave and return to the depot, respectively. Constraints (4) enforce that only service vehicles in use visit nodes and that only one arc is leaving a given visit (i, m). Constraints (5) ensure that a vehicle arriving a visit (j, n) leaves the node from the same visit. This must hold for all nodes except the depot. Constraints (6) make sure that the number of rental cars relocated to a station does not exceed the number of available charging slots at the station. Note that in the model a vehicle occupies a charging slot until the battery is fully charged or at least for the entire planning period. That is, we do not allowing partial or split charging. This might be considered a current limitation of the model. Constraints (7) force either the relocating variable or the postponed charging variable to 1 for all rental cars.

Constraints (8) and (9) state that an operator relocating a rental car is dropped off by the rental car and picked up at the charging station the rental car is relocated to, respectively. Constraints (10) make sure that an operator only makes a given visit b to a charging station once, either by driving a rental car to the charging station or by being transported through the charging station. Constraints (11) enforce that a service vehicle can only transport operators in use and that an operator only can be picked up by one service vehicle at the depot. Constraints (12) ensure that operators are returned to the depot. Constraints (13) maintain the flow of operators in all nodes, ensuring that an operator transported out of a node must be transported to that node or picked up in that node and vice versa. q_{ivd} only exist for $i\in N^{EV}$ and g_{jnbvd} only exist for $j\in N^{CS}$. Constraints (14) make sure that a service vehicle does not exceed its seat capacity transporting operators and force the flow on arcs not driven by a service vehicle to 0.

Constraints (15)–(17) determine the service vehicle arrival time in all nodes. Constraints (15) state that visits by the service vehicle must happen before the end of the planning horizon. Constraint (16) ensure that if the service vehicle travels directly from i to j, the visit at node j happens at a later time than the visit at node i. Constraints (17) state that an operator should have arrived at node j if he/she must be picked up at that node by a service vehicle. Constraints (18)–(20) determine operator arrival times in all nodes. Constraints (18) state that an operator should arrive at a node before the end of the planning horizon. Constraints (19) state that if an operator moves directly from i to j, he/she arrives at j after he arrives at i. Finally, constraints (20) state that if an operator is transported by vehicle v from i to j, he/she arrives at j after the arrival of the service vehicle at i and the duration of the drive from i to j. Nonlinear constraints can be linearized using big-M formulations. Constraints (21) and (22) assign the absolute value of deviations from the ideal state in each charging station node to the variable accounting for deviations. Finally, constraints (23)–(33) define the variable domains.

4. Hybrid genetic search with adaptive diversity control

In this section we present a metaheuristic algorithm for solving the FFECCRP represented by model (1)–(33). The implementation of the heuristic draws on the Hybrid Genetic Search with Adaptive Diversity Control (HGSADC) first presented by Vidal et al. (2012). The motivation for choosing the HGSADC is that is has proven to perform well on a number of vehicle routing problems, see for

Algorithm 1 Hybrid Genetic Search with Adaptive Diversity Control (HGSADC).

```
Initialize population                                                    Section 4.3
iterationsWithoutImprovement ← 0
time ← 0
while iterationsWithoutImprovement < I^NI and time < T^MAXRUN do
    Select parent individuals s_1 and s_2                                Section 4.4
    Generate offspring s_new from s_1 and s_2
    Educate offspring s_new with probability ρ^EDU_offspring             Section 4.5
    if s_new is infeasible then
        Repair s_new with probability ρ^REP_offspring                    Section 4.5
    end if
    if s_new is still infeasible then
        Insert s_new into infeasible subpopulation
    else
        Insert s_new into feasible subpopulation
    end if
    if maximumPopulationSize μ + λ reached then
        Select survivors                                                 Section 4.6
    end if
    Adjust penalty parameters for violating feasibility condition        Section 4.6
    if bestIndividual not improved then
        iterationsWithoutImprovement ← iterationsWithoutImprovement+1
        if bestIndividual not improved for I^DIV iterations then
            Diversify population                                         Section 4.6
        end if
    end if
    time ← updateTime()
end while
Return best feasible individual
```

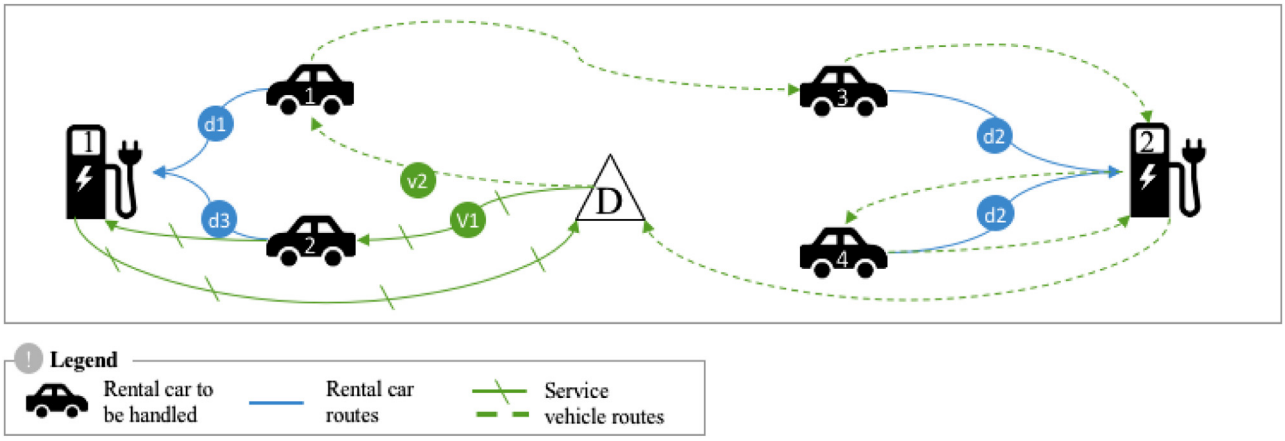
example (Borthen et al., 2018; Bulhões et al., 2018; Vidal et al., 2012; 2013). The original HGSADC has been modified and extended significantly to fit the FFECCRP. Algorithm 1 shows an overview of the HGSADC proposed to solve the FFECCRP. The algorithm evolves a population of individuals, where an individual represents a solution to the FFECCRP. The population is divided into two disjoint subpopulations; a subpopulation of feasible solutions and a subpopulation of infeasible solutions that together make up the entire population. The metaheuristic literature indicates that allowing a controlled exploration of infeasible solutions may enhance the performance of the search (Vidal et al., 2012). Hence, we allow solutions to be infeasible with respect to the maximum duration and the number of service vehicles used as we believe optimal solutions lie near the feasibility boundary of these constraints.

The algorithm breeds new individuals from the population as long as there have been improvements within the last I^{NI} iterations or the maximum running time limit T^{MAXRUN} is not reached. In each iteration, the algorithm picks two parent individuals and combines them, yielding a new individual denoted an *offspring*. The offspring is improved using an *education* procedure and, if infeasible, further improved using a *repair* procedure. The maximum population size (sum of feasible and infeasible subpopulation) is given by μ + λ, where μ is the minimum population size and λ is the generation size. When the maximum population size is reached, the individuals with highest *biased fitness*, i.e. high cost and low diversity contribution, are removed until there are only μ individuals left in the population. This process is referred to as *survivor selection*. To prevent the algorithm from converging to a local optimum, a diversification procedure is performed if there has been no improvement for I^{DIV} iterations. The initial population is created using a construction heuristic and must be large enough to contribute sufficiently to the diversity of the population.

4.1. Individual representation

An individual describes the routes of all service operators and service vehicles. The operator routes include assignment of operators to relocate each rental car, postponement of charging or assignment of rental cars to charging stations, and the relocating order of each operator. The routes of the service vehicles include assignment of transport requests by operators and the visit sequence of each service vehicle.

