Zonification and Pricing in Carsharing: Formulations and Exact Solutions

Jiali Deng[a,*], Giovanni Pantuso[a]

[a]*Department of Mathematical Sciences, University of Copenhagen, Copenhagen, 2100, Denmark*

Abstract

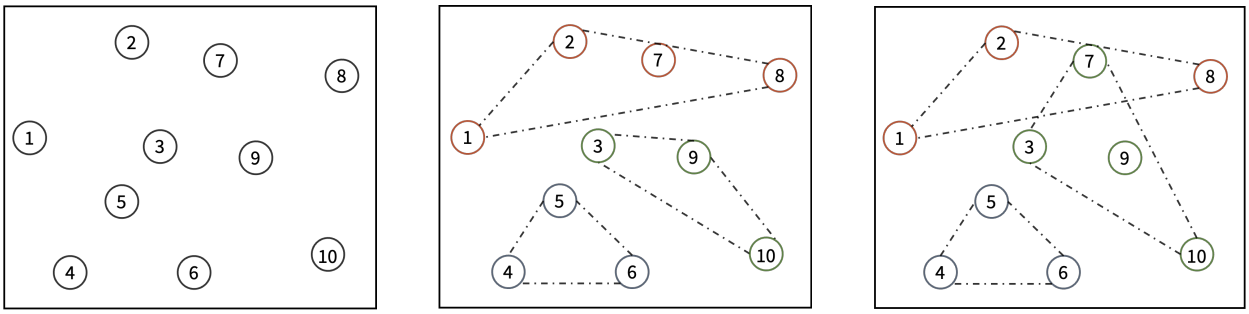
In this article we address the problem of partitioning a carsharing business area into pricing zones. We formalize the problem mathematically showing that it forms a special set partitioning problem. For this problem we propose a binary integer programming problem. Tighter reformulations can then be obtained by reformulating certain constraints emerged in the literature as closed assignment constraints. To solve the resulting mixed integer (possibly nonlinear) programming problem we develop an ad-hoc integer Benders decomposition for which we propose effective problem-specific improvements. Extensive tests based on a real-world carsharing system prove the method empirically effective on instances of size comparable to real-life problems. An analysis of the solutions shows that by jointly optimizing prices and pricing zones, the operator can observe a significant profit increase as well as a higher service rate, compared to currently employed zip-code-based partitions.

Keywords: transportation, one-way carsharing, pricing, zonification and tessellation, integer Benders decomposition

1. Introduction

Carsharing pricing decisions have attracted significant attention in the research literature, see e.g., Boyacı and Zografos (2019); Zhang et al. (2022); Huang et al. (2020); Soppert et al. (2022); Jorge et al. (2015); Pantuso (2022); Müller et al. (2023). They have been identified as a promising instrument to resolve fleet imbalances, see e.g., Illgen and Höck (2019), and improve profits and service rates. The corresponding literature has offered optimization models and methods where prices are considered either as a substitute or a supplement to vehicle rebalancing efforts. Prices are commonly differentiated geographically, see e.g., Jorge et al. (2015); Huang et al. (2020); Pantuso (2022), that is, they depend on the origin and/or destination of the rental. This typically implies that the business area is partitioned into distinct pricing zones that are independent of pricing decisions (Jorge et al., 2015; Boyacı and Zografos, 2019; Li et al., 2022) and provided a priori (Lu et al., 2021; Huang et al., 2020; Pantuso, 2022). See also real-world instances such as GreenMobility

*Corresponding Author
Email address: jd@math.ku.dk (Jiali Deng)



(a) Carsharing stations. (b) Acceptable partition. (c) Non-acceptable partition.

Figure 1: Carsharing stations, acceptable and non-acceptable partitions.

(2025), ShareNow (2025) and Free2move (2025). The decision of how to optimally divide a business area into pricing zones has not been investigated in detail.

In this paper, we focus on the problem of partitioning a set of carsharing stations into distinct pricing zones. We call this a *zonification* problem. The problem, which is motivated by an underlying industrial case, can be briefly described as follows. Consider a one-way station-based carsharing system and assume a given fixed set of stations, see e.g., Figure 1a. The goal of the service provider is to optimally adjust prices and pricing zones periodically during the day, and for small intervals of time (e.g., every hour), in order to adapt to changes in demand patterns. The prices may be differentiated by the origin and/or destination of the trip. This entails informing customers about the current prices from their zone to every other zone upon booking. Thus, for each time interval, the problem becomes that of partitioning the stations into pricing zones and then assigning prices to such zones. The underlying motivation is to ensure a suitable geographical distribution of the fleet as demand changes throughout the day. By periodically adjusting zones and prices, the provider improves their ability to better adapt the service to changes in demand and to influence demand. This would, in turn, help the company provide a suitable distribution of the fleet without excessive use of relocation activities which tend to be expensive and prone to inefficiency.

