Exact solutions to a carsharing pricing and relocation problem under uncertainty

Pantuso, Giovanni

Published in:
Computers and Operations Research

DOI:
10.1016/j.cor.2022.105802

Publication date:
2022

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

Document license:
CC BY

Citation for published version (APA):
A B S T R A C T

In this article we study the problem of jointly deciding carsharing prices and vehicle relocations. We consider carsharing services operating in the context of multi-modal urban transportation systems. Pricing decisions take into account the availability of alternative transport modes, and customer preferences with respect to these. In order to account for the inherent uncertainty in customer preferences, the problem is formulated as a mixed-integer two-stage stochastic program with integer decision variables at both stages. We propose an exact solution method for the problem based on the integer L-Shaped method which exploits an efficient exact algorithm for the solution of the subproblems. Tests on artificial instances based on the city of Milan illustrate that the method can solve, or find good solutions to, moderately sized instances for which a commercial solver fails. Furthermore, our results suggest that, by adjusting prices between different zones of the city, the operator can attract significantly more demand than with a fixed pricing scheme and that such a pricing scheme, coupled with a sufficiently large fleet, significantly reduces the relevance of staff-based relocations. A number of issues, that remain to be addressed in future research, are pointed out in our conclusions.

Keywords: L-shaped method, Stochastic Programming, Carsharing

1. Introduction

High dependency on private vehicles and low occupancy rates increase car usage and congestion in many cities of the world, contributing to pollution and poor urban air quality (Choudhury et al., 2018). Improvements of public transport (Pруд’homme et al., 2012) and road pricing measures (Carpintero and nez, 2011; Dudley, 2013) have, to a large extent, failed to provide sustainable solutions (Choudhury et al., 2018; Barnes et al., 2012; May and Nash, 1996). In this context, shared mobility, and particularly carsharing, has emerged as a viable alternative, linked to, e.g., a decrease in congestion (Crane et al., 2012), pollution (Martin and Shaheen, 2016), land used (Shaheen and Cohen, 2013) and transport costs (Duncan, 2011; Mourad et al., 2019).

In reaction to the high flexibility demanded by users, modern carsharing services are commonly designed for on-demand, short-term, one-way usage (Illgen and Höck, 2019). That is, users are allowed to rent a car without reservation and return it as soon as, and wherever (within the operating area), their journey is completed. The drop-off location/station is thus typically different from the pick-up location/station. Such configuration poses new planning challenges to carsharing operators (CSOs). On-demand rentals make the CSO unaware of when, where and for how long new rentals will occur. One-way rentals create frequent imbalances in the distribution of vehicles, that is an accumulation of vehicles in low-demand zones, and vehicle shortage in high-demand zones (Barth et al., 2004; Boyaci et al., 2015) with levels of service dropping accordingly. A central task for a CSO is to provide a distribution of vehicles in the business area compatible with demand tides and oscillations (Waserhole et al., 2013; Weikl and Bogenberger, 2015).

As a prime form of response to these challenges, CSOs initiate staff-based vehicle relocations between stations/zones of the city before shortages occur and customer satisfaction levels drop (Jorge and Correia, 2013; Illgen and Höck, 2019). That is, CSO’s staff reach designated cars and drives them to different places. This gives rise to the so called Vehicle Relocation Problem (VReP), which consists of determining the relocations to perform in order to prepare for future demand. The research literature covers several problem settings, different levels of detail and granularity of decisions, as well as different mathematical approaches, see e.g., Boyaci et al. (2017), Weikl and Bogenberger (2015), Boyaci et al. (2015), Bruglieri et al. (2014), Jorge et al. (2014), Kek et al. (2009, 2006), Mukai and Watanabe (2005), Barth et al. (2004), Barth and Todd (1999), Folkestad et al. (2020), Hellem et al. (2021). These studies are thoroughly reviewed and classified in a recent survey, see Illgen and Höck (2019).

The survey also reports that staff-based rebalancing could be complemented by manipulating demand through dynamic pricing. In fact, users of urban mobility services typically choose among different transport modes (e.g., metro, carsharing, bikesharing) that vary in a number...
of key attributes including price, see e.g., Zoepf and Keith (2016), Hansen and Pantuso (2018), Pantuso (2020). However, the authors of the survey comment that, at present times, this issue is solely “an important part of future research” (Ilgen and Hŏck, 2019).

Compared with relocation decisions, carsharing pricing strategies have received limited attention in the research literature though the number of available studies is growing. We can distinguish two main categories of pricing strategies, which we refer to as individual and collective, according to their end recipient. Individual pricing strategies are targeted to individual users. They require an interaction between the CSO and the individual user by means of which the final trip price, pick-up or drop-off location are agreed upon. As an example, the operator sends the individual user offers in the form of discounts or bonuses in exchange to a trip configuration which the operator deems beneficial for the entire system. Collective pricing strategies are, instead, targeted to the entire user base. They have the scope of influencing the cumulative rental demand by, e.g., decreasing the price of rentals to/from selected zones, but do not require an interaction with the individual user, nor their reply as to whether the price is accepted or not. The approach proposed in this article belongs to the latter category.

Several pricing strategies can be classified as individual. In Wagner et al. (2015) a method is developed that identifies vehicles placed in low demand zones using idle time as a proxy. The method then offers the user a drop-off location with low expected idle time in exchange for a discount (e.g., free minutes of usage). The method is evaluated in a simulation framework based on the city of Vancouver. The authors report that the average vehicle idle time is decreased by up to 16 percent. In Waserhole and Jost (2016) it is assumed that each customer is interested in a trip between a specific origin and destination and is sensitive to price. The operator then offers a price for the given trip in order to, ideally, incentivize/avoid unfavorable car movements. They model the carsharing system as a continuous-time Markov chain where a pricing policy is input to the model. In Di Febbraio et al. (2018) a user-based relocation method is presented in which the users are offered to leave the car in a location different from the one planned in exchange for a fare discount. The authors formulate the decision problem as mixed-integer nonlinear programming problem, and model customers preferences with respect to the offer of alternative drop-off locations expressing the corresponding utility as a functions of the distance between the desired and offered drop-off locations. In Stokkink and Geroliminis (2021) a predictive, user-based, relocation strategy is introduced for station-based carsharing services. They assume that the CSO can offer each returning customers an incentive to relocate. That is, upon the arrival of a customer, the CSO determines whether to offer an incentive, what the incentive should be and where the vehicle should be relocated. Therefore, an optimization problem is solved upon the arrival of every customer. Estimated customer preferences are used to model their reaction to incentive offers. Station-based services are considered also in Liu et al. (2021) where users send real-time trip requests to the CSO, specifying their origin and destination. Upon receiving requests, the CSO assigns vehicles to users, plans staff-based relocations and determines incentives for customers, to whom a car has already been assigned, in exchange for a change of destination. The focus on (Wu et al., 2021) is instead on free-floating services. The system they consider is organized as follows. Each user sends a request for a vehicle either on-demand or as a reservation for a future rental. With the request they are required to specify intended pick-up and drop-off locations and departure time. The system elaborates the available responses and requests to users with proposed service options. Each service option includes a pick-up and drop-off location, pick-up time, and price. Finally, in Wang et al. (2021) a pricing scheme to induce user-based relocations is introduced, with focus on station-based one-way services. The pricing scheme consists of adding or deducting a fixed expense to the original expense to adjust the users’ preferential pick-up and drop-off location. To study the user behavior a questionnaire is carried out. The pricing problem is finally formulated as an optimization problem.

A number of articles have also focused on collective pricing strategies. A mixed-integer nonlinear programming model is provided in Xu et al. (2018) for the joint problem of deciding fleet size, trip pricing and staff-based relocations for an electric station-based carsharing service. The authors consider demand elasticity with respect to prices using a logit-based function. The authors test the method on a case study based in Singapore.

In Wang and Ma (2019) a pricing scheme is developed with the scope of influencing user demand and keeping the distribution of vehicle at a given balance level. After modeling the relationship between carsharing price and demand, the authors develop a nonlinear optimization model to define pricing schemes which minimize the deviation from inventory upper and lower bounds at each charging station.

A station-based electric one-way service is considered in Xie et al. (2019). The CSO is to decide charging schedules and service prices. The authors assume that rental demand is influenced by prices and adopt a linear elasticity function to express demand as a function of price. Electric carsharing services are considered also in Chen et al. (2019) where a dynamic pricing scheme is proposed with the scope of solving imbalances in the distribution of vehicles, as well as facilitating vehicle-grid-integration. For each origin and destination station the operator can influence demand using two price adjustment levels. Rental demand is connected to prices via a price elasticity to a reference demand for a default price. Pricing decisions are made using a mixed-integer nonlinear program. Finally, in Lu et al. (2021) a bilevel nonlinear mathematical programming model is proposed to determine carsharing prices and staff-based relocations. In the upper level, the carsharing operator determines vehicles relocations and prices. In the lower level, travelers choose travel modes from a cost-minimization perspective and demand is computed using a logit model. The authors assume a one-way station-based carsharing system in competition with private cars.

The method presented in the present paper can be classified as a collective pricing strategy and extends the available literature in a number of ways. First, the available methods do not take into account the impact of alternative transport modes on transport choices. Carsharing services live in multi-modal transport systems and failure to model this heterogeneity may result into myopic models of customers behavior. As an example, compared to Lu et al. (2021), in the present paper customers choices are not limited to private or shared cars. Rather, we assume that customers may choose among any number of available transport services (e.g., bicycle and bus). This, in turn, can help the CSO set prices from/to a given zone also as a function of the alternatives available in the zone. Second, the articles that include demand elasticity, typically limit their attention to elasticity with respect to prices, see e.g., Xu et al. (2018), Lu et al. (2021) and Stokkink and Geroliminis (2021). In this article we allow the operator to model customers preferences with respect to any number of both exogenous and endogenous characteristics of the service such as, but not limited to, travel time and waiting time. Such elasticity is modeled using utility functions which yield a linear optimization problem as long as the function is linear in the endogenous characteristics of the service e.g., price. Finally, extending all available methods, we explicitly account for uncertainty with respect to customer preferences. That is, we consider that a portion of the preferences of each customer is unknown to the operator and is, as such, handled by means of a stochastic program.

The contributions of this article can be stated as follows.

1. We propose a two-stage integer stochastic programming model for the joint pricing and relocation problem. The central idea is to influence demand by acting on prices, and performing preventive relocations accordingly, in order to maximize expected profits. To model the interplay between pricing decisions and
customers choices we follow the recipe first provided by Bierlaire and Azadeh (2016) for integrating demand models within mixed integer linear optimization models. This framework consists of modeling user preferences, and the uncertainty therein, by means of utility functions. A discretization of the unknown portion of the utility, and the adoption of utility functions which are linear in the decision variables of the model, ensure that the resulting optimization model is linear. This framework has also been used by e.g., Paneque et al. (2017) in the context of parking services and Hansen and Pantuso (2018) in the context of carsharing, and a more general description is provided in Paneque et al. (2021).

2. Problem definition and assumptions

We define the problem and clarify modeling assumptions. We introduce the model, particularly the second-stage, in an extensive and discursive manner in order to make the interplay between pricing decisions and customers demand explicit. In Section 4 we describe an integer L-Shaped method to find exact solutions to the problem. This is enabled by a compact reformulation and efficient exact algorithm for the integer subproblems.

3. We provide empirical evidences on the performance of the algorithm and on the solutions obtainable, based on artificial instances built on data from the city of Milan. An instance generator is made available online.

The remainder of this article is organized as follows. In Section 2 we define the problem and clarify modeling assumptions. In Section 3 we provide an extensive formulation of the problem which has the scope of explicitly defining the relationship between pricing decisions and customer choices. We introduce the model, particularly the second-stage, in an extensive and discursive manner in order to make the interplay between pricing decisions and customers demand explicit. In Section 4 we describe an integer L-Shaped method to find exact solutions to the problem. This is enabled by a compact and more tractable reformulation of the second-stage problem where customers choices are pre-processed, and by an exact greedy algorithm for solving the second-stage problem, both described in Section 4. In Section 5 we present a set of artificial instances based on the carsharing services offered in the Italian city of Milan. The same instances as well as an instance generator are made available online. In Section 6 we present the results of a computational study. We shed lights on the efficiency of the algorithm and comment on the solutions obtainable by means of the model introduced. In Section 7 we draw final conclusions, point out existing limitations of this work and discuss possible avenues of future research.

2. Problem definition and assumptions

A CSO offers one-way, reservation-free, carsharing services and is faced with the problem of jointly deciding the prices to charge and relocations to perform in order to comply with demand. The characteristics of the service, and the perimeter of the corresponding decision problem, are clarified by following assumptions.

A0-Target periods The operating hours are partitioned into a number of distinct target periods, that is, portions of the operating hours in which the CSO may, in general, apply different prices and distributions of the fleet, see Fig. 1. Before each target period, and the CSO must decide (i) the prices to apply during the target period and (ii) the relocations to perform in sight of the uncertain rental demand during the target period. The CSO plans for each target period independently based on updated system information (e.g., fleet distribution and individual vehicles’ status) and demand outlook within the target period. In the example in Fig. 1, before 12:00 the CSO must decide the prices to apply in target period 1 (12.00–13.00) and the relocations to perform before that. Thus, in this case the target period lasts for one hour. Similarly, one may consider longer target periods, e.g., morning hours and afternoon hours, as well as shorter ones, e.g., 30 minutes target periods, depending on how often it is sensible to adjust prices in the specific context.

A1-Business area The operating area is made of a finite set of locations, henceforth zones, see Fig. 2. If the carsharing service is station-based a zone naturally represents a station. If the service is free-floating we assume that the business area is suitably partitioned into a number of zones, and each zone is represented by a suitable geographical location.