Each individual s in the population $\mathcal{S}$ is represented by five *chromosomes*. Here we describe these chromosomes and the information they contain. In the subsequent sections we elaborate on how they are used when individuals are evaluated, created, and changed. The first chromosome is the *rental car destination chromosome* $\alpha(s)$, determining the charging station to move a rental car to. Alternatively, determining that the charging of the car is postponed. The second chromosome is the *service operator chromosome* $\beta(s)$, that for each rental car defines which service operator that is going to perform the relocation. The third chromosome is the *relocation sequence chromosome* $\gamma(s)$, that for each operator d defines the order to relocate the rental cars assigned to the operator. Taking the first three chromosomes as given, *transport requests* for the operators that need to be taken care of by the service vehicles are formulated. A transport request is formulated for each pick up of an operator. The transport request is represented by a node pair, the first node is the origin where the operator is picked up and the second node the destination where the operator is dropped off. Each transport request is denoted $\tau_r(s)$ indexed by r and the set of all transport requests is denoted $\mathcal{R}$. The transport request formulation is used to define the fourth chromosome, the *transport request assignment chromosome* $\delta(s)$, that assigns each transport request r (from $\tau_r(s)$) to a service vehicle v. Finally, the last chromosome is



Rental car i	1	2	3	4
$\alpha_i(s)$	1	1	2	2

(a) Rental car destination chromosome $\alpha(s)$

Rental car i	1	2	3	4
$\beta_i(s)$	1	3	2	2

(b) Operator chromosome $\beta(s)$

Operator d	1	2	3
$\gamma_d(s)$	{1}	{3,4}	{2}

(c) Relocation sequence chromosome $\gamma(s)$

Transport request r	1	2	3	4	5	6	7
$\delta_r(s)$	2	1	2	2	2	1	1

(d) Transport request assignment chromosome $\delta(s)$

Service vehicle v	1	2
$\varepsilon_v(s)$	$\{D,EV2,CS1,D\}$	$\{D,EV1,EV3,CS2,EV4,CS2,D\}$

(e) Route chromosome $\varepsilon(s)$

Transport request r	1	2	3	4	5	6	7
Request $\tau_r(s)$	$\{D,EV1\}$	$\{CS1,D\}$	$\{D,EV3\}$	$\{CS2,EV4\}$	$\{CS2,D\}$	$\{D,EV2\}$	$\{CS1,D\}$

(f) Origin and destination of each transport request.

Fig. 2. Example of an individual for a small fictitious problem instance: Four cars in need of charging, two charging stations, two service vehicles, and three operators. The corresponding five chromosomes are given in Figure 2a-e. In the transport request assignment chromosome, the origin and destination of each transport request is stored in a separate list shown in Figure 2f. In the route chromosome, the depot is denoted D, the four rental cars are denoted EV1, EV2, EV3, and EV4 and the charging stations are denoted CS1 and CS2, respectively.

the *route chromosome* $\varepsilon(s)$, that describes the route of each service vehicle. The route chromosome determines the order a service vehicle visits the nodes defined by the transport request assignment chromosome. Fig. 2 illustrates a simple example solution and the corresponding chromosomes.

The following subsections explain how the different chromosomes are generated and combined to create new individuals.

4.2. Evaluation of individuals

A diverse population is important for GAs in order to avoid premature convergence to local optima and loss of information. The evaluation of individuals in the HGSADC is based on the *biased fitness* function presented by Vidal et al. (2012). The biased fitness function evaluates individuals based on their cost, how much they contribute to the diversity of the population, and how much they violate the constraints.

To evaluate the cost of an individual, let $\mathcal{A}(s)$ be the set of routes in individual $s \in \mathcal{S}$. Let c_{sa} be the cost of driving route $a \in \mathcal{A}(s)$, and C_s^E, C_s^{PH}, C_s^V, and C_s^D the cost of deviations from the ideal state, postponed charging, and use of service vehicles and operators in s, respectively. The individuals are allowed to violate the constraints on time used to perform relocation and the number of service vehicles used, i.e. constraints (15) and the size of the set of service vehicles $|\mathcal{V}|$. The *penalty costs* ϕ_{sa}^T and ϕ_s^V account for how much the time constraints are violated in route a and violations in number of service vehicles used in individual s, respectively. These are given by equations (34) and (35), where w^T is the *penalty parameter* per unit violation of the constraints on duration and t_{sa} is the duration of route a in individual s. w^V is the penalty parameter per unit violation of number of vehicles used by individual s, calculated by using the difference between the number of service vehicles used in s, V_s^{USED}, and available service vehicles $|\mathcal{V}|$. The *total cost* C_s of an individual s is calculated by equation (36).

$$\phi_{sa}^T = w^T max\{0, t_{sa} - \overline{T}\} \qquad s \in \mathcal{S}, a \in \mathcal{A}(s) \qquad (34)$$

$$\phi_s^V = w^V max\{0, V_s^{USED} - |\mathcal{V}|\} \qquad s \in \mathcal{S} \qquad (35)$$

$$C_s = \sum_{a \in \mathcal{A}(s)} (c_{sa} + \phi_{sa}^T) + \phi_s^V + C_s^E + C_s^{PH} + C_s^V + C_s^D \qquad s \in \mathcal{S} \qquad (36)$$

The *diversity contribution* of each individual s is defined as the average distance to its closest neighbors. Let $\mathcal{N}_s^{CLO}$ be the set containing the n^{CLO} closest neighbors of s. The diversity contribution, $\Pi(s)$, can then be calculated by equation (37) where $\pi(s, s')$ is the normalized Hamming distance between individual s and s'. The Hamming distance, first presented in Hamming (1950), is here taken as the number of different charging station assignments and the different relocation assignments, i.e. the difference between

Algorithm 2 Determining time windows.

0: $T_{ij} \leftarrow$ travel time between node i and j
1: **for** each operator $d \in D$ **do**
2: **for** each transport request r by operator d, $r \in \{r \mid \eta_r(s) = d, r \in \mathcal{R}(s)\}$ **do**
3: $l_r^o(s) \leftarrow l_{(r-1)}^d(s) + T_{\tau_{(r-1)}^d \tau_r^o}$
4: $l_r^d(s) \leftarrow l_r^o(s) + T_{\tau_r^o \tau_r^d}$
5: **end do**
6: **for** each transport request r by operator d, $r \in \{r \mid \eta_r(s) = d, r \in \mathcal{R}(s)\}$ (reverse direction) **do**
7: $u_r^d(s) \leftarrow u_{(r+1)}^o(s) - T_{\tau_r^d \tau_{(r+1)}^o}$
8: $u_r^o(s) \leftarrow u_r^d(s) - T_{\tau_r^o \tau_r^d}$
9: **end do**
10: **end do**

destination assignment $\alpha(s)$ and $\alpha(s')$ and the relocation assignment $\beta(s)$ and $\beta(s')$. With $\mathbf{1}(cond) = 1$ if condition $cond$ is true and 0 otherwise, the normalized Hamming distance can then be expressed as in equation (38).

$$\Pi(s) = \frac{1}{n^{CLO}} \sum_{s' \in \mathcal{N}_s^{CLO}} \pi(s, s') \qquad s \in \mathcal{S} \quad (37)$$

$$\pi(s, s') = \frac{1}{2|\mathcal{N}^{EV}|} \sum_{i \in \mathcal{N}^{EV}} \left(\mathbf{1}(\alpha_i(s) \neq \alpha_i(s')) + \mathbf{1}(\beta_i(s) \neq \beta_i(s'))\right) \qquad s \in \mathcal{S}, s' \in \mathcal{N}_s^{CLO} \quad (38)$$

Every individual is ranked based on its total cost and its diversity contribution. Let $Rank^C(s)$ and $Rank^D(s)$ be the rank of individual s in terms of total cost and diversity contribution, respectively. The individual with the lowest total cost will have $Rank^C(s) = 1$, and the individual with the highest total cost will have $Rank^C(s) = |\mathcal{S}|$. Equally, the individual s with highest diversity contribution will have $Rank^D(s) = 1$. Finally, the biased fitness, given by equation (39), is calculated using the ranks. n^{ELI} is the number of elite individuals to survive to the next generation. If n^{ELI} equals 0, the cost and diversity ranks are given equal weight and if n^{ELI} equals $|\mathcal{S}|$, the rank is set based on the cost rank only. Hence, the composition of the total population $\mathcal{S}$ is influenced by how diversity is valued relative to the total cost because survivor selection is done based on the biased fitness.

$$BF(s) = Rank^C(s) + \left(1 - \frac{n^{ELI}}{|\mathcal{S}|}\right) Rank^D(s) s \in \mathcal{S} \quad (39)$$

4.3. Constructing the initial population

The main idea behind the construction of the initial population is the following: If the rental car destination, the operator, and the relocation sequence chromosomes are given, the remaining problem, i.e. to determine the transport request assignment and the service vehicle routes, is similar to a dial-a-ride problem (DARP). The first three chromosomes determine all the rental cars and charging stations each operator has to visit, including the visit order, and can therefore be used to formulate transport requests. Each transport request is associated with the operator requesting transport by the variable $\eta_r(s)$, which is equal to d if operator d requires transport request r. By specifying time windows for the formulated transport requests, solution methods used for DARP can be used to construct the transportation request assignment chromosome and the route chromosome.