However, the definition of the zones must be done without compromising the ability of the company to communicate the pricing mechanism in an easy and intuitive manner via mobile applications. In particular, the resulting partition must be such that the zones created form individual "islands" or, in other words, they are "visually disjoint". The partition illustrated in Figure 1b would be acceptable as the zones form detached islands. However, the partition in Figure 1c would not be acceptable as the areas covered by the individual zones overlap. This particular requirement gives rise to a rich set partitioning problem. As we illustrate in Section 2, it shares similarities with tessellation and districting problems, though holding distinct characteristics.

In this paper, we give this special partitioning problem a precise mathematical interpretation and express it as a combinatorial optimization problem. We show that this problem can be formu-

lated as a binary integer programming problem. A special set of constraints in this formulation is known as *closest assignment constraints*. This allows us to obtain and evaluate different formulations for the problem. We then combine the zonification problem with that of setting zone-to-zone prices, hence obtaining a joint zonification and pricing problem. To address the computational complexity of the resulting optimization model, we propose an ad-hoc integer Benders decomposition for which we develop a number of effective improvements. Finally, we test the method on instances derived from a real-life carsharing system.

The contribution and main findings of this paper can be stated as follows.

- We formalize mathematically the problem of partitioning a set of carsharing stations into non-overlapping pricing zones. We begin with a rather abstract definition of the resulting partitioning problem and prove that solutions to this problem ensure non-overlapping zones. We then show that the set of partitions satisfying the given definition can be modeled using binary variables and linear constraints. Further, we show that certain constraints in our binary formulation allow tighter reformulations.

- We then augment the zonification problem to include pricing decisions. This extends the available carsharing literature by providing the first joint zonification and pricing problem. The resulting problem is formulated as a (possibly nonlinear) MIP problem. Linearity of the problem depends on the specification of the measure of performance of zonification and pricing decisions. The integrated zonification and pricing problem can be used to periodically adjust prices and pricing-zones during the day as demand patterns evolve.

- We provide a tailored integer Benders decomposition to find exact solutions to the problem. In addition, we develop a number of problem-specific improvements. Our choice of methodology was driven by the novelty of the problem at hand. We aimed to establish an exact method to ensure availability of optimal solutions. This method may, in turn, provide a useful benchmark potentially guiding future development of other, possibly heuristic, solution methods for the problem. In addition, the problem lends itself naturally to a Benders decomposition, given the hierarchy of decisions, where pricing and zonification decisions are followed by rental decisions.

- We provide empirical evidence on both the performance of the solution method and the practical effect of joint zonification and pricing decisions. This is achieved by solving instances generated from a real-world carsharing service based in Copenhagen. Our results show that the method outperforms by far a commercial solver. This establishes the applicability of the method in real-world practice. Furthermore, our results demonstrate that joint zonification and pricing decisions yield higher profits and service rates compared to a non-partitioned system and a pre-partitioned system using zip-codes.

The remainder of this paper is organized as follows. In Section 2, we review the relevant literature and give a more precise account of our contributions with respect to that. In Section 3, we formally introduce the zonification problem. In Section 4 we extend the zonification problem to account for pricing decisions yielding a (possibly nonlinear) MIP problem. In Section 5 we discuss reformulations of the problem and introduce efficient valid inequalities. In Section 6 we present a tailored Benders decomposition, introducing specific efficiency measures. In Section 7 we report on an extensive computational study. Particularly, we shed light both on the efficiency of the proposed decomposition method and on the practical effect of joint zonification and pricing decisions. Finally, we draw conclusions in Section 8.

2. Literature Review

In this section we summarize the relevant literature. We start by providing some insights into similar partitioning problems. Following, we summarize the relevant literature on pricing decisions in carsharing. In each subsection we conclude the relevance and main differences of our problem with respect to the available literature.

2.1. Spatial partitioning problems

The problem of partitioning the carsharing stations into pricing zones shares similarities with well studied problems. These include, in particular, tessellation problems (Okabe and Suzuki, 1997; Okabe et al., 2000) and districting problems (Duque et al., 2012; Kalcsics and Ríos-Mercado, 2019).