A2-Pricing scheme The price is made of a per-minute fee and a drop-off fee. The per-minute fee is valid throughout the day (i.e., in all target periods) and is independent of the origin and destination of the trip. Instead, the drop-off fee can be different in each target period and for each origin and destination. Fig. 2 provides an example where the per-minute fee is Euro 0.2, independently of the origin and destination, while for each pair of zones a different drop-off fee is set. In the example, a drop-off fee of Euro 1.5 is charged if the car is picked up in zone $z_1$ and returned in zone $z_1$, while a drop-off fee of Euro $1$ is charged if the car is picked up in zone $z_1$ and returned in zone $z_2$. Thus, we assume that the drop-off fee may also be negative to encourage desired movements of cars and increase demand. This setup generalizes the pricing schemes adopted in a number of carsharing services which typically charge a positive drop-off fee only if the customer returns the car in specific, unfavorable, zones of the city, or provide incentives, such as free driving minutes, to pick up a car in specific, unfavorable, zones. In order to keep the pricing scheme easy to communicate to customers, we assume the CSO must choose among only a finite set of possible drop-off fees. In the example of Fig. 2 this set is Euro $\{-1,0,1,1.5,2\}$.

A3-Alternative transport services The business area offers a number of alternative transport services (e.g., public transport and bicycles) outside the control of the CSO. The alternative services may be different for each pair of zones. Each alternative service has unlimited capacity (i.e., each customer can choose any alternative service without decreasing their availability). In the example in Fig. 2, carsharing is offered on all origin–destination pairs, while busses are not an available alternative for moving
Customers are informed The CSO is able to inform customers about the current price from their location to every other zone, prior to rentals. In the example of Fig. 2, the CSO is able to inform a user in zone \( z_1 \) (e.g., on the mobile application used to locate the car) that, if the car is returned in zone \( z_2 \), there will be a drop-off fee of Euro 1.5, in addition to the per-minute fee. Customers are also aware about the availability of alternative transport services. Considering the example in Fig. 2, a customer moving between zone \( z_1 \) and \( z_2 \) knows that they may use bus, bicycle, and carsharing. For all possible transport modes (including carsharing) the user knows the respective prices and characteristics (e.g., waiting time and travel time).

A5-Closed market A customer chooses exactly one transport service among the available ones. This corresponds to saying that a customer does not get to decide their trip. In the example of Fig. 2, a customer moving from zone \( z_1 \) to zone \( z_2 \) will eventually choose to travel either by bicycle, carsharing or bus, and complete its journey.

A6-Customers preferences The CSO is able to describe a portion of customers travel preferences as a function of different observable characteristics of the available transport services (e.g., travel time, price and waiting time). Nevertheless, the choice of each customer depends also on a number of additional elements not observable by the CSO. Therefore, customers preferences are partially unknown to the CSO. The unknown part of customers preferences is fully described by a probability distribution.

A7-Direct rentals Customers traveling with shared cars drive directly from their origin to their destination zone. This assumption is made for simplicity and is without loss of generality. Different travel patterns can be included simply by modeling customer-specific travel times in (4).

A8-Homogeneous fleet All shared vehicles are identical. This assumption is made for the sake of simplicity in the exposition of the reformulation of the second-stage problem and is without loss of generality. Throughout the text we will comment on the necessary modifications in case of a heterogeneous fleet.

A9-Profit maximization The CSO maximizes profits. While other objectives may be considered, such as maximizing demand served, or minimizing zonal deficit of cars, profits are the central objective of private carsharing operators.

A10-One-way trips For the sake of simplicity, we assume one-way trips. That is, customers move from their origin zone to a different zone. The model presented in Section 3 can however accommodate also round trips, provided a suitable specification of the parameters of the trip (e.g., duration).

Based on these assumptions, the problem can be briefly stated as follows. Given (a) a target period, (b) the cumulative mobility demand between each pair of zones in the target period, (c) usage and relocation costs, (d) the current distribution of cars, (e) a model of customers preferences including a probability distribution describing customer preferences unknown to the CSO, the CSO is to decide (i) the drop-off fees to apply during the target period and (ii) the relocations to perform in sight of the uncertain rental demand during the target period in order to maximize expected profits.

3. Mathematical model

Consider a urban area represented by a finite set \( I \) of zones (e.g., charging stations or a suitable partition of the business area) and a CSO offering a finite set of shared vehicles \( \mathcal{V} \). Before the beginning of the target period, the CSO is to decide the drop-off fee between each pair of zones and the relocations to perform to better serve demand in the target period. At the time of planning, the fleet is geographically dispersed in the urban area as the result of previous rentals. Let decision variable \( z_{vi} \) be equal to 1 if vehicle \( v \) is made available for rental in zone \( i \) in the target period, 0 otherwise. Let \( c_{vi}^R \) be the relocation cost born by the CS company to make vehicle \( v \) available in zone \( i \). This cost is zero if the vehicle is initially in zone \( i \) and positive otherwise. Let \( L \) be a finite set of drop-off fees the CSO may apply. Let decision variable \( \lambda_{ij} \) be equal to 1 if fee \( l \) is applied between zone \( i \) and zone \( j \), 0 otherwise. Finally, let \( z := (z_{vi})_{i \in I, v \in \mathcal{V}} \) and \( \lambda := (\lambda_{ij})_{i \in I, j \in I \setminus \{i\}} \). The carsharing pricing and relocation problem is thus

\[
\begin{align}
\max & - \sum_{i \in I} \sum_{v \in \mathcal{V}} c_{vi}^R z_{vi} + Q(z, \lambda) \quad (1a) \\
\sum_{i \in I} z_{vi} &= 1 \quad \forall v \in \mathcal{V} \quad (1b)
\end{align}
\]
\[ \sum_{i \in \mathcal{L}} \lambda_{ij} = 1 \quad i, j \in I \]  
\[ z_{ij} \in [0, 1] \quad i \in I, v \in \mathcal{V} \]  
\[ \lambda_{ij} \in [0, 1] \quad i, j \in I, l \in \mathcal{L}. \]

Constraints (1b) ensure that each vehicle is made available in exactly one zone. Constraints (1c) state that exactly one drop-off fee can be selected between each origin \( i \) and destination \( j \). The objective function (1a) represents the expected profit obtained in the target period. The first term consists of the total relocation cost while the second term \( Q(z, \lambda) \) represents the expected revenue from rentals as a result of the definition of the pricing and relocation activities. The meaning of \( Q \) is to make the per-minute fee and the driving time between zones \( i \) and \( j \) explicit by the end of this section as a result of the definition of the two-stage problem which we are now introducing.

Once relocation \( z \) and pricing \( \lambda \) decisions have been made, the CSO observes the consequent customers rentals. The business area offers a set \( \mathcal{A} \) of alternative transport services, outside the control of the CSO, such as metro, busses and private bicycles. Each service has, in general, a different price and different characteristics. Let decision variable \( p_{ij} \) be the price of service \( v \in \mathcal{V} \cup \mathcal{A} \) between zones \( i \) and \( j \). The price of a carsharing ride between zones \( i \) and \( j \) is

\[ p_{ij} = P_{ij} + \sum_{l \in \mathcal{L}} \lambda_{ijl} \quad \forall v \in \mathcal{V}, i, j \in I \]

where parameter \( P_{ij} \) is the carsharing per-minute fee, \( T_{ij}^{cs} \) the driving time between zones \( i \) and \( j \) and \( L_l \) the value of drop-off fee at level \( l \in \mathcal{L} \) in some currency. Note that, in case of a heterogeneous fleet, it is simply necessary to make the per-minute fee and the driving time vehicle-dependent. Instead, the price of alternative services is entirely exogenous, that is

\[ p_{ij} = P_{ij} \quad \forall v \in \mathcal{V}, i, j \in I \]

where parameter \( P_{ij} \) is the price of alternative service \( v \in \mathcal{A} \) between \( i \) and \( j \) or in \( I \).

Let \( \mathcal{K} \) be the set of customers, with \( \mathcal{K}_i \subseteq \mathcal{K} \) being the set of customers traveling from zone \( i \in I \) and \( \mathcal{K}_{ij} \subseteq \mathcal{K}_i \) the set of customers traveling from \( i \in I \) to \( j \in I \) in the target period.

Consider an individual customer \( k \). The customer is faced with a choice among a finite number of alternative transport services that can bring them to their destination. Using the fairly standard assumption that customers maximize their utility, we can state that each service will provide the customer a different utility, and that the customer will choose the transport service that provides them the highest utility. This utility is known to the customer but not to the CSO.

Consider now the CSO. As we said, the CSO is not aware of the utility provided by the different services to each customer. Rather, the CSO is aware of a number of characteristics of the different services, primarily the price, \( p_{ij} \) and a some additional characteristics, say \( x_{ij}^1, \ldots, x_{ij}^N \) for service \( v \) between \( i \) and \( j \) (e.g., travel time and waiting time), as well as possibly some characteristics of the decision maker. Based on this, the CSO can specify a function that relates these, known, characteristics to the utility obtained by the customer. We denote this function as

\[ F_k(p_{ij}, x_{ij}^1, \ldots, x_{ij}^N) \]

However, there are additional elements that influence the utility that the CSO does not or cannot observe. For this reason, the utility is better represented by

\[ F_k(p_{ij}, x_{ij}^1, \ldots, x_{ij}^N) + \xi_{kv} \]

where \( \xi_{kv} \) is a random variable that captures the difference between the utility that the CSO is able to model and the true utility observed by the customer. Different distributions for \( \xi_{kv} \) will lead to different choice models. As an example, the popular Logit model is obtained when each \( \xi_{kv} \) follows, independently, an identical extreme value distribution (Gumbel type I), and the Probit model is obtained when it follows a multivariate normal distribution. These, and additional choice models, as well as a discussion of their fundamental assumptions and limitations are discussed in, e.g., Train (2009). See also (Ben-Akiva and Bierlaire, 1999) for an exposition related to transport choices.

Let now \( u_{ijkv} \) be a decision variable which captures the utility obtained by customer \( k \in \mathcal{K} \) when moving from \( i \) to \( j \) in \( I \) using service \( v \in \mathcal{V} \cup \mathcal{A} \). Given a realization \( \xi_{kv} \) of the random term \( \xi_{kv} \), the utility is determined by

\[ u_{ijkv} = F_k(p_{ij}, x_{ij}^1, \ldots, x_{ij}^N) + \xi_{kv} \quad \forall i, j \in I, k \in \mathcal{K}_{ij}, v \in \mathcal{V} \cup \mathcal{A} \]

Note that, since the observed characteristics of the different transport services, \( x_{ij}^1, \ldots, x_{ij}^N \), are given, constraints (4) are linear if \( F_k(\cdot) \) is linear in \( p_{ij}, \lambda_{ijl} \), which is instead a decision variable.

Based on the utility provided by the different transport services, customers will make their choices. Let decision variable \( w_{ijkv} \) be equal to 1 if customer \( k \in \mathcal{K}_{ij} \) chooses service \( v \in \mathcal{V} \cup \mathcal{A} \), 0 otherwise. A customer will choose exactly one service (see Assumption A5 in Section 2)

\[ \sum_{v \in \mathcal{V} \cup \mathcal{A}} w_{ijkv} = 1 \quad \forall i, j \in I, k \in \mathcal{K}_{ij} \]

In order for a customer to choose a service, the service must be available. Let thus binary variable \( y_{ikv} \) be equal to 1 if service \( v \in \mathcal{V} \cup \mathcal{A} \) is offered to customer \( k \in \mathcal{K}_i \), 0 otherwise. Alternative services \( v \in \mathcal{A} \) are always offered to customers whenever they are available at all, that is

\[ y_{ikv} = Y_{iv} \quad \forall i \in I, k \in \mathcal{K}_i, v \in \mathcal{A} \]

where parameter \( Y_{iv} \) is equal to 1 if alternative service \( v \) is available in zone \( i \), 0 otherwise. Conversely, a shared car \( v \in \mathcal{V} \) may be offered to customers in zone \( i \) only if it is physically available at \( i \), that is

\[ y_{ikv} \leq z_{iv} \quad \forall i \in I, k \in \mathcal{K}_i, v \in \mathcal{V} \]

In addition, each car \( v \in \mathcal{V} \) can be rented by only one customer. If more than one customers wish to use car \( v \), the car is taken by the first customer arriving at the car. We assume that customers are indexed according to their arrival time at the car, i.e., customer \( k \) arrives before \( q \) if \( k < q \). We impose that a vehicle is offered to a customer only if it is offered also to the customer arriving before them (who perhaps did not take it), that is:

\[ y_{ikv} \leq y_{ikv-1v} \quad \forall i \in I, k \in \mathcal{K}_i, v \in \mathcal{V} \]

A vehicle becomes unavailable for a customer if any customer has arrived before them and rented the car, that is:

\[ z_{iv} - y_{ikv} = \sum_{q \leq k} \sum_{u \in \mathcal{V}} w_{ijkv} \quad \forall i \in I, k \in \mathcal{K}_i, v \in \mathcal{V} \]

that is, if car \( v \) is in zone \( i \) (\( z_{iv} = 1 \)), but it is not offered to customer \( k \) (\( y_{ikv} = 0 \)) we obtain

\[ 1 = \sum_{q \leq k} \sum_{u \in \mathcal{V}} w_{ijkv} \]

meaning that one customer has arrived before \( k \) and rented the car. On the other hand, if the car is offered to customer \( k \) (\( y_{ikv} = 1 \)), then it must be in zone \( i \) (\( z_{iv} = 1 \)—see (7)), and we obtain

\[ 0 = \sum_{q \leq k} \sum_{u \in \mathcal{V}} w_{ijkv} \]

meaning that no customer arriving before \( k \) has taken the car. The same equality holds if the vehicle is not available at all (\( z_{iv} = 0 \) and \( y_{ikv} = 0 \)). Now that we have clarified how the availability of rental cars is regulated, we can state that a service can be chosen only if it is offered to the customer

\[ w_{ijkv} \leq y_{ikv} \quad \forall i, j \in I, k \in \mathcal{K}_{ij}, v \in \mathcal{V} \cup \mathcal{A} \]