The upper and lower limit for the time window of the origin node of transport request r are denoted $l_r^o(s)$ and $u_r^o(s)$ for individual s, respectively. Similarly, the upper and lower limit for the destination node is given by $l_r^d(s)$ and $u_r^d(s)$. The time windows are determined using travel times between nodes and the time windows are set by considering the minimum possible time required by the operator to either get to the origin node (lower limit) or finalize all charging after the destination node (upper limit) as described in Algorithm 2. We use the notation $r-1$ and $r+1$ to denote the transport request directly prior to and after r for operator $\eta_r(s)$.

Finding the transport request assignment and route chromosomes by solving the subproblem as a DARP is done whenever new individuals are created in the HGSADC. However, the DARP itself is NP-hard (Healy and Moll, 1995). Hence, heuristics are needed. Low computational time is prioritized potentially at the expense of solution quality because the algorithm is executed many times. The static DARP as discussed here, as well as variations of the problem, are well studied in the literature. An extensive literature survey of model formulations and heuristic solution methods for the DARP is presented by Cordeau and Laporte (2007). Although this survey is somewhat dated, it includes the majority of significant contributions to solution methods for the static DARP relevant for this problem. More recent papers are Parragh and Schmid (2013), Kirchler and Wolfler Calvo (2013), Braekers et al. (2014), Osaba et al. (2015), Gschwind and Drexl (2016) and (Masmoudi et al., 2017). Because of a simple formulation and low computational effort, the cluster first sweep second algorithm proposed by Xiang et al. (2006) is employed as a construction algorithm for the DARP subproblem.

The initial population is created by the construction heuristic described in Algorithm 3. An individual s is created in four steps. Steps 1 to 3 create chromosomes $\alpha(s)$, $\beta(s)$, and $\gamma(s)$, respectively, and Step 4 creates the remaining chromosomes $\delta(s)$ and $\varepsilon(s)$ by solving a DARP. In the first step, each rental car i is assigned a destination $\alpha_i(s)$. A list $G_i(s)$ of the n^{CS} closest charging stations to rental car i is created. $G_i(s)$ is updated to only include charging stations with available charging slots and charging stations within the range reachable with the given battery level of the rental car. The destination g for rental car i is chosen from $G_i(s)$ with probability ρ_g, but the charging of the rental car can also be postponed. The probability $\rho_g > \rho_{g+1}$, i.e. the probability of choosing the closest charging station is higher than the probability of choosing the second closest, which is higher than the probability of choosing the third closest, etc. The probability of postponing the rental car is lower than the probability of chosing destination $g = n^{CS}$.

To guide how the remaining chromosomes are set, a pseudo time for each operator is used to avoid solutions with large infeasibilities in the total time constraints. Since only a small part of the problem is determined after the first step of Algorithm 3, the destination of each rental car is used to estimate the total duration of the relocation. The travel time between i and j is given

Algorithm 3 Construction heuristic.

individualsCreated $s \leftarrow 0$
while $s < \mu K^{INIT}$ **do**
  STEP 1: Select destination pattern
  Create a sorted list $G_i(s)$ with the closest charging stations to rental car i
  $CS_i^{cap}(s) \leftarrow$ Number of available charging slots at charging station i
  for each rental car $i \in N^{EV}$ **do**
    Choose charging station, *with available charging slot*, $g \in G_i(s)$ with probability ρ_g, where $\rho_g > \rho_{(g+1)}$, or postpone charging
    if charging not is postponed **do**
      $\alpha_i(s) \leftarrow g$
      $CS_g^{Cap}(s) \leftarrow CS_g^{Cap}(s) - 1$
    end if
  end do

  STEP 2: Select relocation assignment pattern
  with probability ρ^{assign} **do**
    Apply Algorithm 3.1 to create relocation assignment pattern with low operator cost
  else do
    Apply Algorithm 3.2 to create relocation assignment pattern with low travel cost
  end do

  STEP 3: Select relocation sequence pattern
  for each operator $d \in D$ **do**
    Create set of rental cars that are relocated by each operator, $\mathcal{F}_d(s) = \{i | \beta_i(s) = d\}$
    while $\mathcal{F}_d(s) \neq \emptyset$ **do**
      with probability ρ^{seq} **do**
        add the rental $i \in \mathcal{F}_d(s)$ that is closest to the position of the operator to $\gamma_d(s)$
      else do
        add random rental car $i \in \mathcal{F}_d(s)$ to $\gamma_d(s)$
      end do
      Remove i from $\mathcal{F}_d(s)$
    end do
  end do

  STEP 4: SOLVE THE DIAL-A-RIDE PROBLEM WITH THE THREE FIRST CHROMOSOMES AS INPUT
  Formulate transport requests and determine time windows using Algorithm 2
  Create list $L(s)$, the node visit sequence sorted by the end time of the time window to serve all transport requests
  Create conflict table $C(s)$ of the transport requests with conflicting time windows
  Create initial service vehicle routes using Algorithm 3.3, use routes to set $\delta_r(s)$ and $\varepsilon_v(s)$

  Educate generated individual with probability $\rho_{construct}^{EDU}$
  if generated individual is infeasible **then**
    Repair individual with probability $\rho_{construct}^{REP}$
  end if

  individualsCreated $s \leftarrow s + 1$
end while

by T_{ij}. However, this time only accounts for the time spent while the rental car is relocated. In addition to this, the operator must be transported to the rental car and picked up at the charging station. This may take longer than the travel times between the pick up point and the drop off point for two reasons. First, the operator may have to wait by the charging station before a service vehicle arrives to pick him/her up. Second, other rental cars or charging stations might be visited by the service vehicle on the way to the operator's drop off point. To account for this, the relocation time is multiplied by a constant $K^{pseudo} > 1$. Results indicate that setting K^{pseudo} dynamically contributes to the diversity of the generated population. Hence, for this problem $K^{pseudo} = 1.5$ initially and is

Algorithm 3.1 Relocation assignment with low operator cost.

```
d ← 1
Create sorted list H(s) of rental cars that are relocated, shortest relocation time first
while H(s) ≠ ∅ do
   EV ← first element of H(s)
   if pseudo time of d ≤ T̄ when EV is assigned to d then
      β_EV(s) ← d
      Update pseudo time and remove EV from H(s)
   else if (d + 1 ≤ |D|) then
      d ← d + 1
   else
      Set α_i(s) to postpone for the remaining rental cars i in H(s)
      H(s) ← ∅
   end if
end do
```

Algorithm 3.2 Relocation assignment with low travel cost.

```
for each charging station i ∈ N^CS do
Create set of rental cars being relocated to charging station H_i(s)
   d ← 1
   while H_i(s) ≠ ∅ do
      EV ← random rental car from H_i(s)
      β_EV(s) ← d
      Remove EV from H_i(s)
      d ← d + 1
      if d > |D| then
         d ← 1
      end if
   end do
end do
```

increased by 0.5 four times during the construction algorithm. The pseudo time can be expressed as:

$$t_d^{pseudo}(s) = \sum_{i \in N^{EV} | \beta_i(s) = d} K^{pseudo} T_{i\alpha_i(s)} \qquad s \in \mathcal{S}, d \in \mathcal{D} \tag{40}$$

Step 2 assigns an operator to relocate each rental car. With probability ρ^{assign} the operator is chosen with priority on using a low number of operators as presented in Algorithm 3.1. Rental cars are assigned to operators with a greedy algorithm, adding rental cars to the operator as long as the pseudo time of the operator does not exceed the planning time $\overline{T}$. If the planning time for an operator is exceeded by adding a rental car, that car is instead added to the next operator. Alternatively, operators are assigned with priority on reducing the distance travelled by service vehicles, presented in Algorithm 3.2. This is done by attempting to assign rental cars to operators so that cars relocated to the same charging station are relocated by different operators to allow service vehicles to do fewer charging station visits and thereby possibly travel a shorter distance. To do this, all rental cars assigned to the same charging station are assigned to different operators. Only if the number of operators is limited, multiple rental cars are relocated to the same charging station by the same operator.