Tessellation problems can be summarized as follows. Given a finite set of distinct, isolated points in a continuous space, we associate all locations in that space with the closest member of the point set (Okabe et al., 2000). The result is a partition of a continuous space. This problem has widespread application and is often referred to with different aliases. The term *Voronoi diagram* appears to be one of the most frequent terms used. Voronoi diagram is a proximity-based partitioning method and provide a convex and compact partition of a continuous space. This method has been widely applied in various logistics districting problems, see e.g., Galvao et al. (2006); Novaes et al. (2009).

Similar to tessellation problems, we are concerned with partitioning a service region into disjoint areas. Nevertheless, there are at least two central differences between our problem and a tessellation problem. First, we operate on a discrete space. That is, we need to partition a discrete set of points in $\mathbb{R}^2$ into sub-regions. Second, Voronoi diagrams start from a given set of points (generators) and then assign the remaining points (typically in $\mathbb{R}^2$) to the closest generator. As it will be more clear in Section 3, in our problem the set of generators is not given. Choosing the optimal set of generators is thus a central task we need to address. Nevertheless, given the similarity between tessellation problems and our problem, we will refer to a solution of our problem as a *discrete tessellation*.

Districting problems are defined as partitioning all basic units in a geographic area into a number of geographic clusters, named districts. Almost all districting approaches require districts to be *contiguous* (Kalcsics and Ríos-Mercado, 2019). The definition of contiguity can vary among problems. If it is possible to identify, for every basic unit, an explicit neighborhood (e.g., all geographic locations sharing the same zip code or connected by a transportation link), then one can use a graph-based definition of contiguity. Particularly, given a graph that describes neighboring relationships, a district is contiguous if the basic units that form a district induce a connected subgraph. Districting problems are often formulated as MIP problems and there exists a wide range of applications. These include, logistics community partition (Jarrah and Bard, 2012; Carlsson and Delage, 2013), commercial territory design (Salazar-Aguilar et al., 2011; Bender et al., 2016), political districting (Bozkaya et al., 2003), electrical power districting (Bergey et al., 2003a,b). Within these applications, the contiguity criteria is usually satisfied by incorporating an exponential number of constraints (Hess et al., 1965) based on a neighbourhood graph representation, though polynomial formulations exist (Duque et al., 2012). In addition to contiguity, districts are often required to be *compact*. Intuitively, a district is said to be geographically compact if it is undistorted (preferably round-shaped) (Kalcsics and Ríos-Mercado, 2019).

There is no uniformly accepted definition of compactness (Mehrotra et al., 1998). Mathematically, we observe compactness is ensured by minimizing metrics such as the *moment of inertia* (Bard and Jarrah, 2009; Hess et al., 1965). This measure is quantified by taking the squared sum of the distances between the units in the district and center of the district, weighted by the population of the units. Another example is provided by Salazar-Aguilar et al. (2011) which minimizes the sum of distances between the units in the same district. However, in general, minimizing distance-based measures may fail to deliver a partition with non-overlapping districts (Kalcsics and Ríos-Mercado, 2019).

Similar to districting problems we are also concerned with partitioning a geographical set of locations in order to obtain somewhat compact subsets. In our case, the requirement of having visually disjoint districts (zones in our case), see Section 1, leads to our own specific interpretation of compactness. Contrary to the vast majority of districting problems, we are however prevented from ensuring compactness by minimizing a measure of compactness. In our problem a set of compact (according to the definition we will give in Section 3) districts or zones is the elementary building block of a pricing problem which pursues maximization of a measure of business performance. We will show that we are able to induce compact zones by means of binary variables and linear constraints.

2.2. Carsharing pricing

The pricing policies considered in the literature can be classified along several dimensions. One such classification is attempted by Pantuso (2022) who divides pricing policies as either *individual* or *collective*. Individual pricing policies are targeted to individual users and require an interaction

between the service provider and the individual user. In such interaction, the final trip price is agreed upon. Collective pricing policies are, instead, targeted to the entire user base. They have the scope of influencing the cumulative rental demand by differentiating prices geographically or across time. In this paper we focus on collective pricing policies.