Among the available services, the customer will chose the one yielding the highest utility. Therefore, for a given zone \( i \in I \), let decision
variable \( v_{ijkw} \) be equal to 1 if both services \( v \) and \( w \) in \( V \cup A \) are available to customer \( k \in K \), 0 otherwise, and decision variable \( \mu_{ijkw} \) be equal to one if service \( v \in V \cup A \) yields a greater utility than service \( w \in V \cup A \) to customer \( k \in K \), moving from \( i \) to \( j \), 0 otherwise. The following constraints state that \( v_{ijkw} \) is equal to one when both services \( v \) and \( w \) are available

\[
y_{ijkw} + y_{ijkw} \leq 1 + v_{ijkw} \quad \forall i, j, k \in K, v, w \in V \cup A, (11)
\]

\[
v_{ijkw} \leq y_{ijkw} \quad \forall i, j, k \in K, v, w \in V \cup A, (12)
\]

\[
v_{ijkw} \leq y_{ijkw} \quad \forall i, j, k \in K, v, w \in V \cup A, (13)
\]

A service is chosen only if it yields the highest utility

\[
w_{ijkw} \leq \mu_{ijkw} \quad \forall i, j, k \in K, v, w \in V \cup A (14)
\]

that is, as soon as \( \mu_{ijkw} \) is set to 0 for some index \( v, w, w_{ijkw} \) is forced to take value 0 and service \( v \) is not chosen by customer \( k \) on \( i-j \). The following constraints ensure that decision variable \( \mu_{ijkw} \) takes the correct value according to the utility

\[
M_{ijk} v_{ijkw} - 2M_{ijk} \leq u_{ijkw} - u_{ijkw} - M_{ijk} \mu_{ijkw}
\]

\[
\forall i, j, k \in K, v, w \in V \cup A, (15)
\]

and

\[
u_{ijkw} - u_{ijkw} - M_{ijk} \mu_{ijkw} \leq (1 - v_{ijkw}) M_{ijk}
\]

\[
\forall i, j, k \in K, v, w \in V \cup A, (16)
\]

where constant \( M_{ijk} \) represents the greatest difference in utility between two services on \( i-j \) for customer \( k \in K \), that is \( M_{ijk} \geq |u_{ijkw} - u_{ijkw}| \), \( v, w \in V \cup A \). Constraints (15)–(16) work as follows. When both services \( v \) and \( w \) are available (\( v_{ijkw} = 1 \) and \( w_{ijkw} = 1 \)), \( \mu_{ijkw} \) forces \( \mu_{ijkw} \) to take value 1, while (15) reduces to \( 0 \leq u_{ijkw} - u_{ijkw} \). When both service \( v \) and \( w \) are available and \( u_{ijkw} < w_{ijkw} \), \( \mu_{ijkw} \) forces \( \mu_{ijkw} \) to take value 0, while (16) reduces to \( 0 \geq u_{ijkw} - u_{ijkw} \). When one of the services is not available (\( v_{ijkw} = 0 \)), constraints (15)–(16) are satisfied irrespective of the value of \( \mu_{ijkw} \). In case of ties (\( u_{ijkw} = w_{ijkw} \)) we impose

\[
\mu_{ijkw} + \mu_{ijkw} \leq 1 \quad \forall i, j, k \in K, v, w \in V \cup A (17)
\]

A service can be preferred only if offered

\[
u_{ijkw} \leq u_{ijkw} \quad \forall i, j, k \in K, v, w \in V \cup A (18)
\]

Let decision variable \( a_{ijk} \) be equal to 1 if fare \( i \) is applied between \( i \) and \( j \) and customer \( k \) chooses shared car \( v \in V \), 0 otherwise. The following constraints ensure the relationship between \( \lambda_{ij} \) and \( w_{ijk} \) and \( a_{ijk} \)

\[
\lambda_{ij} + w_{ijkw} \leq 1 + a_{ijkw} \quad \forall v \in V, i, j, k \in K, l \in L (19)
\]

\[
a_{ijkw} \leq \lambda_{ijl} \quad \forall v, i, j, k \in K, l \in L (20)
\]

\[
\lambda_{ijk} \leq a_{ijk} w_{ijkw} \quad \forall v, i, j, k \in K, l \in L (21)
\]

That is, \( a_{ijk} \) is forced to take value 1 as soon as both \( \lambda_{ij} \) and \( w_{ijkw} \) take value one, and value 0 as soon as either \( \lambda_{ij} \) or \( w_{ijkw} \) take value 0.

Finally, for a given realization \( \xi \) of \( \xi \), and first-stage decision \( (z, \lambda) \), the second-stage problem can be reformulated as

\[
Q(z, \lambda, \xi) := \max \sum_{i \in V} \sum_{j \in E} \sum_{l \in L} \left( p_{ij}^{\xi} y_{ijkw} - c_{ij}^{\xi} \right) \sum_{k \in K} w_{ijkw}
\]

\[
+ \sum_{i \in V} \sum_{j \in E} \sum_{l \in L} \sum_{k \in K} a_{ijk} \lambda_{ijl}
\]

\[
s.t. (2),(3),(4),(5),(6),(7),(8),(9), (10),(11),(12),(13),(14),(15), (16),(17),(18),(19),(20),(21)
\]

where \( C_{ij}^{\xi} \) is the cost borne by the CSO when a vehicle is rented between \( i \) and \( j \), (22a) represents the net revenue generated by the per-minute fee, and (22b) represents the income generated by the drop-off fee.

Thus, we can formally express the expected profit (i.e., the recourse function) as

\[
Q(z, \lambda) := \mathbb{E}_{\xi} \left[ Q(z, \lambda, \xi) \right]
\]

Problem (1) is a two-stage mixed-integer stochastic program with integer decision variables at both stages.

4. L-shaped method

We propose a multi-cut Integer L-Shaped method to find exact solutions to problem (1). The original, single-cut, version of the method was introduced by Laporte and Louveaux (1993). Assuming a set \( S = \{1, \ldots, s\} \) of scenarios (e.g., an iid sample) of \( \xi \), each with probability \( \pi_{s} \), the Master Problem (MP) can be formulated as

\[
\max \sum_{i \in V} \sum_{l \in L} c_{ij}^{\xi} z_{ijl} + \sum_{t \in E} \pi_{t} \phi_{t}
\]

\[
\sum_{i \in V} \sum_{l \in L} z_{ijl} = 1 \quad \forall v, \in V (23a)
\]

\[
\sum_{i \in V} \sum_{j \in E} \sum_{l \in L} \sum_{k \in K} a_{ijk} \lambda_{ijl} = 1 \quad \forall i, j \in J (23b)
\]

\[
z_{ijl} \in \{0,1\} \quad \forall i, j \in I, l \in L (23c)
\]

\[
\phi_{t} \quad \text{free} \quad s \in S (23f)
\]

Let \( \psi := (\phi_{t})_{s \in S} \). For each \( s \in S \), the second-stage problem \( Q(z, \lambda, \xi_{s}) \) is solved as a subproblem. The L-Shaped method consists of solving MP in a Branch & Cut framework where optimality cuts are added at (integer) nodes of the tree. Observe that problem (1) has relatively complete recourse, that is, the second-stage problem \( Q(z, \lambda, \xi_{s}) \) is feasible for every solution that satisfies the first-stage constraints. Consequently, the method requires only the definition of optimality cuts.

The practical viability of the method is enabled by a compact reformulation of \( Q(z, \lambda, \xi_{s}) \), provided in Section 4.1, and an efficient exact algorithm for its solution, introduced in Section 4.2. We then provide the expression of optimality cuts and relaxation cuts in Sections 4.3 and 4.4, respectively. The former are necessary to cut off solutions for which \( \phi_{t} < Q(z, \lambda, \xi_{s}) \), the latter provide non-trivial lower bounds to \( Q(z, \lambda, \xi_{s}) \) and are crucial for the efficiency of the algorithm. Final efficiency measures are provided in Section 4.5.

4.1. Compact formulation of the second-stage problem

For a given realization \( \xi \) of \( \xi \), and first-stage decision \( (z, \lambda) \), the second-stage problem can be reformulated by preprocessing customer choices. The preprocessing phase ensures that the reformulation is linear regardless of whether the utility function adopted is linear in the price. Thus, any choice model can be used, without any restriction to utility function linear in the price.

We introduce the concept of a request. A request represents a customer who wishes to use carsharing for moving from its origin to its destination. For a given realization \( \xi \), let the set \( R(\xi) \) be the set of requests. The set \( R(\xi) \) contains a request for each customer \( k \in K \) for which there exists at least one drop-off level \( l \in L \) such that the customer would prefer carsharing to alternative transport services, that is, for which \( u_{ijkw} > u_{ijkw} \) with \( v \in V \) and \( w \in A \) for some choice of \( l \in L \) (note that all shared cars yield the same utility). Let \( i(r), j(r) \) and \( k(r) \) be the origin, destination and customer of request \( r \), respectively, and \( l(r) \) the highest drop-off fee at which customer \( k(r) \) would prefer carsharing to other services. Note that customer \( k(r) \) will still prefer carsharing at any drop-off fee lower than \( l(r) \) under the reasonable assumption that the customer is sensitive to price. For each realization \( \xi \) of \( \xi \) the set of requests can be populated in \( O(|K| \times |C| \times |A|) \) operations as described in Algorithm 1.
Algorithm 1 Computation of $R(\xi)$.  
1: Input: $\xi$  
2: $R(\xi) \leftarrow \emptyset$  
3: for customer $k \in \mathcal{K}$ do  
4: $i \leftarrow (k), j \leftarrow j(k)$ \{i(k) and j(k) are the origin and destination of customer $k$\}  
5: $L_{i}^{M A X} \leftarrow -\infty$ \{The highest drop-off fee at which customer $k$ will choose carsharing\}  
6: $L_{j}^{M A X} \leftarrow -\infty$  
7: for drop-off level $l \in L$ do  
8: $g^{C S} \leftarrow P^{C S}T_{l}^{C S} + L_{i}$ \{Calculate the price of a carsharing ride\}  
9: $U_{c}^{C S} \leftarrow F_{1}(P_{c}^{C S}, x_{l_{i}}^{C S}, ..., x_{l_{j}}^{C S}) + \zeta_{k_{l}}$ for some $v \in V$ \{Calculate the utility of carsharing\}  
10: for service $v \in A : Y_{v} = 1$ do  
11: \[ p_{v}^{A} \leftarrow P_{v} \] \{Calculate the price of a ride with alternative $v$\}  
12: $U_{v}^{A} \leftarrow F_{2}(P_{v}, x_{l_{i}}^{A}, ..., x_{l_{j}}^{A}) + \zeta_{l_{v}}$ \{Calculate the utility of alternative $v$\}  
13: end for  
14: if $U_{c}^{C S} > \max_{v \in A}(U_{v}^{A})$ and $L_{i} > L_{i}^{M A X}$ then  
15: $L_{i}^{M A X} \leftarrow L_{i}$  
16: end if  
17: end for  
18: end for  
19: if $L_{i}^{M A X} > -\infty$ then  
20: $r \leftarrow [R(\xi) \cup \{r\}]$ \{In this case there exists a drop-off fee at which $k$ prefers carsharing. Thus we create a request\}  
21: $R(\xi) \leftarrow R(\xi) \cup \{r\}$  
22: $i(r) \leftarrow i$  
23: $j(r) \leftarrow j$  
24: $k(r) \leftarrow k$  
25: $l(r) \leftarrow L_{i}^{M A X}$  
26: end if  
27: end for  
28: return $R(\xi)$  

Observe that, in case of a heterogeneous fleet, Algorithm 1 should be edited at lines 8–9 and 14 to account for the fact that each vehicle gives, in general, a different utility.

Let then $R_{i} = P_{i}^{v}T_{l}^{C S} - C_{i,j}^{C S} - L_{i}$, for $l \leq l(\xi)$, be the net revenue generated if request $r$ is satisfied at drop-off fee level $l$. Let $R_{i}(\xi) = \{p \in R(\xi) : i(p) = i(r), k(p) < k(r)\}$ be the set of requests which have a precedence over $r$. Let $R_{i}(\xi) = \{r \in R(\xi) : i(r) = i, j(r) = j\}$. Let $L_{i}(\xi) = \{l \in L : l_{i} \leq l \leq l_{j}\}$. Finally, let decision variable $y_{i,v}$ be equal to 1 if request $r$ is satisfied by vehicle $v$ at level $l$, 0 otherwise. We can now reformulate the second-stage problem as follows.

\begin{equation}
Q(\lambda, \zeta, \xi) = \max \sum_{r \in R(\xi)} \sum_{v \in V} R_{i}y_{i,v} \quad (24a)
\end{equation}

\begin{equation}
\sum_{r \in R(\xi)} y_{i,v} \leq 1 \quad (24b)
\end{equation}

\begin{equation}
\sum_{r \in R(\xi)} y_{i,v} \leq 1 \quad (24c)
\end{equation}

\begin{equation}
y_{i,v}z_{i,j} + \sum_{r \in R(\xi)} \sum_{l \leq l(\xi)} y_{i,v} \leq z_{i,j}(r) \quad (24d)
\end{equation}

\begin{equation}
y_{i,v}z_{i,j} + \sum_{r \in R(\xi)} \sum_{l \leq l(\xi)} y_{i,v} \leq z_{i,j}(r) \quad (24e)
\end{equation}

\begin{equation}
\sum_{r \in R(\xi)} y_{i,v} \leq \lambda_{i,v}(r) \quad (24f)
\end{equation}

\begin{equation}
y_{i,v} \in \{0, 1\} \quad (24g)
\end{equation}

The objective function (24a) represents the net revenue obtained by the satisfaction customer requests. Constraints (24b) ensure that each request is satisfied at most once. Constraints (24c) ensure that each vehicle satisfies at most one request. Constraints (24d) state that a request can be satisfied by vehicle $v$ only if the vehicle is in zone $i(r)$, and the vehicle has not been assigned to a customer with a lower index (that is, arriving at the vehicle before $k(r)$). Constraints (24e) state that a request $r$ is a level $l$ must be satisfied by vehicle $v$ if level $l$ has been chosen $(\lambda_{i,v}(r) = 1)$, and the vehicle is available at $i(r)$ $(z_{i,j}(r) = 1)$, unless the car has been used to satisfy the request of a customer with a higher priority (second term on the left-hand-side), or $r$ has been satisfied by another vehicle (third term on the left-hand-side). Constraints (24f) state that a request can be satisfied at level $l$ only if level $l$ is applied to all customers traveling between $i$ and $j$. Note that $z$ and $\lambda$ are input data in problem (24).