The third step of the algorithm sets the relocation order of the cars assigned to each operator. Until all cars have been included in the sequence, a new car is added to the end of the sequence. The car closest to the position of the operator after the previous relocation is added with a probability ρ^{seq}, otherwise a random car is added. Using the time windows, the origin and destination nodes of the transport requests are sorted, lowest upper limit first, in a list $L(s)$. The transport requests are split into origin and destination nodes because a service vehicle assigned to that request does not necessarily drive directly from the origin to the destination, other nodes can be visited in-between. Requests that are in conflict, i.e. not possible to fulfill with the given time windows on the same route, are stored in a conflict table $C(s)$. Using $L(s)$ and $C(s)$, routes are created using the sweep heuristic proposed by Xiang et al. (2006). The algorithm iterates through the list $L(s)$ adding unvisited nodes that are not in conflict with any of the nodes already in the route. Furthermore, destination nodes are only added to the route if the origin node already is in the route. After all elements of $L(s)$ are searched, a new route is created and all unvisited nodes in $L(s)$ are searched and added by the same criteria. The resulting assignment of transport requests to service vehicles and service vehicle routes are stored in the transport request assignment and route chromosomes, respectively.

4.4. Parent selection and crossover

The offspring generation scheme of the HGSADC selects two parent individuals, s_1 and s_2, and generates one offspring s_{new}. Each parent is selected by a binary tournament, i.e. randomly picking two individuals from the entire population and choosing the one with best biased fitness as the parent, as proposed by Vidal et al. (2012). The four-stepped crossover operator is described in Algorithm 4. In the first step (Step 1), the genes to inherit from each parent are decided. This is done by randomly dividing the set of rental cars in three disjoint sets: Λ_1, Λ_2, and Λ_{mix} containing rental cars inheriting patterns from s_1, s_2, and both, respectively.

Step 2 inherits data from s_1. The destination and operator for all rental cars in Λ_1 are copied directly from s_1 to s_{new}. Two random cut-off points υ_1 and υ_2, $\upsilon_1 \leq \upsilon_2$, are picked for the set Λ_{mix},

Algorithm 4 Crossover operator.

STEP 0: Inheritance rule
Pick two random numbers between 0 and $|N^{EV}|$ according to a uniform distribution.
Let n_1 and n_2 be the smallest and the largest of these numbers, respectively
Randomly select n_1 rental cars to form the set Λ_1
Randomly select $n_2 - n_1$ remaining rental cars to form the set Λ_2
The remaining $|N^{EV}| - n_2$ rental cars make up the set Λ_{mix}

STEP 1: Inherit data from s_1
for each rental car i belonging to the set Λ_1 **do**
  Copy the destination $\alpha_i(s_1)$ to $\alpha_i(s_{new})$ and the operator $\beta_i(s_1)$ to $\beta_i(s_{new})$
end for
Pick two random cut-off points υ_1 and υ_2 dividing the set Λ_{mix}
for each rental car i in the subset between υ_1 and υ_2 **do**
  Copy the destination $\alpha_i(s_1)$ to $\alpha_i(s_{new})$ and the operator $\beta_i(s_1)$ to $\beta_i(s_{new})$
end for
Copy relocation sequence $\gamma_d(s_1)$ to $\gamma_d(s_{new})$ for all drivers and rental cars so far inherited from s_1

STEP 2: Inherit data from s_2
for each rental car $i \in \Lambda_2 \cup \Lambda_{mix}$ **do**
  if $\alpha_i(s_{new}) = \emptyset$ and destination assignment not violates capacity at charging station $\alpha_i(s_2)$ **do**
    Copy the destination $\alpha_i(s_2)$ to $\alpha_i(s_{new})$
    Copy the relocation assignment $\beta_i(s_2)$ to $\beta_i(s_{new})$
  else if $\alpha_i(s_{new}) = \emptyset$ **do**
    Assign rental car i to the closest available charging station or postpone
    if rental car i not postponed **do**
      Copy the operator $\beta_i(s_2)$ to $\beta_i(s_{new})$
    end if
  end if
end do
Copy the relocation sequence from s_2 to s_{new} for all drivers and rental cars inherited form s_2
Apply improvement heuristic to improve relocation sequence pattern

STEP 3: Route service vehicles
Apply step 4 from construction heuristic (Algorithm 3) to formulate transport requests and route service vehicles

Algorithm 3.3 Route construction heuristic (Xiang et al., 2006).

for each unvisited node i in list $L(s)$ **do**
  if node i is a pick up site **then**
    Add node i as the first pick up site in a new route
    for each unvisited node j after node i in list $L(s)$ **do**
      if node j is a pick up site and does not *conflict* with any request already in this route or node j is a delivery site and its corresponding pick up site is already in this route **then**
        Add node j to the tail of this route
      end if
    end do
  end if
end do

and the destination and the operator for the rental cars in the sequence between these cut-off points are copied from s_1 to s_{new}. Furthermore, the relocation sequence for the rental cars inherited from s_1 are copied directly from s_1 to s_{new}.

In Step 3, data is inherited from s_2. For all the remaining rental cars in Λ_2 and Λ_{mix}, the destination is copied to s_{new} if capacity constraints on the charging stations are not violated. If the capacity constraints are violated, the rental car is assigned to the closest charging station with available charging slots. The operator is copied directly. The relocation sequence are copied directly from s_2 to s_{new}, except for the rental cars already in $\gamma_d(s_{new})$. This ensures that all rental cars are relocated without conflict between operators. An improvement heuristic minimizing the travel distance of the operator is then applied to improve the relocation sequence patterns.

Finally, in Step 4, transport requests and service vehicle routes are constructed using Step 4 from the construction heuristic (Algorithm 3.3). Due to the design of the crossover operator, offspring individual s_{new} is feasible except in the time constraints and number of service vehicles used.

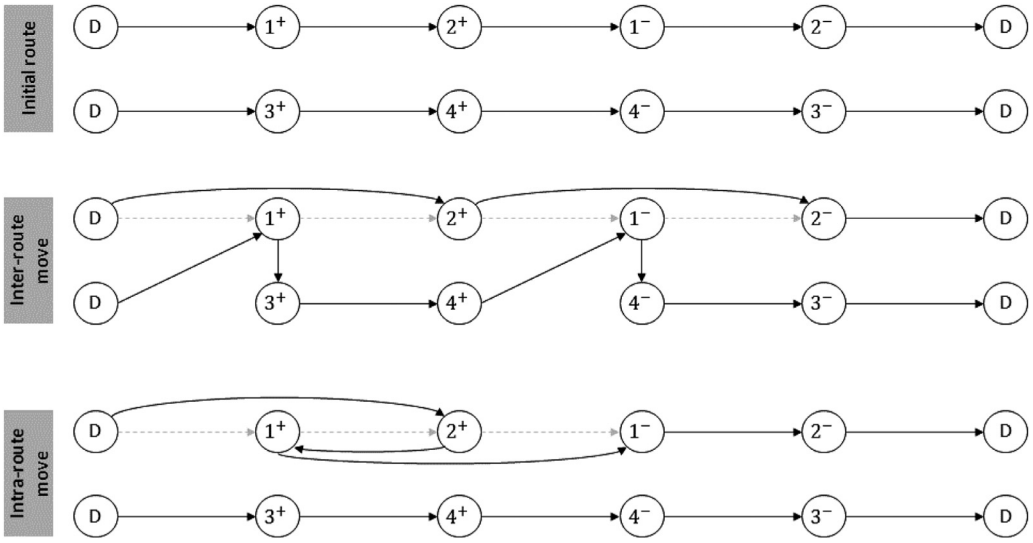


Fig. 3. Illustration of relocate-operator, inter and intra-route moves allowed. 1^+ and 1^- denote pick up and drop off nodes of transport request 1, respectively. Equivalent notation applies for transport requests 1 to 4. This illustration shows an inter-route move and and intra-route move of request 1.

4.5. Education

The education phase aims to decrease the total cost of an individual by improving the relocation sequence, transport request assignment, and route chromosomes. As different rental car destination and operator chromosomes are evaluated as a part of the overall HGSADC, these are not altered in the education module. Simple improvement operators are sought in order to run a large number of improvement iterations with little computational effort. The education module also includes a repair procedure to make infeasible individuals feasible.