Among the collective pricing policies, several devise spatially differentiated prices. In these methods, the price changes according to the origin and/or destination of the trip. These methods are, in general, aimed to hedge against potential geographical mismatch between supply and demand. Waserhole and Jost (2016) consider two origin-destination based pricing policies, namely static and dynamic. The former disregards changes occurring in the carsharing system and set prices based on average values. The latter adapts to the arrival of new information. Both pricing policies set prices according to individual pairs of carsharing stations. Ren et al. (2019) assume a fleet of shared electric vehicles and define prices that vary with time and with the origin and destination of the trip. Particularly, prices are set between pairs of carsharing stations. Their pricing mechanism also takes into account the profit obtained from the vehicle-to-grid interaction. Xu et al. (2018) study fleet sizing and trip pricing decisions for a station-based one-way carsharing system. They assume demand is elastic with respect to price and define prices that can vary according to the origin and destination station as well as the time and the specific user group. Angelopoulos et al. (2018) assume a pricing mechanism where price incentives are offered to users traveling from stations with a surplus of vehicle to stations with a shortage of vehicles. Zhang et al. (2022) consider a station-based carsharing service and assume the price depends on the origin and destination station. User may however receive a price incentive to pick up the car from a nearby station. A common feature in Waserhole and Jost (2016); Ren et al. (2019); Xu et al. (2018); Angelopoulos et al. (2018); Zhang et al. (2022) is that prices depend on the specific origin and destination station of the rental.

Huang et al. (2020) consider a station-based carsharing service and explore a pricing mechanism based both on the origin and destination of the rental. Additionally, their investigated pricing mechanism incorporates penalties for leaving and arriving at specific stations. The two penalties are independent on each other. Unlike in the previous studies, the authors assume that the business area is partitioned into zones, and prices depend on the origin and destination zone (or both). Pricing zones are also assumed in the work of Hansen and Pantuso (2018); Pantuso (2020), later extended by Pantuso (2022) and Eilertsen et al. (2024). They propose a pricing mechanism made of a per-minute fee and a drop-off fee. The former is independent while the latter is dependent on the origin and destination zone of the trip. Golalikhani et al. (2024) consider practical carsharing price plans. These involve both fixed costs (e.g., one-time registration fee) and variable costs that are related to travel distance and time. The one-way car-sharing system is divided into different areas according to their demand volume. They assume carsharing prices are dependent on the origin zone and the starting time of trips. Soppert et al. (2022) differentiate prices for a free-floating carsharing service according to the origin zone of the trip. Müller et al. (2023) consider a

customer-centric pricing mechanims where prices can vary according to the origin and time of the rental. They compare the pricing mechanisms to a number of benchmark location-based pricing mechanisms. Among these we find pricing mechanisms where the business area is partitioned into zones of $1Km \times 1Km$ and the way the price is set for a given zone gives rise to different pricing mechanisms. Li et al. (2022) consider a one-way station-based service. They assume prices depend on the origin and destination station of the rental. However, to reduce the dimension of the problem, they cluster carsharing stations into zones and assume that the same price applies to any station within the same zone. Zones are determined prior to solving the pricing problem. Boyacı and Zografos (2019) consider a one-way, station-based, carsharing service. The pricing mechanism entails offering incentives to users for flexibility in choosing the origin and destination station. Lu et al. (2021) consider a one-way station-based carsharing service. They study the problem of determining rental prices in a context of competition with private cars. Particularly, they determine the prices between any pair of zones for each time. Zones are input to the problem. Jorge et al. (2015) consider a one-way station-based system. They start by determining the zones of the city by clustering stations based on the similarity of their ideal demand and supply. This is, in turn, computed assuming relocation activities have cost zero. Following, they determine the prices which can vary with the origin and destination zone of the rental as well as with time.

In Table 1 we summarize the articles that consider a subdivision of the service area into pricing zones. In particular, for each article we report

- whether prices adapt to the origin or to the destination of the trip, or to both,
- whether the method proposed in the article focuses on pricing and/or relocation decisions,
- whether pricing zones are given as input or are decided in the method proposed, as well as the method used to generate the zones. Furthermore, we report whether pricing and zonification decisions are integrated.

A common characteristic to these studies is that they all assume that the business area is partitioned into zones. This is done with the purpose of determining either rental prices or relocations. Zones are typically given a priori (Huang et al., 2020; Hansen and Pantuso, 2018; Pantuso, 2020; Lu et al., 2021; Pantuso, 2022; Soppert et al., 2022; Müller et al., 2023; Eilertsen et al., 2024) based on, e.g., an existing real-world subdivision of the business area into districts. When zones are determined by the authors, the decision is always made prior to, and independently of, pricing or relocation decisions (Jorge et al., 2015; Boyacı and Zografos, 2019; Li et al., 2022). The method used to determine zones is typically a clustering algorithm.