4.2. Solution of the second-stage problem

Given a solution $(z, \lambda)$ to MP and a scenario $\xi$, the optimal second stage profit $Q(z, \lambda, \xi)$ and solution can be computed by the greedy procedure sketched in Algorithm 2.

Algorithm 2 Greedy algorithm for computing $Q(z, \lambda, \xi)$ and its optimal solution.

1: INPUT: $z$, $\lambda$, $R(\xi)$.
2: $Y_{v}^{A} \leftarrow Y(\{v\}$ is the set of available vehicles.).
3: $Y_{i,j} \leftarrow 0 \forall v \in V, r \in R(\xi), l \leq L_{i}(\xi)$.
4: $Q(z, \lambda, \xi) \leftarrow O$
5: Sort requests $R(\xi)$ in non-decreasing order of the customer index $k(r)$ (Remember that customers with a lower index have the precedence over customers with a higher index, see (9)).
6: for request $r \in R(\xi)$ do  
7: $l_{i}(r) \leftarrow \min \{l_{i}, l_{j} \}$ \{Identify the fee applied between $i(r)$ and $j(r)$\}  
8: if $l_{i}(r) \leq l_{i}(r)$ then  
9: for $v \in Y_{v}^{A}$ do  
10: if $z_{i,j}(r) = 1$ then  
11: $Y_{i,v} \leftarrow Y_{i,v} \leftarrow 1$
12: $Y_{v}^{A} \leftarrow Y_{v}^{A} \setminus \{v\}$ \{Vehicle $v$ becomes unavailable\}  
13: $Q(z, \lambda, \xi) \leftarrow Q(z, \lambda, \xi) + R_{i,v}(l_{i}(r))$
14: end if  
15: end for  
16: end if  
17: end for  
18: return $Q(z, \lambda, \xi)$ and $Y_{i,v}$ for all $v \in V, r \in R(\xi), l \leq L_{i}(\xi)$.
Assume an upper bound \( U_i \) on \( \max_{s \in S} Q(z, \lambda, \xi_s) \) exists for all \( s \in S \). In the case of a homogeneous fleet a valid upper bound \( U_i \) on \( \max_{s \in S} Q(z, \lambda, \xi_s) \) is

\[
U_i = \sum_{R \in R_i} \max \{ R_{ijs}(\xi_s), 0 \}
\]

That is, the upper bound assumes that all, and only, the requests which generate a positive revenue are satisfied, and that these are satisfied at the highest drop-off fee \( l(i) \). If the fleet is not homogeneous, we would have a vehicle-specific revenue \( R_{vij} \) for satisfying request \( r \) at level \( i \). In this case, a valid upper bound can be obtained by assuming all requests are satisfied by the vehicle which yields the highest non-negative revenue, i.e.,

\[
U_i = \sum_{R \in R_i} \max \{ \max R_{ijs}(\xi_s), 0 \}
\]

Given a solution \((z', \lambda')\) to MP (e.g., at a given node \( t \) of the Branch & Cut procedure), let \( Z_{it}^s \subseteq Y \times I \) and \( Z_{it}^s \subseteq \mathcal{Y} \times \mathcal{I} \) be the set of tuples \((i, j)\) for which \( z_{it}^s = 1 \) and \( z_{it}^s = 0 \), respectively. Similarly, let \( Z_{i}^s \subseteq \mathcal{I} \times \mathcal{I} \) and \( Z_{i}^s \subseteq \mathcal{I} \times \mathcal{I} \) be the set of tuples \((i, j, l)\) for which \( \lambda_{ijl} = 1 \) and \( \lambda_{ijl} = 0 \), respectively. Proposition 4.1 defines a valid set of optimality cuts.

**Proposition 4.1.** Let \((z', \lambda')\) be the \( t \)th feasible solution to MP, and \( Q(z, \lambda, \xi_s) \) its second-stage value for scenario \( s \). The set of cuts

\[
\phi_j \leq Q(z, \lambda, \xi_s) - U_j
\]

for all \( \phi_j \leq Q(z, \lambda, \xi_s) - U_j \) reduces to

\[
\sum_{(i,j) \in E} z_{ij} - \sum_{(i,j) \in A^*_i} \lambda_{ij} \leq \sum_{(i,j) \in A^*_i} \lambda_{ij} - \sum_{(i,j) \in A^*_i} \lambda_{ij} \]

Proof. It is sufficient to observe that, for \((z, \lambda) = (z', \lambda')\), we have

\[
\sum_{(i,j) \in E} z_{ij} - \sum_{(i,j) \in A^*_i} \lambda_{ij} \leq \sum_{(i,j) \in A^*_i} \lambda_{ij} - \sum_{(i,j) \in A^*_i} \lambda_{ij}
\]

and optimality cut (25) reduces to

\[
\phi_j \leq Q(z, \lambda, \xi_s), \quad \forall s \in S
\]

On the other hand, if \((z, \lambda) \neq (z', \lambda')\) we get

\[
\sum_{(i,j) \in E} z_{ij} - \sum_{(i,j) \in A^*_i} \lambda_{ij} \leq \sum_{(i,j) \in A^*_i} \lambda_{ij} - \sum_{(i,j) \in A^*_i} \lambda_{ij}
\]

and, observing that \( Q(z, \lambda, \xi_s) - U_j \leq 0 \), the right-hand-side of the cut becomes greater than or equal to \( U_j \).

Thus, since MP is a maximization problem, the set of cuts enforces \( \phi_j \leq Q(z, \lambda, \xi_s) \) whenever \((z, \lambda) = (z', \lambda')\) and yields a valid upper bound on the remaining solutions.

### 4.4. Relaxation cuts

Ordinary Benders decomposition cuts can be derived by solving the LP relaxation of the subproblems \( Q(z, \lambda, \xi_s) \). Relaxation cuts are not to be confused with optimality cuts as they are, in general, not tight at the point \((z', \lambda')\) at which they are generated. However, relaxation cuts provide a, possibly, non-trivial upper bound on \( Q(z, \lambda, \xi_s) \), that is an upper bound which might be lower than \( U_i \) for several \((z, \lambda)\) solutions. A relaxation cut is obtained as follows

\[
\phi_j \leq \sum_{r \in R_i} \mathbf{x}^r + \sum_{v \in V} \mathbf{x}^v + \sum_{r \in R_i} \mathbf{x}^r - \sum_{i \in \mathcal{I}} \sum_{v \in \mathcal{V}} \mathbf{x}^v z_{iij}(\xi_s)
\]

where \( \mathbf{x}^r \), \( \mathbf{x}^v \), \( \mathbf{x}^D \), \( \mathbf{x}^K \) are the values of the dual solution to \( Q(z, \lambda, \xi_s) \) corresponding to constraints (24b), (24c), (24d), and (24e), respectively.

### 4.5. Other efficiency measures

MP exhibits symmetric solutions. If no vehicle is made available in a given zone \( i \), all configurations of the drop-off fees between zone \( i \) and the remaining zones \( j \) are equivalent. In fact, no customer will be served on those \((i, j)\) pairs. This problem can be solved by mean of the following constraints

\[
\sum_{v \in V} z_{ij} + \lambda_{ij} \geq 1 \quad \forall i, j \in I
\]

Constraints (27) ensure that, when no vehicle is available in zone \( i \) (i.e., \( \sum_{v \in V} z_{ij} = 0 \)) we arbitrarily chose drop-off fee number 1 (i.e., \( \lambda_{ij} = 1 \)). On the other hand, if \( \sum_{v \in V} z_{ij} \geq 1 \) constraints (27) are satisfied regardless of the choice of a drop-off fee.

### 5. Test instances

In this section we present the test instances we used to run a computational study whose results are presented in Section 6. The test instances mimic carsharing services in the Italian city of Milan. The city hosts a number of carsharing companies and, according to the municipality of Milan (SIAD, 2020), in 2018 there were a total of 3108 free-floating shared vehicles, with an average of 16851 daily rentals, and 149 station-based shared vehicles with an average of 108 daily rentals. We start by describing how the instances were constructed and finally we clarify the specific choices of control parameters for our tests. For the sake of replicability, an instance generator is made publicly available at the address https://github.com/GioPan/instancesPricingAndRepositioningProblem.

#### 5.1. Zones and alternative transport services

We build upon, and expand, the instances used by Hansen and Pantuso (2018). The authors consider ten key locations in the business area of the city of Milan which we use as representatives of as many zones, thus setting \( I = \{1, \ldots, 10\} \), see Fig. 3. The authors consider as alternative transport services public transport (PT-consisting of a combination of buses, metro and superficial trains) and bicycles (B). Therefore we set \( A = \{PT, B\} \). For each pair of zones, the authors provide all the information necessary to calculate the utility as further explained in Section 5.2.

#### 5.2. Customers and utility functions

According to (4) each customer \( k \) is characterized by a known utility function and a random variable \( \xi_k \), which represents the portion of the preferences of the customer with respect to service \( v \) that the CSO cannot explain. We start by introducing the portion of the utility estimated by the CSO.

We adopt the utility function described in Modesti and Sciomachen (1998). This function provides an estimate of the utility of each transport service as a function of price, travel time, walking time (e.g., to reach the service and from the service to destination) and waiting time. These represent the characteristics observable by the CSO.

For each customer \( k \in K \) traveling between \( i \) and \( j \) with transportation service \( v \) the utility function is

\[
U_{ijk} = \sum_{v \in V} \beta^v p_{ijk} + \sum_{v \in V} \beta^v T_{ijk}^v + \sum_{v \in V} \beta^v W_{ijk}^v
\]

where \( \beta^v \) are the coefficients of the functional form. The estimated utility function is

\[
U_{ijk} = \sum_{v \in V} \beta^v p_{ijk} + \sum_{v \in V} \beta^v T_{ijk}^v + \sum_{v \in V} \beta^v W_{ijk}^v
\]
The meaning of each parameter and coefficient of function (28) is clarified in Table 1 and function \( r : \mathbb{R} \to \mathbb{R} \), defined as \( r(t) = \left\lceil \frac{t}{100} \right\rceil \), allows us to model the utility of cycling and walking as a piece-wise linear function: the utility of walking and cycling decreases faster as the walking and cycling time increases, see Modesti and Sciomachen (1998).

We use the \( \beta \) coefficients of the original utility function provided by Modesti and Sciomachen (1998) and marginally adapted to the carsharing context by Hansen and Pantuso (2018) (e.g., price sensitivity has been adapted from Italian Lira to Euro). The values of the coefficients are the following: \( \beta^{CS} = -1, \beta^{PT} = -2, \beta^B = -2.5, \beta^{Walt} = -3 \) and \( \beta^{Walt} = -6 \). For the price sensitivity \( \beta^P \) the authors create two customer segments. They assign \( \beta^P = -188.33 \) if a customer belongs to the lower-middle class or \( \beta^P = -70.63 \) if a customer belongs to the upper-middle class. We randomly assigned customers (with equal probability) to the either upper-middle class \( (\beta^P = -70.63) \) or lower-middle class \( (\beta^P = -188.33) \). In more general cases, the parameters of utility functions can be estimated, provided the availability of data records on actual customers choices. The estimation procedure itself depends on several elements and underlying assumptions. As an example, a classical procedure to estimate the parameters of a Logit model is to maximize the log-likelihood function. Alternative methods include maximum simulated likelihood, simulated moments as well as Bayesian estimation. This topic is treated in detail in, e.g., Train (2009).

For the time parameters (\( T \)-parameters) in Table 1 we use the values estimated by Hansen and Pantuso (2018) on the actual transport services in the city of Milan in 2017. These values can also be found in the files accompanying the instance generator we make available online at https://github.com/GioPan/InstancesPricingAndRepositioningProblem. It should be noted that, in more general applications, the \( T \)-parameters for the different services might change significantly during the day as a result of issues such as different traffic patterns, road congestion or time-varying public transport schedules. Therefore, the \( T \)-parameters should be understood as specific for the target period under consideration.

The price parameters are set as follows. The price of a bicycle ride is set to \( P^B_{ij} = 0 \) for all \( (i, j) \) pairs, the price for public transport services is \( P^{PT,ij} = 2 \) (in Euro) for all \( (i, j) \) corresponding to the current price of an ordinary ticket valid for 90 minutes between each origin and destination within the municipality of Milan (price valid on November 2020). The per-minute fee of carsharing is set to \( P^V = 0.265 \) Euros per minute (the average of current per-minute fees offered by the CSOs in the city of Milan). The drop-off fees considered are \( L_1 = -2, L_2 = -1, L_3 = 0, L_4 = 1, L_5 = 2 \) Euro in the base case \((L = \{1,2,3,4,5\})\). Further analysis on the drop-off fee will be described in Section 6.