Neighbors are defined by a *neighborhood operator* based on Braekers et al. (2014). A transport request is removed from its current position in a route and inserted in either another position in the same route or in a different route. The neighborhood operator is illustrated in Fig. 3. Transport requests can be inserted in positions that require modification of the relocation sequence chromosome. This happens if the modified routes force an operator to visit the rental cars in a different order than the order defined in the relocation sequence chromosome. A change in this chromosome also requires the transport requests to be modified, so that the transport requests align with the flow of operators set by the relocation sequence. Furthermore, if an improving inter-route move is found, the transport request assignment chromosome is modified so that it captures that a new service vehicle relocates the transport request. A *first improvement* strategy is implemented, meaning that the first improvement found is accepted and the search for better solutions continues by considering the next transport request. First improvement is chosen because it has been shown that there is little difference between *best improvement* and first improvement (Breedam, 2001). The education procedure terminates when no improvements are found.

Individuals that are feasible after education is performed are referred to as *naturally feasible individuals*. If an individual is infeasible, the individual is *repaired* with probability ρ^{REP} attempting to make it feasible. This is done by multiplying the penalty parameters by ten and running the education procedure again. If the individual still is infeasible, the penalty parameters are multiplied by 100 and the education procedure executed. If the individual still is infeasible, a module forcing the individual to become feasible is employed.

The force feasibility module consists of two parts. The first part repairs individuals that are using too many service vehicles and the second individuals that exceeds the maximum time limit. If too many service vehicles are used, the module searches through all routes to find the vehicle that handles the fewest transport requests. Then, all the rental cars corresponding to these transport requests are postponed. The postponed rental cars are removed from the relocation sequence chromosome of the relevant operators and the DARP is re-solved with the updated chromosomes to determine the transport request assignment and route chromosomes. This procedure is repeated until enough rental cars are postponed so that the service vehicle limit is no longer exceeded. If an individual is exceeding the maximum time limit constraint, all routes are searched through to find the route with the longest duration. Then, the rental car corresponding to the last transport request in the route is postponed. Similar to the first part of repair, the relocation sequence chromosome is updated and the DARP re-solved. The procedure is repeated until the individual no longer exceeds the maximum relocation time. Note that even though repair guarantees feasibility, the procedure is not run for all individuals. Hence, infeasible solutions are still present in the population.

4.6. Population management

Three population management schemes are employed to improve the performance of the genetic search algorithm. *Survivor selection* is performed to increase the quality of the population by removing the worst quality individuals based on the biased fitness. Survivor selection is executed on a population whenever the number of individuals in the population reaches its maximum limit $\mu + \lambda$. Individuals are removed until there are μ individuals left.

Penalty parameter adjustment updates the penalty parameters for every 100 iterations with the goal of attaining the target ratio ζ^{REF} of feasible individuals. If the proportion of feasible individuals is below 5% less than the target ratio, the penalty parameter is adjusted up by $\xi^{UP} > 1$. Similarly, if the target ratio is above 5% more than the target ratio the penalty parameter is adjusted down by $\xi^{DOWN} < 1$.

Diversification is executed to prevent the algorithm from converging to a local optima. If no improvement is made to the best individual in I^{DIV} iterations, two thirds of the worst individuals are

Fig. 4. Illustration of an extract of an instance showing the status of rental cars and the location of charging stations and the depot. Zones around charging stations are identified by rectangles.

Table 1
Details of computer and solver used in the computational study.

Processor:	Intel(R) Core(TM) i7-6700 CPU 3.40 GHz
RAM:	32GB
Operating system:	Windows 10 Education 64-bit
Xpress-IVE version:	1.24.08 64 bit
Xpress optimizer version:	28.01.04
Mosel version:	3.10.0
Java version:	8
Maximum computational time:	3600 s

removed from each subpopulation. Then, μK^{DIV} new individuals are generated using the construction heuristic.

5. Computational study

The FFECCRP has been solved using both the commercial MIP solver Xpress and with the HGSADC algorithm, which has been implemented in Java. The hardware and software specifications of the computational study are given in Table 1.

5.1. Instances and implementation

Test instances are created based on data from the focal CSO. An extract of one of the instances is illustrated in Fig. 4 showing the status of each rental car, the location of charging stations, and the depot. All rental cars in need of charging within 30 minutes drive from the depot are considered. An overview of the size of the test instances and their parameters is shown in Table 2. Three test instances of each size with different initial distribution of rental cars are created. The letters *a*, *b*, and *c* are used to distinguish between test instances of equal size. Different test instances are used for calibration and performance testing to avoid overfitting the model and algorithm to the data.

Travel times are retrieved from Google maps and assumed equal for both service vehicles and rental cars. We assume that the ideal state is an even distribution of rental cars, that is, the ideal state of the system is set so that the number of rental cars is equal in all charging stations. The initial state in each charging station consists of a random number of cars. However, the total number of rental cars in the system is equal to the total number of rental cars in the ideal state. As an example, Fig. 4 shows an excerpt of an instance consisting of three charging station and nine

Table 2
Overview of the key parameters of the constructed test instances.

Instance	# cars to be relocated	# charging stations	# service vehicles	# operators	Planning period duration (min)
4_2	4	2	1	4	120
6_3	6	3	2	6	120
8_4	8	4	3	8	120
60_20	60	20	10	40	120
100_35	100	35	14	56	120
125_40	125	40	16	64	120
150_45	150	45	18	72	120
175_50	175	50	20	80	120
200_55	200	55	22	88	120

Table 3
Overview of the parameters used in the HGSADC and their values.

Parameter	Value	Description
μ	35	Minimum population size
λ	100	Generation size
I^{NI}	10,000	Max. number of iterations without improvement
η^{DIV}	0.2	Proportion of I^{NI}, such that $I^{DIV} = \eta^{DIV} \times I^{NI}$
η^{ELI}	0.5	Proportion of elite individuals, $n^{ELI} = \eta^{ELI} \times \lvert\mathcal{S}\rvert$
η^{CLO}	0.2	Proportion of individuals considered in diversity contribution, such that $n^{CLO} = \eta^{CLO} \times \mu$
K^{INIT}	20	Construction heuristic size factor
K^{DIV}	20	Diversification size factor
$\rho^{EDU}_{construct}$	0.75	Probability of education in construction heuristic
$\rho^{REP}_{construct}$	0.25	Probability of repair in construction heuristic
$\rho^{EDU}_{crossover}$	0.5	Probability of education in crossover
$\rho^{REP}_{crossover}$	0.5	Probability of repair in crossover
ζ^{REF}	0.6	Desired ratio of feasible individuals
w^T	2	Duration violation penalty
w^V	0.5	Number of vehicles violation penalty
ξ^{UP}	1.25	Penalty adjustment factor, up
ξ^{DOWN}	0.75	Penalty adjustment factor, down
T^{MAXRUN}	3,600	Maximum running time (seconds)

rental cars. The ideal state is thus three cars for each charging station. The initial state at each charging station (i.e., the number of cars with sufficient battery level in the zone containing the charging station) is one car for each charging stations. A possible solution would be that of relocating the two cars in need of charging in the top-left zone to the charging station in the bottom-left zone, and the two-cars in need of charging in the right-hand-side zone to its own charging station. Once the cars with depleted battery are charged, this would leave each zone with three available cars as in the ideal state.

The solutions will clearly depend on how the cost parameters are set, and especially the postponement costs and costs for deviation from the ideal state can be difficult to estimate. These cost parameters are set to reflect the relative size of costs for the focal CSO. Costs per unit time have been estimated based on operator salaries and transport costs with the service vehicles. The unit costs are then scaled with travel or planning period time to arrive at the final cost parameters. All costs associated with the service vehicle are included in the travel cost and have been estimated to a travel cost of ten cents per minute travelled. The employee cost per hour is assumed to be 10 Euros per operator, which gives a total cost of 20 Euros per operator since the planning period is 120 minutes.

The postponement and deviation costs are set by observing the CSO's notion a good solution. The deviation cost is assumed to be 10 Euros per deviation and reflects the profit loss and badwill of the trips lost due to no available cars. If handling of a rental car is postponed (i.e., not performed during the current planning period), the rental car is unavailable for users in the period following the planning period. As it may result in lower customer satisfaction as fewer rental cars are available, the cost of postponing is assumed to be 25 Euros for the instances with less than 15 cars in need of handling. For the larger instances, 50 Euros have been used as postponement cost, as preliminary testing showed that this gave more reasonable solutions. In real life, the cost parameters discussed here are dependent on the preferences of the CSO and how frequently the SFFCCRP is resolved.