We extend the existing literature by jointly addressing zonification and pricing decisions. As evident in Table 1, our work is the first to integrate these two critical aspects. Zonification decisions provide operators an additional lever to better tailor pricing decisions to specific, optimized, subregions of the business area. Predefined partitions may, in general, not serve this purpose as

well. Since demand patterns change during the day, a subdivision of the city which works well e.g., early in the morning, may not adequately capture customer movements in other phases of the day. This element emerged also in the context of ride-hailing services, where Shi et al. (2024) pointed out that fixed partition-based pricing may fail to accurately capture changes in supply and demand over time, and opted for a dynamic zone-based pricing strategy. Hence, by integrating pricing and zonification decisions, operators gain the ability to adjust services in response to changing demand patterns, leveraging the granularity of their business area representation.

Table 1: Comparative analysis on zonification methods in existing literature and our study. In the "Pricing strategy" column, we use (1), (2), and (3) to represent an origin- and destination-based, origin-based, and destination-based flexible prices, respectively.

Authors	Pricing strategy	Zone determination				Purpose	
		Input	Decision	Integrated	Method	Pricing	Relocation
Di Febbraro et al. (2012)	(3)	√			Real-world data	√	√
Jorge et al. (2015)	(1)		√		K-means	√	
Hansen and Pantuso (2018)	(1)	√			Evenly spread areas	√	
Boyacı and Zografos (2019)	(2), (3)		√		MIP model		√
Huang et al. (2020)	(1), (2), (3)	√			Real-world data	√	√
Pantuso (2020, 2022)	(1)	√			Evenly spread areas	√	√
Lu et al. (2021)	(1)	√			Real-world data	√	√
Soppert et al. (2022)	(2)	√			Real-world data	√	
Li et al. (2022)	(1)		√		K-means	√	√
Müller et al. (2023)	(2)	√			$1Km \times 1Km$ units	√	
Eilertsen et al. (2024)	(1)	√			Real-world data	√	√
Golalikhani et al. (2024)	(2)	√			Real-world data	√	
This paper	(1)		√	√	MIP model	√	

3. Zonification Problem

We start by introducing the problem in an abstract way. Later we show how this problem relates to that of defining pricing zones as discussed in Section 1.

Given a discrete metric space $(\mathcal{I}, d)$ we are concerned with the problem of finding a special partition of $\mathcal{I}$ whose characteristics are described in Definition 3.1.

Definition 3.1 (Discrete tessellation)**.** *Let $\mathcal{G} \subseteq \mathcal{I}$. A collection $\mathcal{V}(\mathcal{G}) \subset 2^{\mathcal{I}}$ of subsets of $\mathcal{I}$ is called the discrete tessellation of $\mathcal{I}$ induced by $\mathcal{G}$ iff the following properties hold:*

1. *(Disjunction) For every two sets $\mathcal{V}$,$\mathcal{U} \in \mathcal{V}(\mathcal{G})$ we have $\mathcal{V} \cap \mathcal{U} = \emptyset$*
2. *(Cover) $\bigcup \mathcal{V}(\mathcal{G}) = \mathcal{I}$*
3. *(One generator) For every $\mathcal{V} \in \mathcal{V}(\mathcal{G})$ we have $|\mathcal{V} \cap \mathcal{G}| = 1$*
4. *(Closest to generator) For every set $\mathcal{V} \in \mathcal{V}(\mathcal{G})$ let $c \in \mathcal{V}$ be the element such that $\{c\} = \mathcal{V} \cap \mathcal{G}$. Then, for every element $v \in \mathcal{V}$ of the set we have that $d(v, c) \leq d(v, k)$ for all $k \in \mathcal{G}$.*

Properties 1 and 2 define a partition of $\mathcal{I}$ (i.e., a disjoint cover). Property 3 ensures that each set in the partition contains exactly one element of the set $\mathcal{G}$. The elements of $\mathcal{G}$ are thus

understood as the *generators* of the tessellation. Finally, property 4 characterizes the partition as a tessellation. It states that each point in $\mathcal{I}$ is assigned to the subset that contains its closest generator from $\mathcal{G}$ in the sense of the metric d. Thus, each set of the partition $\mathcal{V}(\mathcal{G})$ contains the points of $\mathcal{I}$ that are closest to the single element of $\mathcal{G}$ in $\mathcal{V}$ than to any other element of $\mathcal{G}$ in the sense of the metric.