### 5.3. Individual customers profiles

The utility function in Section 5.2 entails that all customers within a given class (upper- or lower-middle class) are characterized by identical
preferences with respect to travel time, price, waiting time and walking time. Customers are told apart by their preferences with respect to unobserved features of the services, captured by $\xi_i$.

However, the availability of large amounts of customer data may allow the CSO to profile customers at the individual level, i.e., to assign each customer an individual utility function. We are not aware of publicly available utility functions which are able distinguish between individual customers. Therefore, in order to assess the effect of individual customer profiles, at least on the performance of the algorithm, we use an additional configuration in which an individual utility function for each customer is obtained by applying a random perturbation to the coefficients provided by Hansen and Pantuso (2018). Particularly, for each customer $k$, $\beta^k_d$ will be uniformly drawn in $[-188.33, -70.63]$, where $-188.33$ is the $\beta^d$ coefficient for lower-middle class customers and $-70.63$ is the $\beta^c$ coefficient for upper-middle class customers in the general case, see Section 5.2. This allows us to obtain customers which can be anywhere between the upper- and lower-middle class.

The remaining $\beta$ coefficients will be uniformly drawn in $[0.88, 1.29]$, where $\beta$ is the value provided by Hansen and Pantuso (2018). As an example, for each $k$, we will draw $\beta^c_d$ in $[-1.6, -2.4]$. The lower $\beta^c_d$ the less utility the customer will obtain for each minute spent in public transportation.

5.4. Uncertainty

The random term of the utility $\xi_i$ is modeled as a Gumbel (Extreme Value type I) distribution with mean 0 and standard deviation $\sigma$. This corresponds to using a Logit choice model (see Train, 2009; Ben-Akiva and Bierlaire, 1999). The value of $\sigma$ is set as the empirical standard deviation of $U_{ijk} = F_i(\rho_{ij}, t_{ij}, T_{ij}^{CS}, T_{ij}^{PT}, T_{ij}^{MR}, T_{ij}^{WM}, T_{ij}^{WM'})$ for all $i, j \in I$, $v \in V \cup A, k \in K_i$. This entails that the expectation term in the objective function of (1) (i.e., $Q(z, \lambda)$) is a multidimensional integral that makes the solution of the problem prohibitive. For this reason we approximate $\xi_i$ by iid samples drawn from its the underlying Gumbel distributions. The resulting discrete stochastic program goes under the name of Sample Average Approximation (SAA), see Kleijwegt et al. (2002). Its optimal objective value provides an unbiased estimator of the true objective value. The full model of the SAA is provided in Appendix A.

5.5. Position of customers and vehicles

We partition customers into sets $K_i$, and then further into sets $K_{ij}$ is such a way to test different configurations of demand, e.g., center to outskirt and vice-versa. Each one of the ten zones in our instances is characterized by a degree of “centrality”. We use the walking distance from the Dome of Milan as a proxy of centrality, see Fig. 3. Let $d_i$ be the walking distance from zone $i \in I$ to the Dome. Customers $K$ are first randomly partitioned into disjoint subsets $K_i$ with a probability $x_i$ which depends on the centrality of the zone as follows

$$x_i = \frac{\sum_{j \in I} d_{ij}}{\sum_{i \in I} d_{ij}} \tag{29}$$

where $y_i = e^{-a^F ROM} x_i$ with $a^F ROM \in [0, 1]$ and $a_4 = d_i \sum_{j \in I} d_{ij} / |I|$ is the deviation from the mean distance. In words, as $a^F ROM$ increases, the zones closer to the center (negative $a_4$) will receive a higher probability and the zones far from the center a lower probability, resulting in a higher concentration of customers in the central zones. Further, all customers assigned to a given zone $i$ will be randomly assigned a destination zone $j$, and thus inserted into subset $K_{ij}$, with a probability (29). This time $y_i = e^{-a^F ROM} x_i$ with $a^T O \in [0, 1]$. Again, as $a^T O$ increases more customers will be directed to central zones. As an example, setting a low value of $a^F ROM$ and a high value of $a^T O$ will create instances with higher demand from the outskirt to the center. The partitioning of customers is sketched in Algorithm 3.

Algorithm 3 Algorithm for the partition of customers into subsets $K_i$ and $K_{ij}, (i, j) \in I \times I$.

1: Input: $K_i, I, d_i, \text{for each } i \in I, a^F ROM \in [0, 1], a^T O \in [0, 1]$
2: $K_j = K_{ij} = \emptyset$ for $(i, j) \in I \times I$
3: for zone $i \in I$ do
4: Calculate $a_i = d_i - \sum_{j \in I} d_{ij} / |I|$
5: Calculate $y_i^F ROM = e^{-a^F ROM} x_i$ and $y_i^T O = e^{-a^T O} x_i$
6: end for
7: for zone $i \in I$ do
8: Calculate $x_i^F ROM = \frac{\sum_{j \in I} y_i^F ROM d_{ij} d_{ij}}{\sum_{j \in I} d_{ij}^2}$
9: Calculate $x_i^T O = \frac{\sum_{j \in I} y_i^T O d_{ij} d_{ij}}{\sum_{j \in I} d_{ij}^2}$
10: end for
11: for Customer $k \in K_i$ do
12: Draw an origin zone $i$ from $I$ according to the probability distribution $(x_i^F ROM)_{i \in I}$
13: $K_i \leftarrow K_i \cup \{k\}$
14: Draw a destination zone $j$ from $I$ according to the probability distribution $(x_i^T O)_{i \in I}$
15: $K_{ij} \leftarrow K_{ij} \cup \{k\}$
16: end for
17: return $K_i$ and $K_{ij}$ for $(i, j) \in I \times I$

We assume the decision maker is a CSO with a fleet of $|V|$ homogeneous vehicles. Each vehicle $v$ is randomly assigned to an initial zone $i$ according to probability (29), where $y_i^v = e^{-a^F ROM} x_i$ with $a^F ROM \in [0, 1]$. Also in this case, as $a^F ROM$ increases more cars will be initially located in central zones.

5.6. Costs

We assume a fleet of Fiat 500 cars with classical combustion engine. The relocation cost $C^T O$, equal for all vehicles, is set as the cost of the fuel necessary for a ride between $i$ and $j$, plus the per-minute salary of the driver multiplied by the driving time. The per-minute salary of the driver is set to 0.20 Euro/minute. It is calculated from the Italian national collective contract for logistics services valid at October 1st 2019 (available at https://www.lavoro-economia.it/ccnl/ccnl.aspx?c=328) as follows: the average per minute salary of the five lowest salary levels is increased by 30% to account for e.g., night shifts and holidays, yielding approximately 0.20 Euro/minute. Finally, the cost $C^T O$ is set equal to the fuel necessary for a ride between $i$ and $j$. The fuel consumption is calculated based on the specifics of a Fiat 500 petrol car and assuming an average speed of 50km/h and a fuel price of 1.60 Euro/liter. All the data necessary to generate the instances described, as well as an instance generator implemented in Java, are made available at https://github.com/GioPan/instancesPricingAndRepositioningProblem.

5.7. Control parameters

The control parameters for the instances used in the tests are summarized in Table 2. For each control parameter we report the different values used in the tests. The control parameters were chosen in order to test different vehicles-to-customers ratios, ranging from $1/2$ to $1/12$,
In addition, the different configurations of the parameters and different absolute values for the number of customers and vehicles. In addition, the different configurations of the parameters \( a^{\text{ROM}}, a^{\text{TO}} \) and \( a^{V} \) yield different configurations of demand (e.g., center to outskirt and outskirt to center) and of the carsharing system (e.g., vehicles located in the center and in the outskirt). The number of scenarios \(|S|\) is arbitrarily set equal to 10 on all instances. In Section 6.2 we provide insights on how the L-Shaped method scales with the number of scenarios. As explained in Section 5.4 scenarios represent i.i.d. samples of the underlying Gumbel distribution.

6. Results

This section is divided into three parts. First, we provide implementation details and setup of the experiments in Section 6.1. Following, in Section 6.2 we report on the performance of the L-Shaped method especially in comparison with a commercial solver. Finally, in Section 6.3 we provide an analysis of the solutions and comment on their managerial implications.

6.1. Experiments setup

The L-Shaped method and the extensive form the SAAs (see Appendix A) were implemented in Java using the CPLEX 12.10 libraries. Particularly, the L-Shaped method was implemented by solving the master problem in a Branch & Cut framework and adding optimality cuts as lazy constraints at integer nodes. Unless otherwise specified, we used CPLEX’s default parameters both when solving the extensive SAA and when using the L-Shaped method. This entails, e.g., a target relative optimality gap of 0.01%. The only exception, unless otherwise specified, is a time limit of 1800 seconds. In the L-Shaped method, relaxation cuts and optimality cuts were applied only at integer nodes throughout the entire tree. Particularly, relaxation cuts were crucial to the implementation. The performance of the algorithm without relaxation cuts was extremely poor. Tests were run on machines with 2x2.4 GHz AMD Opteron 2431 6 core CPU and 24Gb RAM. We remind the reader that, unless otherwise specified, the SAAs are solved with ten iid samples (scenarios) of random variable \( \xi \), see Section 5.4. We stress that the number of scenarios does not represent the number of instances, as in a scenario-analysis procedure. Rather, by definition, the stochastic program takes into account all scenarios simultaneously. The impact of the number of scenarios on the computational complexity of the problems solved is assessed in Section 6.2.

6.2. Analysis of the L-shaped method

In the first part of the tests we compared the performance of the L-Shaped method to that of CPLEX for solving the SAA on all configurations of the control parameters in Table 2 for which \(|K| > |V|\). The scope of our experiments is thus to obtain empirical evidences as to whether, and to what extent, the L-Shaped method scales better than using CPLEX without any decomposition strategy. The tables in this section report the optimality gap (\( \text{gap} \)) and elapsed time (\( t \) in seconds) for both CPLEX and the L-Shaped method. For the L-Shaped method they also report the optimality gap at the root node (\( \text{gap}_R \)) and after 50% of the time limit (\( \text{gap}_{50} \)). All gaps are expressed as percentages and are calculated as \( 100 \times |\text{bestbound} - \text{bestinteger}|/|\text{bestinteger}| \). The size of the SAA problem without decomposition is reported in Appendix B.

We start by reporting the results on the default setup, that is, in which customers are not profiled at the individual level (i.e., customers have identical sensitivities to prices, driving time, waiting and walking time, see Section 5.2). Tables 3 and 4 report the results on the instances with \( a^{V} = 0.2 \) (more vehicles initially located in zones far from the center) and \( a^{V} = 0.8 \) (more vehicles initially located in central zones),
appears sensibly lower compared to the results in Tables 3 and 4. Also a number of them. We observe also that the average optimality gap from 9.82% to 9.24% on the instances with \( \alpha = 0 \).\n
The results illustrated in Tables 3 and 4 are similar to those observed earlier in Tables 3 and 4. CPLEX remains a viable option only for the smallest instances, while the L-Shaped method is able to find a solution, in some cases a high quality one, to all instances and to solve a number of them. We observe also that the average optimality gap appears sensibly lower compared to the results in Tables 3 and 4. Also

respectively. Each table reports on a total of 32 instances, one for each configuration of the control parameters. The results in Tables 3 and 4 are rather similar. We observe that CPLEX is a viable alternative only for the smallest instances. As the number of vehicles grows CPLEX fails to deliver a feasible solution, and runs into memory problems. On the other hand, the L-Shaped method is able to provide a solution to all instances tested, and in many cases it provides a high quality solution, with a rather small optimality gap. We can also observe that, while the optimality gap at the root node is on average much higher than the final optimality gap, the gap after 50% of the allowed time is only a few percentage points higher. This illustrates, that the L-Shaped method may also deliver good solutions in a relatively short time (15 minutes).

A sensible reduction of the optimality gap can be achieved by applying valid inequality (27). Such valid inequality has the effect of removing some symmetric solutions, as explained in Section 4.5. We observed that the addition of (27) decreased the average optimality gap from 9.82% to 9.24% on the instances with \( \alpha = 0 \) and from 8.19% to 7.10% on the instances with \( \alpha = 0.8 \). The addition of (27) resulted particularly beneficial on the instances which yielded the largest optimality gaps reported in Tables 3 and 4. All the results on the performance of the L-Shaped method with the addition of (27) are reported in Appendix C.

We turn now our attention on the performance of the algorithm when each customer is profiled individually (see Section 5.3). The results in Tables 5 and 6 are obtained with \( \alpha = 0.2 \) and 0.8, respectively. Valid inequality (27) is always added to the models.

The results illustrated in Tables 5 and 6 are similar to those observed earlier in Tables 3 and 4. CPLEX remains a viable option only for the smallest instances, while the L-Shaped method is able to find a solution, in some cases a high quality one, to all instances and to solve a number of them. We observe also that the average optimality gap appears sensibly lower compared to the results in Tables 3 and 4. Also

respectively. Each table reports on a total of 32 instances, one for each configuration of the control parameters. The results in Tables 3 and 4 are rather similar. We observe that CPLEX is a viable alternative only for the smallest instances. As the number of vehicles grows CPLEX fails to deliver a feasible solution, and runs into memory problems. On the other hand, the L-Shaped method is able to provide a solution to all instances tested, and in many cases it provides a high quality solution, with a rather small optimality gap. We can also observe that, while the optimality gap at the root node is on average much higher than the final optimality gap, the gap after 50% of the allowed time is only a few percentage points higher. This illustrates, that the L-Shaped method may also deliver good solutions in a relatively short time (15 minutes).