The set $\mathcal{M}_i$ is defined for each node i, representing possible visits to a node i for both service vehicles and operators. Preliminary testing has revealed that setting the number of visits to charging stations equal to the lower bound plus one ensures the best trade off between solution quality and computational time. The lower bound for service vehicles at each charging station is set to the number of available charging slots at the station divided by the seat capacity of the service vehicles, rounded up to the nearest integer. Similarly for operators, the lower bound at each charging station is set to the number of available charging slots at the station divided by the total number of operators, rounded up to the nearest integer. All rental cars can only be visited once and it is desirable to allow all operators to transport all rental cars. Hence the number of allowable visits to rental cars is set to one for both service vehicles and operators.

5.2. Parameter tuning

The HGSADC relies on a set of correlated parameters and configuration choices for its key operators. Different values for each parameter are tested individually, keeping the rest of the parameters fixed. Once a parameter value is chosen, the remaining tests are performed with all prior parameter values set to the chosen values. Each test is performed five times due to the non-deterministic nature of genetic algorithms. The calibration results for the parameters are shown in Table 3.

5.3. Performance of the HGSADC

The MIP solver can only solve instances with very few cars in need of charging. All three instances both with four and six rental cars in need of charging are solved to optimality within the maximum running time of 1 h. The instances with four rental cars take on average around only one second, while the three instances with six rental cars take from 7 to 2088 s to solve. The instances with eight rental cars could not be solved to optimality with the MIP solver and returned optimality gaps from seven to 67%. Hence, the MIP solver cannot reliably produce high quality solutions for instances with more than six rental cars in need of charging. Furthermore, for instances with more than eight rental cars, the MIP solver could not even find feasible solutions within one hour, and in several cases it even fails to load the problem into memory. In comparison, the HGSADC is capable of finding the optimal solutions of all the instances with four and six rental cars in need of charging solved to optimality by the MIP solver within a few seconds.

To further investigate the performance of the HGSADC, 15 large instances of five different sizes are solved ten times each, and the average run times and the average gaps after ten minutes and after one hour execution are reported in Table 4. The table also shows the coefficients of variance of the gap and computational time after 1 h. It should be noticed that since we do not know the optimal solutions for these instances, the gaps are calculated from

Table 4
Final results of running the HGSADC on 15 instances with 100 to 200 rental cars in need of charging. Optimality gaps are calculated with respect to the best known solution.

Instance	Avg. time (s)	Avg. gap% after 600s	Avg. gap %	Coeff. of Var. gap %	Coeff. of Var. time %
100_35_a	1212.1	2.6	1.9	0.9	76.4
100_35_b	693.5	1.6	1.5	1.3	32.7
100_35_c	743.5	1.3	1.2	0.9	29.3
125_40_a	1185.8	2.7	2.0	1.6	85.5
125_40_b	1302.6	2.5	1.9	1.2	68.2
125_40_c	972.7	1.7	1.3	0.6	49.1
150_45_a	1774.9	2.6	1.4	1.0	48.8
150_45_b	760.7	1.0	1.0	0.8	26.0
150_45_c	1362.9	2.1	1.3	0.9	47.4
175_50_a	1453.7	1.8	1.2	1.0	32.2
175_50_b	1849.1	2.9	1.7	1.1	51.1
175_50_c	2137.4	1.8	0.6	0.5	31.2
200_55_a	1496.9	1.5	1.0	0.4	26.4
200_55_b	1265.5	1.0	0.6	0.5	36.7
200_55_c	2403.4	2.1	1.2	0.8	42.8
Average	1374.3	1.9	1.3	0.9	45.6

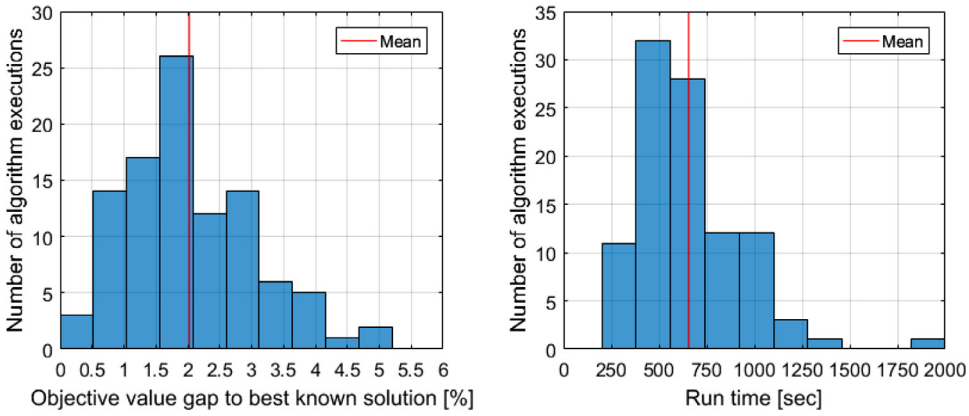


Fig. 5. Histogram of objective value gap to the best known solution and run time from running the 100_35_c instance 100 times.

the best known solution obtained among the ten runs for each instance. The average run time for the tested instances ranges from 693.5 to 2403.4 s. However, there is a relative large variation in run times indicated by the average coefficient of variance of the run time of 45.6%. One of the runs of the 100_35 instances, two of the 125_40 instances, two of the 150_45 instances, two of the 175_50, and three of the 200_55 instances ran for the maximum run time. However, all of these runs found a solution with less than 4.1% gap from the best available solution after ten minutes and less than 3.5% gap at the end of the execution, making the solutions usable for most practical purposes. There are several factors contributing to the large variation in run time. The most important reason is simply randomization. Depending on the initial individuals created and which individuals are chosen as parents, the algorithm may have to run more iterations to reach good solutions. An observation made is that the average gap to the best known solution and the coefficients of variance of the gap and run time decrease with increasing instance size. This indicates that the algorithm scales well to large instances.

The gap after ten minutes (600 s) is reported because we assume that running the algorithm for a maximum of ten minutes is desirable in real life scenarios. The average gap to the best found solution after ten minutes is 1.9% with no averages above 2.9%. Furthermore, the average coefficient of variation of the objective value after ten minutes is equal to the value at the end of the algorithm execution at 0.9%. This demonstrates that the algorithm is able to produce acceptable solutions reliably within ten minutes. The average gap at the end of the algorithm execution is 1.3%. For the largest instances with 200 rental cars, these numbers are even lower, with an average gap of less than 1.0% and coefficient of variation of 0.6%. These results are a clear indicator of the capabilities of the algorithm to produce consistent, high quality solutions for realistic carsharing systems.

To gain a deeper understanding of the stability of the algorithm, 100 runs on the 100_35_c instance have been executed. This instance is chosen randomly as the purpose is only to show the algorithm's performance. The results of these runs are presented in the histograms in Fig. 5. As can be seen from the plot, the mean objective value gap to the best known solution is 2.0%. 58 out of the 100 solutions found have equal or smaller gap than the mean. Of the 42 solutions with objective value gap above the mean, 39 are below 4.0%. The remaining three solutions have gaps of 4.3, 5.1, and 5.1%, respectively. The mean run time of the algorithm is 655.7 s. 61 of the 100 algorithm executions completed in less or equal run time as the mean. 91 of the runs completed in less than 1000 s. Of the remaining nine algorithm runs, eight completed in less than 1314 s and one outlier required 1970 s to complete.

Finally, the HGSADC consists of many modules making it a relatively complex algorithm to design and implement. To rationalize the added complexity, it is essential that the algorithm provides a significant improvement in solution quality and/or computational time compared to the construction heuristic or a simple GA. To provide evidence of the value of the HGSADC, Table 5 compares the results of running the HGSADC with different modules on four instance classes. The average computational times and gaps from the best known solution are reported. In the first and the second columns, the results of running only the construction module of

Table 5
Average running time and optimality gap reported for runs of the construction heuristic without education and repair, the construction heuristic with education and repair, the HGSADC without education and repair, and the HGSADC with education and repair, respectively. Optimality gaps are calculated with respect to the best known solution.

Instance	CH^1		$CH+E/R^2$		$HGSADC^3$		$HGSADC+E/R^4$	
	Time (s)	Gap %	Time (s)	Gap %	Time (s)	Gap %	Time (s)	Gap %
6_3*	≈0	0.1	3.6	0.0	28.4	0.0	22.8	0.0
8_4	≈0	18.6	≈0	11.0	31.2	2.9	26.4	0.7
60_20	N/A^5	N/A^5	5.6	8.3	2757.8	5.8	593.0	1.4
100_35	N/A^5	N/A^5	27.8	5.9	1417.8	3.9	769.4	2.0

*Proven optimal.
1: The construction heuristic without education and repair.
2: The construction heuristic with education and repair
3: The construction heuristic with education and repair, the HGSADC iterations without education and repair
4: The HGSADC with all configurations.
5: No feasible solution found.