We are particularly concerned with the problem of finding an optimal tessellation according to some measure of performance. That is

$$\max_{\mathcal{G}\in 2^{\mathcal{I}}} \big\{ R\big(\mathcal{V}(\mathcal{G})\big) \mid \text{Properties (1)-(4) hold} \big\} \tag{1}$$

where R is a mapping from the set of all partitions of $\mathcal{I}$ to the real numbers. Thus, the problem can be seen as that of finding generators $\mathcal{G} \subseteq \mathcal{I}$ whose induced tessellation maximizes R. Of particular interest are the problems where we look for a partition made of exactly S subsets. That is, where we restrict the maximization over the set $\mathcal{G}^S := \{\mathcal{G} \in 2^{\mathcal{I}} || \mathcal{G}| = S\}$.

The abstract problem of finding an optimal discrete tessellation finds a concrete application in the carsharing zonification problem as sketched in Section 1. This relates the zonification problem specifically to the discrete tessellation as defined in Definition 3.1. In the following, we may use the terms "discrete tessellation" and "zonification" interchangeably. The partition of the carsharing stations must be such that the subsets are "visually disjoint", see Figure 1b. We interpret this requirement mathematically as a partition that generates mutually non-overlapping convex hulls. Proposition 3.1 shows that, by working with partitions of the type introduced in Definition 3.1, we automatically satisfy this property.

Proposition 3.1. *Let $\mathcal{I}$ be a finite subset of $\mathbb{R}^n$. Let $\mathcal{V}(\mathcal{G}) \subset 2^{\mathcal{I}}$ be the discrete tessellation of $\mathcal{I}$ induced by $\mathcal{G} \subseteq \mathcal{I}$ as defined in Definition 3.1. Then, for all sets $\mathcal{V}$,$\mathcal{U} \in \mathcal{V}(\mathcal{G})$ we have*

$$iConv(\mathcal{V}) \cap iConv(\mathcal{U}) = \emptyset$$

where $iConv(\mathcal{V})$ (resp. $iConv(\mathcal{U})$) is the interior of the convex hull $Conv(\mathcal{V})$ (resp. $Conv(\mathcal{U})$) of the points in the set $\mathcal{V}$ (resp. $\mathcal{U}$).

Proof. We prove Proposition 3.1 by contradiction. Assume $\mathcal{V}(\mathcal{G})$ is a discrete tessellation and that $iConv(\mathcal{V}) \cap iConv(\mathcal{U}) \neq \emptyset$ for some $\mathcal{V}$ and $\mathcal{U} \in \mathcal{V}(\mathcal{G})$. Then, there exists either some $v \in \mathcal{V}$ in $iConv(\mathcal{U})$ or some $u \in \mathcal{U}$ in $iConv(\mathcal{V})$. Assume, without loss of generality, that $v \in iConv(\mathcal{U})$. Then, we also have that $v \in Conv(\mathcal{U})$ since $iConv(\mathcal{U}) \subset Conv(\mathcal{U})$. Furthermore, since $iConv(\mathcal{U})$ is open, there exists $r > 0$ such that $\mathcal{B}(v, r) = \{u' \in \mathbb{R}^n | d(u', v) < r\} \subseteq iConv(\mathcal{U})$. Consequently, there exists some $u' \in \mathcal{B}(v, r)$ (which may or may not be in $\mathcal{U}$) such that

$$d(v, g^{\mathcal{U}}) \leq d(u', g^{\mathcal{U}})$$

where $g^{\mathcal{U}} \in \mathcal{G}$ is the generator of $\mathcal{U}$.

Since $iConv(\mathcal{U}) \subset Conv(\mathcal{U})$ then we also have that $u' \in Conv(\mathcal{U})$. Thus, there exist $\alpha_i \geq 0$ for $i = 1, \ldots, N$, with $\sum_{i=1}^{N} \alpha_i = 1$, such that $u' = \sum_{i=1}^{N} \alpha_i u_i$ where $u_i \in \mathcal{U}$ for $i = 1, \ldots, N$. Then we have

$$d(v, g^{\mathcal{U}}) \leq d(\sum_{i=1}^{N} \alpha_i u_i, g^{\mathcal{U}})$$

which is only possible if there exists some $\alpha_i > 0$ and u_i such that

$$d(v, g^{\mathcal{U}}) \leq d(u_i, g^{\mathcal{U}})$$

However, since $d(u_i, g^{\mathcal{U}}) \leq d(u_i, g)$ for all $g \in \mathcal{G}$, it follows that $d(v, g^{\mathcal{U}}) \leq d(v, g)$ for all $g \in \mathcal{G}$. This violates property 4 in Definition 3.1, and contradicts that $\mathcal{V}(\mathcal{G})$ is a discrete tessellation. Hence, we must have that $iConv(\mathcal{V}) \cap iConv(\mathcal{U}) = \emptyset$. □