A sensible reduction of the optimality gap can be achieved by applying valid inequality (27). Such valid inequality has the effect of removing some symmetric solutions, as explained in Section 4.5. We observed that the addition of (27) decreased the average optimality gap from 9.82% to 9.24% on the instances with \( \alpha = 0 \) and from 8.19% to 7.10% on the instances with \( \alpha = 0.8 \). The addition of (27) resulted particularly beneficial on the instances which yielded the largest optimality gaps reported in Tables 3 and 4. All the results on the performance of the L-Shaped method with the addition of (27) are reported in Appendix C.

We turn now our attention on the performance of the algorithm when each customer is profiled individually (see Section 5.3). The results in Tables 5 and 6 are obtained with \( \alpha = 0.2 \) and 0.8, respectively. Valid inequality (27) is always added to the models.

The results illustrated in Tables 5 and 6 are similar to those observed earlier in Tables 3 and 4. CPLEX remains a viable option only for the smallest instances, while the L-Shaped method is able to find a solution, in some cases a high quality one, to all instances and to solve a number of them. We observe also that the average optimality gap appears sensibly lower compared to the results in Tables 3 and 4. Also

respectively. Each table reports on a total of 32 instances, one for each configuration of the control parameters. The results in Tables 3 and 4 are rather similar. We observe that CPLEX is a viable alternative only for the smallest instances. As the number of vehicles grows CPLEX fails to deliver a feasible solution, and runs into memory problems. On the other hand, the L-Shaped method is able to provide a solution to all instances tested, and in many cases it provides a high quality solution, with a rather small optimality gap. We can also observe that, while the optimality gap at the root node is on average much higher than the final optimality gap, the gap after 50% of the allowed time is only a few percentage points higher. This illustrates, that the L-Shaped method may also deliver good solutions in a relatively short time (15 minutes).

A sensible reduction of the optimality gap can be achieved by applying valid inequality (27). Such valid inequality has the effect of removing some symmetric solutions, as explained in Section 4.5. We observed that the addition of (27) decreased the average optimality gap from 9.82% to 9.24% on the instances with \( \alpha = 0 \) and from 8.19% to 7.10% on the instances with \( \alpha = 0.8 \). The addition of (27) resulted particularly beneficial on the instances which yielded the largest optimality gaps reported in Tables 3 and 4. All the results on the performance of the L-Shaped method with the addition of (27) are reported in Appendix C.

We turn now our attention on the performance of the algorithm when each customer is profiled individually (see Section 5.3). The results in Tables 5 and 6 are obtained with \( \alpha = 0.2 \) and 0.8, respectively. Valid inequality (27) is always added to the models.

The results illustrated in Tables 5 and 6 are similar to those observed earlier in Tables 3 and 4. CPLEX remains a viable option only for the smallest instances, while the L-Shaped method is able to find a solution, in some cases a high quality one, to all instances and to solve a number of them. We observe also that the average optimality gap appears sensibly lower compared to the results in Tables 3 and 4. Also

respectively. Each table reports on a total of 32 instances, one for each configuration of the control parameters. The results in Tables 3 and 4 are rather similar. We observe that CPLEX is a viable alternative only for the smallest instances. As the number of vehicles grows CPLEX fails to deliver a feasible solution, and runs into memory problems. On the other hand, the L-Shaped method is able to provide a solution to all instances tested, and in many cases it provides a high quality solution, with a rather small optimality gap. We can also observe that, while the optimality gap at the root node is on average much higher than the final optimality gap, the gap after 50% of the allowed time is only a few percentage points higher. This illustrates, that the L-Shaped method may also deliver good solutions in a relatively short time (15 minutes).

A sensible reduction of the optimality gap can be achieved by applying valid inequality (27). Such valid inequality has the effect of removing some symmetric solutions, as explained in Section 4.5. We observed that the addition of (27) decreased the average optimality gap from 9.82% to 9.24% on the instances with \( \alpha = 0 \) and from 8.19% to 7.10% on the instances with \( \alpha = 0.8 \). The addition of (27) resulted particularly beneficial on the instances which yielded the largest optimality gaps reported in Tables 3 and 4. All the results on the performance of the L-Shaped method with the addition of (27) are reported in Appendix C.

We turn now our attention on the performance of the algorithm when each customer is profiled individually (see Section 5.3). The results in Tables 5 and 6 are obtained with \( \alpha = 0.2 \) and 0.8, respectively. Valid inequality (27) is always added to the models.

The results illustrated in Tables 5 and 6 are similar to those observed earlier in Tables 3 and 4. CPLEX remains a viable option only for the smallest instances, while the L-Shaped method is able to find a solution, in some cases a high quality one, to all instances and to solve a number of them. We observe also that the average optimality gap appears sensibly lower compared to the results in Tables 3 and 4. Also

respectively. Each table reports on a total of 32 instances, one for each configuration of the control parameters. The results in Tables 3 and 4 are rather similar. We observe that CPLEX is a viable alternative only for the smallest instances. As the number of vehicles grows CPLEX fails to deliver a feasible solution, and runs into memory problems. On the other hand, the L-Shaped method is able to provide a solution to all instances tested, and in many cases it provides a high quality solution, with a rather small optimality gap. We can also observe that, while the optimality gap at the root node is on average much higher than the final optimality gap, the gap after 50% of the allowed time is only a few percentage points higher. This illustrates, that the L-Shaped method may also deliver good solutions in a relatively short time (15 minutes).

A sensible reduction of the optimality gap can be achieved by applying valid inequality (27). Such valid inequality has the effect of removing some symmetric solutions, as explained in Section 4.5. We observed that the addition of (27) decreased the average optimality gap from 9.82% to 9.24% on the instances with \( \alpha = 0 \) and from 8.19% to 7.10% on the instances with \( \alpha = 0.8 \). The addition of (27) resulted particularly beneficial on the instances which yielded the largest optimality gaps reported in Tables 3 and 4. All the results on the performance of the L-Shaped method with the addition of (27) are reported in Appendix C.

We turn now our attention on the performance of the algorithm when each customer is profiled individually (see Section 5.3). The results in Tables 5 and 6 are obtained with \( \alpha = 0.2 \) and 0.8, respectively. Valid inequality (27) is always added to the models.

The results illustrated in Tables 5 and 6 are similar to those observed earlier in Tables 3 and 4. CPLEX remains a viable option only for the smallest instances, while the L-Shaped method is able to find a solution, in some cases a high quality one, to all instances and to solve a number of them. We observe also that the average optimality gap appears sensibly lower compared to the results in Tables 3 and 4. Also
using ten scenarios to approximate the underlying continuous random variable. Figs. 4 and 5 report the optimality gap obtained with the L-Shaped method and CPLEX as the sample size increases, for the smallest (|V| = 50, |K| = 200) and largest (|V| = 200, |K| = 600) instances, respectively, with identical customers. Particularly, Fig. 4(a) and Fig. 4(b) report the gap of the L-Shaped method after 30 minutes and 5 hours, respectively, while Figs. 4(c) and 4(d) report the gap of CPLEX after 30 minutes and 5 hours, respectively. Similarly, Fig. 5 reports the gaps of the L-Shaped method after 30 minutes and 5 hours on the largest instance. Tests on the largest instances were conducted only for the L-Shaped method as it is already evident from Tables 3 and 4 that, on those instances, CPLEX fails to deliver solutions already with a sample of ten scenarios.

As intuitions suggests, when using the L-Shaped method, the optimality gap grows with the number of scenarios, both on the smallest (Fig. 4) and the largest (Fig. 5) instances. For the smallest instances, with a time limit of 30 minutes the growth appears mild, and with 100 scenarios, the optimality gap remains in the neighborhood of 4% in the worst case, with an average optimality gap in the neighborhood of 2% (see Fig. 4(a)). After 5 hours, the L-Shaped method drops the optimality gap even further, with a worst case gap in the neighborhoods of 2%. This gives room for a more dense approximation of the uncertainty, i.e., a larger sample size, compared to the 10 scenarios used in our previous tests. The performance of CPLEX on the same instances is dramatically worse (see Fig. 4(c)). Given a 30-minute time limit, on the smallest sample sizes, CPLEX outperforms the L-Shaped method, but as the sample size grows to 50 or higher the solver’s optimality gaps are orders of magnitude higher than those of the L-Shaped method (compare Fig. 4(c) and Fig. 4(a)). The performance of CPLEX improves with a 5 hours time limit (see Fig. 4(d)), though performing much worse than the L-Shaped method at least for the largest sample sizes (compare Fig. 4(d) and Fig. 4(b)). On the largest instances, with a 30-minute time limit, the gap growth for the L-Shaped method remains limited up to a sample size of 25, but grows dramatically with a larger sample size, see Fig. 5(a). Also the variance of the optimality gap grows with the sample size, limiting the reliability of the method for large numbers of scenarios. Nevertheless, with a longer time limit the L-Shaped method is able to reduce the optimality gap approximately ten times on the largest sample sizes, see Fig. 5(b).

Summarizing, the results in Figs. 4 and 5 illustrate that, for small instances, a solution time of 30 minutes is sufficient to obtain high quality solutions also with a more extensive approximation of the uncertainty. However, especially on the largest instances, a 30-minute time limit might result too small to accommodate for a better description of the uncertainty. Nevertheless, allowing the L-Shaped method to run for a longer time (e.g., 5 hours) can yield substantial reductions of the optimality gap. In any case, Fig. 4 illustrates that the L-Shaped method scales significantly better than CPLEX as the sample size increases.

The observed performance of the L-Shaped method allows us a conclusive reflection on the envisaged usage of the method. In Tables 3-6 we have let both the algorithm and CPLEX run for 30 minutes and observed that our algorithm scales significantly better. Therefore, depending on the practical operating needs of the CSO, the algorithm might already provide a practice-ready tool. That is, if the CSO is able to wait for a solution for 30 minutes, our tests provide empirical evidences that the algorithm delivers a solution and often a high-quality one, with the vehicles-to-customers ratio being a strong driver of the quality of the solution obtained. Tables 3-6 also illustrate that the algorithm was able to find good solutions after 15 minutes (50% of the solution time, see column gap50). Therefore, if the CSO has tighter time requirements, a potential strategy is to terminate the algorithm earlier, knowing that this implies giving up something in terms of quality of the solution. However, our tests show that the optimality gap after 15 minutes is typical not dramatically higher than the optimality gap.
obtained after 30 minutes. Nevertheless, there might arise situations in which waiting for a solution for 30 or even 15 minutes might be impractical, e.g., if the demand landscape changes more frequently and relocation and pricing plans are required more often. In these cases, the proposed algorithm might prove inefficient and one might have to consider developing faster heuristic algorithms. If this is the case, the performance of the L-Shaped method, and the dual bounds it delivers, provide a reliable benchmark. An example is provided in Appendix E where we test a simple Iterated Local Search and assess its performances using the bounds provided by the L-Shaped method. For the smallest instances the heuristic is able to provide primal solutions of quality comparable or even better than the L-Shaped method. Nevertheless, the quality of the solutions delivered drops significantly as the size of the instance increase, indicating that further refinement is needed. Finally, regardless of the solution time, the algorithm proposed may be used by the CSO to obtain solutions that allow them to support managerial choices or analyze policy implications, e.g., subsidies, plans to expand the fleet or to hire additional staff for relocations activities. Some insights on the impact of a pricing scheme on profits and demand are provided in Section 6.3.

6.3. Analysis of the solutions

In this section we present some evidences based on the analysis of the solutions obtained by the proposed model. The analysis was performed using the default configuration, i.e., without individual customer profiles, as we believe it is a more realistic configuration to achieve by CSOs.

The analysis in this section is based on the results obtained on the instances with the largest number of customers (600) and with different distributions of vehicles and customers, namely

<table>
<thead>
<tr>
<th>V</th>
<th>K</th>
<th>V̄ FROM</th>
<th>V̄ TO</th>
<th>CPLEX</th>
<th>L-Shaped</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>gap</td>
<td>t</td>
<td>gap</td>
<td>gap</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>0.2</td>
<td>0.2</td>
<td>0.0044</td>
<td>311.01</td>
</tr>
<tr>
<td>50</td>
<td>400</td>
<td>0.2</td>
<td>0.2</td>
<td>1.5001</td>
<td>1802.42</td>
</tr>
<tr>
<td>50</td>
<td>600</td>
<td>0.2</td>
<td>0.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>0.2</td>
<td>0.2</td>
<td>0.0099</td>
<td>1246.59</td>
</tr>
<tr>
<td>100</td>
<td>400</td>
<td>0.2</td>
<td>0.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>100</td>
<td>600</td>
<td>0.2</td>
<td>0.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>200</td>
<td>400</td>
<td>0.2</td>
<td>0.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>200</td>
<td>600</td>
<td>0.2</td>
<td>0.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>0.8</td>
<td>0.8</td>
<td>0.0097</td>
<td>375.72</td>
</tr>
<tr>
<td>50</td>
<td>400</td>
<td>0.8</td>
<td>0.8</td>
<td>0.9894</td>
<td>1802.12</td>
</tr>
<tr>
<td>50</td>
<td>600</td>
<td>0.8</td>
<td>0.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>0.8</td>
<td>0.8</td>
<td>0.0091</td>
<td>671.58</td>
</tr>
<tr>
<td>100</td>
<td>400</td>
<td>0.8</td>
<td>0.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>100</td>
<td>600</td>
<td>0.8</td>
<td>0.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>200</td>
<td>400</td>
<td>0.8</td>
<td>0.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>200</td>
<td>600</td>
<td>0.8</td>
<td>0.8</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

D1 Vehicles in the outskirt and demand from center to outskirt \( (V = 0.2, \ V̄ FROM = 0.8, \ V̄ TO = 0.2) \)

D2 Vehicles in the center and demand from outskirt to center \( (V = 0.8, \ V̄ FROM = 0.2, \ V̄ TO = 0.8) \)

D3 Vehicles in the center and demand from center to outskirt \( (V = 0.8, \ V̄ FROM = 0.8, \ V̄ TO = 0.2) \)

D4 Vehicles in the outskirt and demand from outskirt to center \( (V = 0.2, \ V̄ FROM = 0.2, \ V̄ TO = 0.8) \)

The number of vehicles is set either to 50 or to 200, corresponding to vehicles-to-customers ratios of 1/12 and 1/3, respectively. It should be noted that, as pointed out by e.g., Huang et al. (2018), Jian et al. (2019), the number of available cars in a zone influences customer demand. In our approach, in which customers are considered at the individual level, the connection between the number of available cars and demand is handled jointly by the utility function and, especially, by the optimization model. That is, there is a potential demand, made of the users which, according to their utility function, would choose carsharing, if available, at the drop-off fee level set in the first stage, and there is a realized demand (i.e., actual rentals), which takes into account that not all potential customers may find an available car. This is done through constraints (9) that state that an available car is taken by the first customer arriving at the car. It should be further clarified that, in our experiments, we assume that customers do not wait for more cars to become available, i.e., the waiting time is always set to zero in our instances, corresponding to saying that, if no car is available, the customer will immediately choose another transport service. Indeed, in some real-life carsharing services, some waiting time could be taken into account. That is, when there are no cars available in the zone, some user might decide to wait until a car is returned.
However, we believe the assumption that customers do not wait is the most appropriate especially in a free-floating one-way service, where both customers and the CSO have limited information on whether and when a new car will be returned close to the user.