Table 6
Comparison of costs, and number of deviations and postponements when repositioning is omitted and when repositioning is performed. The numbers are the average of five runs with the HGSADC. The test instance has 100 rental cars in need of charging. The average change when repositioning is considered is reported compared to when repositioning is not considered.

	No repositioning	With repositioning	Change %
Number of postponed cars	46.6	42.6	-8.6
Number of deviations	84.6	79.0	-6.6
Distance driven [km]	562.5	581.1	3.3
Number of service vehicles	12.8	13.8	7.8
Number of operators used	50.4	53.8	6.8
Postponement cost	2330.0	2130.0	-8.6
Deviation cost	846.0	790.0	-6.6
Travel cost	112.5	116.5	3.3
Service vehicle cost	256.0	276.0	7.8
Operator cost	1008.0	1076.0	6.8
Real cost	4552.5	4388.2	-3.6

the algorithm without and with education and repair, respectively, are presented. Then, the third column presents the results of running the algorithm with education and repair in the construction heuristic but without education and repair after an offspring is created. Finally, the results of the full algorithm are presented in the fourth column. Firstly, we observe that including education and repair in the construction heuristic enables the algorithm to find feasible solutions for all instances used in this comparison. Secondly, it is clear that each module added to the algorithm contributes to a significant reduction in the gap to the best known solution for all the instances. However, the improvements come at the cost of added computational time in the three first columns. Lastly, the rightmost column illustrates that the full HGSADC-algorithm significantly outperforms the other configurations of the HGSADC as it finds the lowest gap to the best known solution for all instances in considerably less time than the HGSADC without education and repair. The reduced computational time from the third to the fourth column in spite of added complexity is a result of the full algorithm finding high quality solutions faster leading to fewer iterations.

Based on these results we conclude that the algorithm is able to produce solutions with stable quality within a reasonable time for most executions of the algorithm. In addition, the results demonstrate that the HGSADC is suitable for problems with complex synchronization constraints including spatial and temporal interdependencies.

6. Economical implications

A central hypothesis of this paper is that combining necessary daily operations with repositioning will increase the operational costs of the CSO marginally, while harvesting the full benefits of repositioning. To investigate the effect of repositioning, two different configurations of the HGSADC are compared. In the first configuration all cars are either relocated to the closest charging station or postponed. This represents the charging procedure without repositioning. In the second configuration the full HGSADC is run. An instance with 100 rental cars to relocate has been run five times for each configuration and the average costs, number of deviations, and postponements are reported in Table 6. The average change when repositioning is considered is reported in the fourth column of the table, compared to when repositioning is not considered.

Even though the operational costs increase when repositioning is considered, the total cost of the system decreases with 3.6% when repositioning is performed when using the cost parameters described in Section 5. The reduction in total costs can be attributed to the decreased number of postponements and deviations. As the total cost (objective function) includes lost profits when deviations are present and cars are postponed, the total cost to a large degree captures the profit effect of the repositioning operations. Hence, the 3.6% decrease in total costs can be directly transferred to gross profit margin improvement, thereby representing a significant improvement of the economic viability of the CSO.

For some CSOs, charging without repositioning might closer resemble their operational mode, or the deviation cost in these systems may be too cumbersome to derive. When only considering charging at the closest available charging station, the HGSADC can simulate a situation where only charging is performed. When the HGSADC is run with no repositioning, the computational time is reduced by 61.2% and the stability of the solutions increase due to a smaller search space. This implies that the HGSADC can be

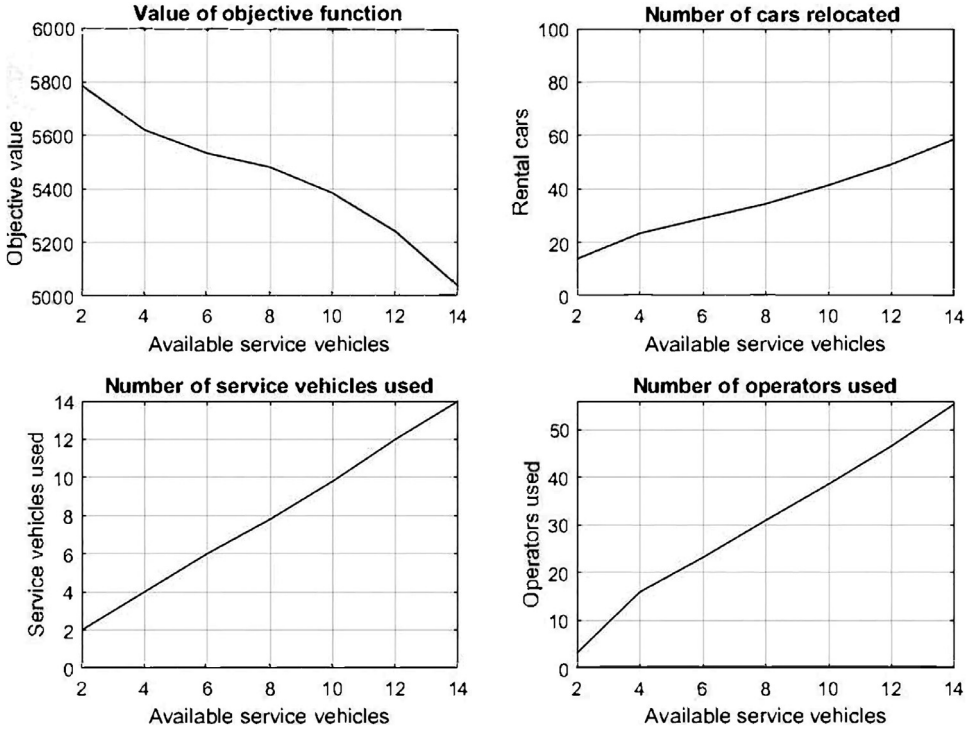


Fig. 6. The plots of the average objective value, number of rental cars relocated, number of vehicles used, and number of operators used when the number of available operators and service vehicles are varied. The number of available operators increases linearly by a factor of four as the number of service vehicles increases. There are 100 rental cars in need of charging.

highly valuable even if repositioning is not done in conjunction with charging.

It is assumed in our model and solution method that a given number of operators and service vehicles are available at the depot. By varying these numbers, the marginal benefit of added operators and service vehicles can be found. Figure 6 shows the plot of the objective value, number of rental cars relocated, number of vehicles used, and number of operators used by the HGSADC solution when the number of available service vehicles is varied. The number of operators is increased linearly with the number of service vehicles. We have assumed a service vehicle capacity of four operators, thus keeping a one to four ratio between service vehicles and operators. A clear insight from the figure is that the number of available operators and service vehicles used for charging and repositioning has a significant effect on the profitability of the system. In our case, the objective value decreases as the number of operators and service vehicles increase as the benefit of charging more rental cars exceeds the added costs of service vehicles and operators, and the increased travel costs. By running the algorithm using their own cost estimates, CSOs can find the marginal benefits applicable for their system. Using historical data, CSOs can use the algorithm to determine the strategically optimal number of operators to hire and service vehicles to invest in.

A strength of the proposed formulation is the division of the business area into smaller areas surrounding the charging stations. By varying the size of the areas, the model can indirectly factor in the flexibility of users, as described by Correia et al. (2014). Furthermore, if cars that require charging are made unavailable in the booking system, the HGSADC can easily be applied to charging and repositioning throughout the day, without modifications. These operations will then be performed based on information about charging requirements, traffic, and states available when the algorithm execution is started. We expect large scale carsharing systems to realize the biggest benefit from employing the algorithm. This is because these systems can have a higher density of cars enabling higher utilization of service vehicles but added planning complexity.

7. Conclusion

This paper presents a mathematical formulation and a genetic algorithm for the Free-Floating Electric Carsharing Charging and Repositioning Problem (FFECCRP). Many companies already have a fleet of service vehicles and staff to move rental cars to charging stations. Henceforth, considering repositioning to improve the distribution of cars in the system while moving cars to charging stations shows potential to realize the benefits of repositioning without a large increase in operational costs. Ultimately, this will improve the profits and the economic viability of carsharing systems. A novel Mixed Integer Programming (MIP) model is developed to solve the problem. Because the MIP model is computationally cumbersome to solve, a Hybrid Genetic Search with Adaptive Diversity Control (HGSADC) is proposed.

The HGSADC is capable of solving instances with up to 200 rental cars in need of charging yielding seemingly high quality solutions in an average computational time of less than 2400 s. The stability of the algorithm is acceptable for practical purposes with an average gap to the best known solution of 1.3% and an objective value coefficient of variance of 0.9%. When comparing solutions from the HGSADC with solutions produced by the algorithm when repositioning not is considered, the number of postponed rental cars and the number of deviations decrease by 9.4% and 7.1%, respectively. A reduction in postponed cars implies that more rental cars are relocated when repositioning is considered. The reduction in deviations imply that the rental cars relocated are relocated to more favorable destinations considering the distribution of rental cars in the system when repositioning is considered. Hence, we conclude that combining charging with repositioning is beneficial for CSO.

The HGSADC developed for the FFECCRP demonstrates the performance of HGSADC-algorithms on routing problems with complex synchronization constraints. The FFECCRP consists of two closely linked routing problems, one for the routing of rental cars to charging stations and one for routing service vehicles transporting operators to rental cars and from charging stations. As the drop off time and location of an operator affects the time and location of his/her pick up, spatial and temporal interdependencies emerge. The work in this paper outline the merit of genetic algorithms for solving this complex problem type. In conclusion, solving the FFECCRP with the HGSADC produces high quality solutions within reasonable computational time for realistic problem sizes.

Acknowledgment

We are grateful to the reviewers, whose comments helped us improve the paper. This research was carried out with financial support from the DynamITe project (project number 246825) funded by the Research Council of Norway.

References

Borthen, T., Loennechen, H., Wang, X., Fagerholt, K., Vidal, T., 2018. A genetic search-based heuristic for a fleet size and periodic routing problem with application to offshore supply planning. Eur. J. Transp. Logist. 7, 121–150.

Boyaci, B., Zografos, K., Geroliminis, N., 2015. An optimization framework for the development of efficient one-way car-sharing systems. Eur. J. Oper. Res. 240 (3), 718–733.

Boyaci, B., Zografos, K.G., Geroliminis, N., 2017. An integrated optimization-simulation framework for vehicle and personnel relocations of electric carsharing systems with reservations. Transp. Res. Part B 95, 214–237.

Braekers, K., Caris, A., Janssens, G.K., 2014. Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots. Transp. Res. Part B 67, 166–186.

Breedam, A.V., 2001. Comparing descent heuristics and metaheuristics for the vehicle routing problem. Comput. Oper. Res. 28 (4), 289–315.

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

Bulhões, T., Hà, M., Martinelli, R., Vidal, T., 2018. The vehicle routing problem with service level constraints. Eur. J. Oper. Res. 265, 544–558.

Cepolina, E.M., Farina, A., 2012. A new shared vehicle system for urban areas. Transp. Res. Part C 21 (1), 230–243.

Cordeau, J.-F., Laporte, G., 2007. The dial-a-ride problem: models and algorithms. Ann. Oper. Res. 153 (1), 29–46.

Correia, G.H.A., Antunes, A.P., 2012. Optimization approach to depot location and trip selection in one-way carsharing systems. Transp. Res. Part E: Logist.Transp. Rev. 48 (1), 233–247.

Correia, G.H.A., Jorge, D.R., Antunes, D.M., 2014. The added value of accounting for users' flexibility and information on the potential of a station-based one-way car-sharing system: an application in Lisbon, Portugal. J. Intell. Transp. Syst. 18 (3), 299–308. doi:10.1080/15472450.2013.836928.

Gschwind, T., Drexl, M., 2016. Adaptive large neighborhood search with a constant-time feasibility test for the dial-a-ride problem. Discussion paper number 1624.

Hamming, R.W., 1950. Error detecting and error correcting codes. Bell Labs Tech. J. 29 (2), 147–160.

Hansen, R., Pantuso, G., 2018. Pricing car-sharing services in multimodal transportation systems: an analysis of the cases of copenhagen and milan. Lecture Notes Comput. Sci..

Healy, P., Moll, R., 1995. A new extension of local search applied to the Dial-A-Ride Problem. Eur. J. Oper. Res. 83 (1), 83–104.

Herbawi, W., Knoll, M., Kaiser, M., Gruel, W., 2016. An evolutionary algorithm for the vehicle relocation problem in free floating carsharing. In: 2016 IEEE Congress on Evolutionary Computation, CEC 2016, pp. 2873–2879. doi:10.1109/CEC.2016.7744152.

Herrmann, S., Schulte, F., Voß, S., 2014. Increasing acceptance of free-floating car sharing systems using smart relocation strategies: a survey based study of car2go Hamburg. In: International Conference on Computational Logistics. Springer, pp. 151–162.

Jorge, D., Molnar, G., Correia, G.H.A., 2015. Trip pricing of one-way station-based carsharing networks with zone and time of day price variations. Transp. Res. Part B 81, 461–482.

Kaspi, M., Raviv, T., Tzur, M., 2014. Parking reservation policies in one-way vehicle sharing systems. Transp. Res. Part B 62, 35–50.

Kek, A.G.H., Cheu, R.L., Meng, Q., Fung, C.H., 2009. A decision support system for vehicle relocation operations in carsharing systems. Transp. Res. Part E: Logist.Transp. Rev. 45 (1), 149–158.

Kirchler, D., Wolfler Calvo, R., 2013. A granular tabu search algorithm for the dial-a-ride problem. Transp. Res. Part B 56, 120–135.

Kuhne, K.S., Rickenberg, T.A., Breitner, M.H., 2016. An optimization model and a decision support system to optimize car sharing stations with electric vehicles. In: Lübbecke, M., Koster, A., Letmathe, P., Madlener, R., Peis, B., Walther, G. (Eds.), Operations Research Proceedings 2014: Selected Papers of the Annual International Conference of the German Operations Research Society (GOR), RWTH Aachen University, Germany, September 2-5, 2014. Springer International Publishing, Cham, pp. 313–320. doi:10.1007/978-3-319-28697-6_44.

Masmoudi, M.A., Braekers, K., Masmoudi, M., Dammak, A., 2017. A hybrid genetic algorithm for the heterogeneous dial-a-ride problem. Comput. Oper. Res. 81, 1–13.

Nourinejad, M., Zhu, S., Bahrami, S., Roorda, M.J., 2015. Vehicle relocation and staff rebalancing in one-way carsharing systems. Transp. Res. Part E 81, 98–113.

Osaba, E., Onieva, E., Diaz, F., Carballedo, R., Lopez, P., Perallos, A., 2015. An asymmetric multiple traveling salesman problem with backhauls to solve a dial-a-ride problem. In: 2015 IEEE 13th International Symposium on Applied Machine Intelligence and Informatics (SAMI), pp. 151–156. doi:10.1109/SAMI.2015.7061865.

Parragh, S.N., Schmid, V., 2013. Hybrid column generation and large neighborhood search for the dial-a-ride problem. Comput. Oper. Res. 40 (1), 490–497.

Repoux, M., Boyaci, B., Geroliminis, N., 2015. Simulation and optimization of one-way car-sharing systems with variant relocation policies. Transportation Research Board 94th Annual Meeting.

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

Schulte, F., Voß, S., 2015. Decision support for environmental-friendly vehicle relocations in free-floating car sharing systems: the case of car2go. Procedia CIRP 30, 275–280.

Vidal, T., Crainic, T.G., Gendreau, M., Lahrichi, N., Rei, W., 2012. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. Oper. Res. 60 (3), 611–624. doi:10.1287/opre.1120.1048.

Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2013. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. Comput. Oper. Res. 40, 475–489.

Weikl, S., Bogenberger, K., 2013. Relocation Strategies and Algorithms for Free-Floating Car Sharing Systems. IEEE Intell. Transp. Syst. Mag. 5 (4), 100–111. doi:10.1109/MITS.2013.2267810.

Weikl, S., Bogenberger, K., 2015. A practice-ready relocation model for free-floating carsharing systems with electric vehicles–mesoscopic approach and field trial results. Transp. Res. Part C 57, 206–223.

Xiang, Z., Chu, C., Chen, H., 2006. A fast heuristic for solving a large-scale static dial-a-ride problem under complex constraints. Eur. J. Oper. Res. 174 (2), 1117–1139. doi:10.1016/j.ejor.2004.09.060.

Zhao, M., Li, X., Yin, J., Cui, J., Yang, L., An, S., 2018. An integrated framework for electric vehicle rebalancing and staff relocation in one-way carsharing systems: Model formulation and lagrangian relaxation-based solution approach. Transp. Res. Part B 117, 542–572.