We start by presenting the effect of pricing strategies on profits and relocations. For each distribution of customers and vehicles, we solved two configurations of the model. In the first configuration the prices were optimally set by model (1). In the second configuration the drop-off fee was set to 0 (the average of the drop-off fees considered) everywhere, corresponding to a situation in which the CSO applies only a per-minute fee and does not adjust prices with respect to the origin and destination and to the time of the day.

Tables 9 and 10 report a number of solution statistics for the case with 50 and 200 customers, respectively. Expected profits for the case without dynamic pricing are reported as a percentage of the expected profits with dynamic pricing. In both the case with 50 and 200 vehicles we observe that the expected profit without pricing is approximately 65 to 80% of the expected profit obtained by adjusting prices. The main driver of the higher expected profit generated by a pricing strategy is the higher number of requests generated, approximately double both in the case with 50 and in the case with 200 vehicles. That is, by adjusting prices the CSO is able to attract significantly more demand and increase competition.
In the case with 200 vehicles and without dynamic pricing, the CSO is able to satisfy the great majority of the requests (more than 90% – see Table 10) performing very few, if any, relocations – we will return to this point later. However, in this case the CSO is able to attract less than (approximately) 1/6 of the customers. With dynamic pricing, the CSO is able to attract close to 1/3 of the customers and serve slightly more than 50% of the requests. In total, the number of rentals are approximately the same, with and without dynamic pricing. However, by adjusting prices the CSO is able to increase the revenue. In fact, as shown in Fig. 6, the most used drop-off fee is the highest (2 Euro), illustrating that the CSO is able to exploit the higher willingness to pay of some customers.

In the case with 50 vehicles (thus one vehicle every twelve potential customers), without dynamic pricing the CSO is able to satisfy approximately half of the total requests due to the reduced number of vehicles, see Table 9. Also in this case the number of requests is lower (approximately half) then the number of requests obtained by adjusting prices. With dynamic pricing, the CSO is able to attract close to 1/3 of the customers and to satisfy only approximately 25% of them. Also in this case, the CSO benefits from the higher competition. In more than 50% of the origin–destination pairs the CSO is able to apply the highest drop-off fee (2 Euro) see Fig. 7 and to reposition the fleet in such a way to satisfy the requests with the highest revenue. Thus, a trend we observe in Table 9 is that a pricing strategy allows the CSO to attract more demand but to satisfy only part of it. While this allows the CSO to exploit competition, many customers do not see their wish to use carsharing satisfied. This negative user experience might have an impact in the long run. This is however beyond the scope of this study.

Interestingly, in the case with 200 vehicles (one every three customers–see Table 10) the need for relocations is almost null, regardless of how prices are set. The fleet is large enough to cover sufficiently well the entire business area and serve almost all requests. On the other hand, with a fleet 50 vehicles (one every twelve customers–see Table 9) the need for relocations is more evident. The fleet is now insufficient to cover the entire demand. In the case without dynamic pricing, fewer relocations are needed compared to the case with dynamic pricing. This is due to the lower demand attracted (approximately 80 to 107 requests with a fleet of 50 vehicles). Many more relocations are performed when dynamic pricing is applied as a consequence of the higher demand generated (at least 167 requests for 50 vehicles). Thus the CSO finds it beneficial to move vehicles where they can generate more revenue.

A natural follow up question is the impact of relocations on profits. Therefore, we focused on those instances where more relocations were suggested, i.e., with 50 vehicles and dynamic pricing, see Table 9. We solved the same instances, but this time preventing the model from making any relocations. That is, vehicles were forced to remain in their initial positions. The results indicate that the expected profit without relocations is only marginally lower. Particularly, 98.5% for distribution D1, 98.3% for D2, 97.4% for D3 and 98.8% for D4. This is due to the fact that relocations are expensive and can yield only a very minor increase in revenue. That is, vehicles would always be rented, even if not relocated. By relocating a vehicle the CSO is able to charge a higher drop-off fee, but bears the relocation cost. This results in a marginal profit increase.
Relocations are however likely to generate a higher impact on profits when the vehicles-to-customers ratio is even smaller. We performed the same test with a vehicles-to-customers ratio equal to 1/100 (i.e., 10 vehicles 1000 customers). The results show that the profit without relocations was 70.07% with distribution $D_1$, 75.26% with $D_2$, 76.22% with distribution $D_3$ and 78.07% with distribution $D_4$ (these percentages are calculated on the best upper bounds since a near optimal solution was available for all instances). In all cases the percentage of vehicles relocated ranged between 10 and 20%, similar to the case with 50 vehicles and 600 customers. This means that the percentage of relocations remained approximately the same with a smaller vehicles-to-customers ratio, but had a much higher impact on profits. Thus, it appears that a dynamic pricing strategy, coupled with a sufficiently large fleet (say more than one vehicle every twelve customers in our case), decreases significantly the need for staff-based relocations. Otherwise, relocations remain an important tool even with a dynamic pricing strategy.

7. Conclusions, limitations and future work

We presented a novel optimization model for jointly deciding car-sharing prices and relocations. The problem is modeled as a two-stage integer stochastic program in order to account for uncertainty in customers preferences. An exact solution algorithm based on the integer L-Shaped method has been proposed. Extensive tests have been performed on instances based on the municipality of Milan, in order to assess both the performance of the solution algorithm and the type of solutions obtained. The instances have been made available online for the sake of future research.

Results illustrate that, within times compatible with business practice, the method solves or finds a high quality solution to most instances. In addition, it finds a feasible solution to all instances considered. In contrast, CPLEX delivers a solution to only a few, small, instances.

The analysis of the solutions illustrates that a pricing strategy helps the CSO to significantly increase expected profits. This is due to the increased demand generated and the resulting competition. In our instances the demand was approximately doubled compared to a situation without dynamic pricing. This in turn generates higher expected profits by exploiting customer’s higher willingness to pay. The results also show that, by adopting a zone-based pricing strategy and employing a large enough fleet, the impact of staff-based relocations on profits becomes marginal. On the other hand, the impact of relocations becomes more evident as the size of the fleet decreases.

A number of limitations remain to be addressed in future research, as we comment in what follows. A pricing strategy which varies with each origin and destination, or frequently throughout the day, may not be applicable in all contexts, or raise concerns related to the potential complexity for users who would rather prefer a simpler pricing strategy. The scope of this article was that of introducing a general model framework which could then be adapted to specific contexts and improved. For example, the model proposed can be easily adapted to different time and space resolutions, i.e., it is possible to define the length of the target period and the discretization of the business area based on the...
Results of the L-Shaped method with the addition of Eq. (27) on the instances with $\alpha^2 = 0.2$ with a time limit of 18000 seconds and 1% target optimality gap.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>$u^{\text{ROM}}$</th>
<th>$u^{\text{TO}}$</th>
<th>$\text{gapPR}$</th>
<th>$\text{gap50}$</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>200</td>
<td>0.2</td>
<td>0.2</td>
<td>0.9890</td>
<td>18.6752</td>
<td>–</td>
</tr>
<tr>
<td>50</td>
<td>400</td>
<td>0.2</td>
<td>0.2</td>
<td>10.2092</td>
<td>66.5573</td>
<td>11.1192</td>
</tr>
<tr>
<td>50</td>
<td>600</td>
<td>0.2</td>
<td>0.2</td>
<td>22.9573</td>
<td>87.2480</td>
<td>23.9743</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>0.2</td>
<td>0.2</td>
<td>0.6386</td>
<td>9.5524</td>
<td>–</td>
</tr>
<tr>
<td>100</td>
<td>400</td>
<td>0.2</td>
<td>0.2</td>
<td>0.9219</td>
<td>21.2273</td>
<td>–</td>
</tr>
<tr>
<td>100</td>
<td>600</td>
<td>0.2</td>
<td>0.2</td>
<td>7.4259</td>
<td>35.8381</td>
<td>8.8381</td>
</tr>
<tr>
<td>200</td>
<td>400</td>
<td>0.2</td>
<td>0.2</td>
<td>0.0000</td>
<td>6.3509</td>
<td>–</td>
</tr>
<tr>
<td>200</td>
<td>600</td>
<td>0.2</td>
<td>0.2</td>
<td>0.8479</td>
<td>10.4516</td>
<td>–</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>0.2</td>
<td>0.8</td>
<td>0.8965</td>
<td>17.6529</td>
<td>–</td>
</tr>
<tr>
<td>50</td>
<td>400</td>
<td>0.2</td>
<td>0.8</td>
<td>6.6519</td>
<td>62.4842</td>
<td>7.6630</td>
</tr>
<tr>
<td>50</td>
<td>600</td>
<td>0.2</td>
<td>0.8</td>
<td>13.2604</td>
<td>90.2582</td>
<td>13.8926</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>0.2</td>
<td>0.8</td>
<td>0.3083</td>
<td>7.2972</td>
<td>–</td>
</tr>
<tr>
<td>100</td>
<td>400</td>
<td>0.2</td>
<td>0.8</td>
<td>0.9374</td>
<td>19.6630</td>
<td>–</td>
</tr>
<tr>
<td>100</td>
<td>600</td>
<td>0.2</td>
<td>0.8</td>
<td>6.5696</td>
<td>34.0425</td>
<td>6.9128</td>
</tr>
<tr>
<td>200</td>
<td>400</td>
<td>0.2</td>
<td>0.8</td>
<td>0.0000</td>
<td>41.7344</td>
<td>–</td>
</tr>
<tr>
<td>200</td>
<td>600</td>
<td>0.2</td>
<td>0.8</td>
<td>0.7697</td>
<td>12.6665</td>
<td>–</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>0.8</td>
<td>0.2</td>
<td>1.3911</td>
<td>23.9553</td>
<td>1.5307</td>
</tr>
<tr>
<td>50</td>
<td>400</td>
<td>0.8</td>
<td>0.2</td>
<td>15.1496</td>
<td>85.1665</td>
<td>15.6460</td>
</tr>
<tr>
<td>50</td>
<td>600</td>
<td>0.8</td>
<td>0.2</td>
<td>36.5586</td>
<td>108.3737</td>
<td>37.5433</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>0.8</td>
<td>0.2</td>
<td>0.9165</td>
<td>15.4557</td>
<td>–</td>
</tr>
<tr>
<td>100</td>
<td>400</td>
<td>0.8</td>
<td>0.2</td>
<td>2.7552</td>
<td>22.7406</td>
<td>3.2210</td>
</tr>
<tr>
<td>100</td>
<td>600</td>
<td>0.8</td>
<td>0.2</td>
<td>13.9898</td>
<td>42.5987</td>
<td>14.5641</td>
</tr>
<tr>
<td>200</td>
<td>400</td>
<td>0.8</td>
<td>0.2</td>
<td>0.4081</td>
<td>12.4280</td>
<td>–</td>
</tr>
<tr>
<td>200</td>
<td>600</td>
<td>0.8</td>
<td>0.2</td>
<td>1.2695</td>
<td>15.3980</td>
<td>2.6600</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>0.8</td>
<td>0.8</td>
<td>0.9616</td>
<td>22.5871</td>
<td>–</td>
</tr>
<tr>
<td>50</td>
<td>400</td>
<td>0.8</td>
<td>0.8</td>
<td>11.8177</td>
<td>55.7576</td>
<td>13.0508</td>
</tr>
<tr>
<td>50</td>
<td>600</td>
<td>0.8</td>
<td>0.8</td>
<td>33.2157</td>
<td>106.8156</td>
<td>34.2212</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>0.8</td>
<td>0.8</td>
<td>0.9309</td>
<td>13.3519</td>
<td>–</td>
</tr>
<tr>
<td>100</td>
<td>400</td>
<td>0.8</td>
<td>0.8</td>
<td>1.8436</td>
<td>17.1371</td>
<td>2.0311</td>
</tr>
<tr>
<td>100</td>
<td>600</td>
<td>0.8</td>
<td>0.8</td>
<td>10.3464</td>
<td>46.7222</td>
<td>15.3117</td>
</tr>
<tr>
<td>200</td>
<td>400</td>
<td>0.8</td>
<td>0.8</td>
<td>0.5470</td>
<td>7.2607</td>
<td>–</td>
</tr>
<tr>
<td>200</td>
<td>600</td>
<td>0.8</td>
<td>0.8</td>
<td>2.0970</td>
<td>16.7444</td>
<td>2.7073</td>
</tr>
</tbody>
</table>

Comparison of the solutions with and without dynamic pricing on the instances with 50 vehicles and 600 customers.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Metric</th>
<th>With dynamic pricing</th>
<th>Without dynamic pricing</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Expected Profit [%]</td>
<td>100</td>
<td>81.78</td>
</tr>
<tr>
<td>Min</td>
<td>$\text{Max} {\text{gap}}$</td>
<td>167</td>
<td>80</td>
</tr>
<tr>
<td>Max</td>
<td>$\text{Max} {\text{gap}}$</td>
<td>195</td>
<td>107</td>
</tr>
<tr>
<td>Expected % Requests satisfied</td>
<td>24</td>
<td>42</td>
<td></td>
</tr>
</tbody>
</table>

Comparison of the solutions with and without dynamic pricing on the instances with 50 vehicles and 600 customers.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Metric</th>
<th>With dynamic pricing</th>
<th>Without dynamic pricing</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Expected Profit [%]</td>
<td>100</td>
<td>70.12</td>
</tr>
<tr>
<td>Min</td>
<td>$\text{Max} {\text{gap}}$</td>
<td>168</td>
<td>80</td>
</tr>
<tr>
<td>Max</td>
<td>$\text{Max} {\text{gap}}$</td>
<td>195</td>
<td>107</td>
</tr>
<tr>
<td>Expected % Requests satisfied</td>
<td>53</td>
<td>90</td>
<td></td>
</tr>
</tbody>
</table>

Comparison of the solutions with and without dynamic pricing on the instances with 50 vehicles and 600 customers.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Metric</th>
<th>With dynamic pricing</th>
<th>Without dynamic pricing</th>
</tr>
</thead>
<tbody>
<tr>
<td>D2</td>
<td>Expected Profit [%]</td>
<td>100</td>
<td>70.87</td>
</tr>
<tr>
<td>Min</td>
<td>$\text{Max} {\text{gap}}$</td>
<td>168</td>
<td>80</td>
</tr>
<tr>
<td>Max</td>
<td>$\text{Max} {\text{gap}}$</td>
<td>195</td>
<td>107</td>
</tr>
<tr>
<td>Expected % Requests satisfied</td>
<td>56</td>
<td>94</td>
<td></td>
</tr>
</tbody>
</table>

Comparison of the solutions with and without dynamic pricing on the instances with 50 vehicles and 600 customers.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Metric</th>
<th>With dynamic pricing</th>
<th>Without dynamic pricing</th>
</tr>
</thead>
<tbody>
<tr>
<td>D3</td>
<td>Expected Profit [%]</td>
<td>100</td>
<td>71.18</td>
</tr>
<tr>
<td>Min</td>
<td>$\text{Max} {\text{gap}}$</td>
<td>168</td>
<td>82</td>
</tr>
<tr>
<td>Max</td>
<td>$\text{Max} {\text{gap}}$</td>
<td>187</td>
<td>99</td>
</tr>
<tr>
<td>Expected % Requests satisfied</td>
<td>56</td>
<td>94</td>
<td></td>
</tr>
</tbody>
</table>

specific needs. In addition, the model may be easily modified to enforce that e.g., drop-off fees vary only according to the pick-up place or only according to the drop-off place. Future research may provide further modifications and improvements.

The size of the instances used in this study is comparable with the size of the station-based carsharing in Milan which, according to SIAD (2020), in 2018 counted 149 shared vehicles and, on average 108 daily rentals (see Table 10 for a comparison). Other examples are the station-based carsharing offered by Letgo (https://letgo.dk/), which currently operates a fleet of around 200 vehicles in Copenhagen, and VY (https://www.vy.dk/en/travelling-with-us/other-modes-of-transport/city-car/) that operates a fleet of 250 vehicles in Oslo. Nevertheless, bigger fleets and a higher number of customers are likely to limit the practical efficiency of our exact method and call for faster, e.g., heuristic, methods.

The performance of the algorithm with respect to a higher number of zones remains to be assessed. Our instances, generated on the basis of Hansen and Pantuso (2018), contained ten zones. Supposedly, a
more granular discretization of the business area is likely to have a negative impact on the practical applicability of the method. However, the benefits of a finer partition of the business area into pricing zones is to be addressed by further research, particularly in the case of free-floating services. Effective discretization strategies and methods are, to our knowledge, still an open research question.

As reported by Zoepf and Keith (2016), the choice of carsharing users is also influenced by elements such as the type of vehicle and its proximity to the user. In addition, comfort, weather conditions, and purpose of the trip are all factors which might influence customers decisions. While proximity is considered in the form of walking time in the utility function we used in our experiments, the remaining elements are not captured explicitly, but are rather included in the portion of customer preferences that the CSO cannot explain. Future research might be set up to extend the model and utility function used in the tests in order to better capture customers behavior.

Several other sources of uncertainty affect the problem, that have not been considered in this study. These include, e.g., the total number of customers appearing in each zone, and their destination. Our model might account for this uncertainty by setting a sufficiently large number of customers. Travel times, both with carsharing and with alternative transport services are also, to a certain degree, uncertain in practice. The impact of this uncertainty on solutions and profits remains to be understood.

The analysis of the solutions indicates that, by dynamically adjusting prices, the CSO is able to attract significantly more demand. However, with a vehicles-to-customers ratio of 1 to 12 (see Table 9), the portion of the demand satisfied was, approximately, only 25%. That is, the majority of the customers who would have used carsharing did not have the chance to do so. As a consequence, users may perceive a low availability of the service. The effect of this in the long term remains to be clarified.
Finally, our model is currently unable to use pricing as a preventive measure to encourage a profitable distribution of the fleet. Consider two subsequent target periods, say \(t_1\) and \(t_2\), and two zones, say \(A\) and \(B\). Assume that the CSO expects high demand in zone \(A\) in period \(t_2\). They may, consequently, set a lower price or an incentive in \(t_1\), for renting cars in zone \(B\) and delivering them in zone \(A\), and/or disincetivize movements in the opposite direction. In order to be able to optimize these decisions the proposed model should be extended to account for a multistage decision process.

### CRediT authorship contribution statement

Giovanni Pantuso: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization.

### Appendix A. Sample average approximation

Let \(\xi_1, \ldots, \xi_s\) be an \(S\)-dimensional iid sample of \(\xi\) and let \(S = \{1, \ldots, S\}\). Let decision variable \(y_{rls}\) be equal to 1 if request \(r \in R(\xi_i)\) is satisfied by vehicle \(v\) at level \(l\) under realization \(s\), 0, otherwise. The SAA of problem (1) can be stated as follows.

\[
\max \sum_{r \in R(\xi_1)} \sum_{l} \sum_{s} r_{rls} + \frac{1}{S} \sum_{s} \sum_{r \in R(\xi_1)} \sum_{l} r_{rls} y_{rls} \quad (30a)
\]

\[
\sum_{v \in V} z_{vl} = 1 \quad \forall l \in L \quad (30b)
\]

\[
\sum_{i} \lambda_{ijt} = 1 \quad i \in I, j \in J \quad (30c)
\]

\[
\sum_{v \in V} y_{rls} \leq 1 \quad r \in R(\xi_i), s \in S \quad (30d)
\]

### Appendix B. Size of the instances

This section reports the size of the instances for the base case in Table 11 and for the case with customers profiled individually in Table 12.

### Appendix C. Effect of valid inequality

See Tables 13 and 14.

### Appendix D. Results on the instances with individual customer profiles after five hours

See Tables 15 and 16.
Appendix E. Example heuristic

In this section we present a simple Iterated Local Search (ILS) to find primal solutions to the problem. In a nutshell, an ILS works as follows: given an initial (current at a generic iteration) solution it performs a local search. To escape local optima, the solution returned by the local search is randomly perturbed and the local search restarted. This procedure is repeated until a stopping criteria is met. In what follows we explain how this procedure is adapted to our problem.

Algorithm 4 Local Search.

1: INPUT: $\zeta$, Operator $f(\zeta)$ to use (i.e., $f^H(\zeta)$ or $f^A(\zeta)$).
2: INPUT: TIMELIMIT
3: FOUND=TRUE
4: $\zeta^{CURRENT}=\zeta$
5: while FOUND AND ELAPSEDTIME $\leq$ TIMELIMIT do
6: $\zeta^{N} \leftarrow f(\zeta^{CURRENT})$
7: if $z(\zeta^{N}) > z(\zeta^{CURRENT})$ then
8: $\zeta^{CURRENT} \leftarrow \zeta^{N}$
9: else
10: FOUND=FALSE
11: end if
12: end while
13: return $\zeta^{CURRENT}$

We encode solutions $\zeta$ to the original problem using

- a matrix $A(\zeta) \in \{0,1\}^{B \times |I|}$ defining the drop-off-fees. The $i$ - $j$th element of the matrix is an integer $l$ in $L$ identifying the drop-off-bye between zone $i$ and $j$.

Let $H(a,b)$ be the Hamming distance between two vectors or, in the case of matrices, concatenations of rows. We define two types of neighborhoods.

- $N^H(\zeta) = \{ \zeta' | H(I(\zeta),I(\zeta')) < 1, A(\zeta') = A(\zeta) \}$. In words, it defines all solutions which can be obtained from $\zeta$ by changing solely one vehicle position.

- $N^A(\zeta) = \{ \zeta' | I(\zeta) = I(\zeta'), H(A(\zeta),A(\zeta')) = 1 \}$. In words, it defines all solutions which can be obtained by changing solely one drop-off fee.

We define two first improvement operators:

- $f^H(\zeta) : \{0,1\}^{|I|} \times \{0,1\}^{B \times |I|} \rightarrow N^H(\zeta)$ scans the neighborhood $N^H(\zeta)$ and returns the first improving solution (i.e., with a higher fitness value) if it exists, $\zeta$ otherwise.

- $f^A(\zeta) : \{0,1\}^{|I|} \times \{0,1\}^{B \times |I|} \rightarrow N^A(\zeta)$ scans the neighborhood $N^A(\zeta)$ and returns the first improving solution (i.e., with a higher fitness value) if it exists, $\zeta$ otherwise.

Let $z(\zeta)$ be the fitness function defining how each solution is evaluated. It is defined as the objective function of the original problem (1a). For each solution $\zeta$ considered, the second-stage revenue is computed as illustrated in Section 4.2. Given a solution $\zeta$, we define two types of local search

- $LS^H(\zeta) : \{0,1\}^{|I|} \times \{0,1\}^{B \times |I|} \rightarrow N^H(\zeta)$ which performs a local search on the $N^H(\zeta)$ neighborhood using the $f^H(\zeta)$ first improvement operator.

- $LS^A(\zeta) : \{0,1\}^{|I|} \times \{0,1\}^{B \times |I|} \rightarrow N^A(\zeta)$ which performs a local search on the $N^A(\zeta)$ neighborhood using the $f^A(\zeta)$ first improvement operator.
Algorithm 4 sketches the local search procedures.

Finally, we define a function \( P(\zeta, R) : \{\mathbb{N}^{\mid \mathcal{V} \mid} \times \{\mathbb{N}^{\mid \mathcal{I} \mid} \} \rightarrow \{\mathbb{N}^{\mid \mathcal{V} \mid} \times \{\mathbb{N}^{\mid \mathcal{I} \mid} \} \} \) that randomly re-assigns \( \% \) of the positions and \( \% \) of the fees. The entire Iterated Local Search is sketched in Algorithm 5.

Algorithm 5 Iterated Local Search.

1: INPUT: \textsc{MAXRESTARTWITHOUTIMPROVEMENT}, \textsc{TIMELIMIT}, \textsc{R}
2: \#\textsc{RESTARTSWITHOUTIMPROVEMENT} ← 0
3: \textsc{\$BEST} ← \textsc{\$CURRENT}
4: \textbf{while} \textsc{\$CURRENT} ≤ \textsc{\$MAXRESTARTWITHOUTIMPROVEMENT} AND \textsc{ELAPSEDTIME} ≤ \textsc{\$TIMELIMIT} \textbf{do}
5: \textsc{\$CURRENT} ← \textsc{LS}(\textsc{\$CURRENT})
6: \textbf{if} \textsc{\$CURRENT} < \textsc{\$BEST} \textbf{then}
7: \textsc{\$BEST} ← \textsc{\$CURRENT}
8: \#\textsc{RESTARTSWITHOUTIMPROVEMENT} ← \#\textsc{RESTARTSWITHOUTIMPROVEMENT} + 1
9: \textbf{else}
10: \#\textsc{RESTARTSWITHOUTIMPROVEMENT} ← 0
11: \textbf{end if}
12: \textbf{if} \textsc{\$CURRENT} > \textsc{\$BEST} \textbf{then}
13: \textsc{\$BEST} ← \textsc{\$CURRENT}
14: \textbf{end if}
15: \textsc{\$CURRENT} ← \textsc{\$P}(\zeta, R)
16: \textbf{end while}
17: \textsc{return} \textsc{\$BEST}

In Algorithm 5 we set \textsc{MAXRESTARTWITHOUTIMPROVEMENT} to 3, \textsc{TIMELIMIT} to 1800 seconds, and \( R \) to \( 30 \% \).

Table 17 and 18 report the optimality gap and solution time of the ILS compared to that of the L-Shaped method on all instances with identical customer profiles. The optimality gap of the ILS is calculated using the bound delivered by the L-Shaped method as

\[
\text{\textsc{gap}} = \frac{|\textsc{\$OBJECTIVE} - \textsc{\$BESTBOUND}|}{|\textsc{\$OBJECTIVE}|} \times 10^{10}
\]

The tables show that, for the smallest instances, the performance of the ILS is comparable to that of the L-Shaped method. On a number of instances (e.g., with \( |\mathcal{V}| = 50 \)) the ILS even delivers better primal solutions than the L-Shaped method. Nevertheless, as the size of the instances increases the performance of the ILS drops.

References