Proposition 3.1 clarifies that solving problems of type (1) automatically ensures that the resulting partition generates convex hulls with disjoint interiors as required. The convex hulls may however intersect on the boundary, which is acceptable. Observe that the proof of Proposition 3.1 works only in one direction. Hence, there may exist partitions which satisfy the claim but do not agree with Definition 3.1.

We proceed by showing that a discrete tessellation can be enforced by integer linear programming constraints. We introduce binary variables $a \in \{0,1\}^{|\mathcal{I}|\times|\mathcal{I}|}$. Variable a_{ii} takes value 1 if $i \in \mathcal{I}$ is designated as a generator, while variable a_{ij} takes value 1 if element j is a generator and element i belongs to the same subset as i. The set of all feasible discrete tessellations made of exactly S subsets can be expressed using $\mathcal{O}(|\mathcal{I}|^3)$ linear constraints as follows:

$$\mathcal{T} := \left\{ a \in \{0,1\}^{|\mathcal{I}|\times|\mathcal{I}|} \;\middle|\; \begin{array}{cc} \sum_{i\in\mathcal{I}} a_{ii} = S & \\ \sum_{j\in\mathcal{I}} a_{ij} = 1 & \forall i \in \mathcal{I} \\ a_{ij} \leq a_{jj}, & \forall i,j \in \mathcal{I} \\ d(i,j_1)a_{i,j_1} \leq d(i,j_2)a_{j_2,j_2} + d(i,j_1)(1-a_{j_2,j_2}) & \forall i,j_1,j_2 \in \mathcal{I} \end{array} \right\}$$

The constraints that define $\mathcal{T}$ ensure that (1) we choose exactly S points as generators, each representing a subset, (2) each point $i \in \mathcal{I}$ is assigned to exactly one subset (identified by a generator), (3) a point $i \in \mathcal{I}$ can be assigned to a subset represented by point $j \in \mathcal{I}$ only when j is chosen as a generator, and (4) given two subsets (represented by points j_1 and j_2), a node i is always assigned to the generator that is no further than the other.

Then, the set of all feasible discrete tessellations is obtained as follows. For every $\bar{a} \in \mathcal{T}$ we have $\mathcal{G}(\bar{a}) = \{i \in \mathcal{I} : \bar{a}_{ii} = 1\}$ and

$$\mathcal{V}(\mathcal{G}(\bar{a})) = \left\{\mathcal{V}_i = \{i\} \cup \{j \in \mathcal{I} | \bar{a}_{ij} = 1\} \; \forall i \in \mathcal{G}(\bar{a})\right\}$$

Thus, we can rewrite problem (1) as the following integer program

$$\max_{a \in \mathcal{T}} R(a) \tag{2}$$

where (with a slight abuse of notation) $R : \mathcal{T} \to \mathbb{R}$ is defined as $R(a) := R(\mathcal{V}(\mathcal{G}(a)))$.

4. Pricing Problem

In this section, discrete tessellations of the type introduced in Section 3 are used to generate pricing zones for a carsharing service. The pricing problem can be thus summarized as that of finding a discrete tessellation of the carsharing stations and assigning prices between any pair of subsets (henceforth zones). The measure of performance $R(\mathcal{V}(\mathcal{G}))$ is now understood as a measure of business performance (e.g., profits) generated by the rentals occurred as a consequence of the defined zones and prices.

We assume that the service provider may choose the price to apply between any pair of zones from a discrete set. This is consistent, for example, with the mechanism proposed by Soppert et al. (2022); Müller et al. (2023) where the per-minute price is selected from a finite set of prices, or by Pantuso (2022) where a drop-off fee is selected from a finite set of prices. We denote the set of price levels by $\mathcal{L}$. Binary variables λ_{ijl} take value 1 if price $l \in \mathcal{L}$ is applied between the zones generated by $i \in \mathcal{I}$ and $j \in \mathcal{I}$ (if i and j are chosen as generators), 0 otherwise. Following, we define decision variables α_{ijl} to take value 1 if price level l is adopted between stations $i \in \mathcal{I}$ and $j \in \mathcal{I}$, 0 otherwise. Let $\alpha := (\alpha_{ijl})_{i,j \in \mathcal{I}, l \in \mathcal{L}}$ and $\lambda := (\lambda_{ijl})_{i,j \in \mathcal{I}, l \in \mathcal{L}}$. The pricing problem can thus be expressed as follows:

$$\max\ Q(a, \lambda, \alpha) \tag{3a}$$

$$\text{s.t.} \quad \sum_{l \in \mathcal{L}} \lambda_{ijl} \geq a_{ii} + a_{jj} - 1, \qquad \forall i, j \in \mathcal{I} \tag{3b}$$

$$\sum_{l \in \mathcal{L}} \lambda_{ijl} \leq a_{ii}, \qquad \forall i, j \in \mathcal{I} \tag{3c}$$

$$\sum_{l \in \mathcal{L}} \lambda_{ijl} \leq a_{jj}, \qquad \forall i, j \in \mathcal{I} \tag{3d}$$

$$a_{i_1,j_1} + a_{i_2,j_2} + \lambda_{j_1,j_2,l} \leq \alpha_{i_1,i_2,l} + 2, \qquad \forall i_1, i_2, j_1, j_2 \in \mathcal{I}, \forall l \in \mathcal{L} \tag{3e}$$

$$\sum_{l \in \mathcal{L}} \alpha_{ijl} = 1, \qquad \forall i, j \in \mathcal{I} \tag{3f}$$

$$a \in \mathcal{T} \tag{3g}$$

$$\lambda, \alpha \in \{0, 1\}^{|\mathcal{I}| \times |\mathcal{I}| \times |\mathcal{L}|} \tag{3h}$$

The function $Q(a, \lambda, \alpha)$ represents the performance (e.g., profit or service rate) obtained by the rentals occurred as a consequence of the prices set between each pair of stations. This function is

general and can adapt to the specific configuration of the carsharing service. A possible specification will be provided in Section 7.1. Constraints (3b) state that if both i and j are designated as generators of a zone, then a price level must be assigned between the zones they generate. Constraints (3c) and (3d) ensure that at most one price level is chosen between any pair of zones, and no price level is chosen if either i or j are not designated as generators. Constraints (3e) ensure that, if station i_1 is assigned to zone j_1 and station i_2 is assigned to zone j_2, then the price level between zones j_1 and j_2 applies to stations i_1 and i_2. Constraints (3f) ensure that exactly one price level is applied between each pair of stations. Finally, constraints (3g) ensure that the a variables define a discrete tessellation. Observe that problem (3) is, in general, a nonlinear MIP problem.

5. Reformulations and Valid Inequalities

We begin by improving the formulation of $\mathcal{T}$ by rewriting the constraints and identifying redundant ones. In particular, consider constraints

$$d(i,j_1)a_{i,j_1} \leq d(i,j_2)a_{j_2,j_2} + d(i,j_1)(1-a_{j_2,j_2}) \quad \forall i,j_1,j_2 \in \mathcal{I}$$

We can scale down both sides by $d(i,j_1)$. That is, for each $i,j_1,j_2 \in \mathcal{I}$

$$d(i,j_1)a_{i,j_1} \leq d(i,j_2)a_{j_2,j_2} + d(i,j_1)(1-a_{j_2,j_2}) \iff a_{i,j_1} \leq \left(\frac{d(i,j_2)}{d(i,j_1)} - 1\right)a_{j_2,j_2} + 1 \qquad (4)$$

Observe that, when $d(i,j_1) \leq d(i,j_2)$, the right-hand side of constraints (4) is always larger than 1, making constraints (4) redundant. When $d(i,j_1) > d(i,j_2)$, the coefficient of a_{j_2,j_2} can be rounded down to -1. Therefore, we obtain the following reformulation of $\mathcal{T}$, that is

$$\mathcal{T} := \left\{ a \in \{0,1\}^{|\mathcal{I}|\times|\mathcal{I}|} \;\middle|\; \begin{array}{cc} \sum_{i\in\mathcal{I}} a_{ii} = S & \\ \sum_{j\in\mathcal{I}} a_{ij} = 1 & \forall i \in \mathcal{I} \\ a_{ij} \leq a_{jj}, & \forall i,j \in \mathcal{I} \\ a_{i,j_1} \leq 1 - a_{j_2,j_2}, & \forall i,j_1,j_2 \in \mathcal{I} : d(i,j_1) > d(i,j_2) \end{array} \right\}$$

Constraints $a_{i,j_1} \leq 1 - a_{j_2,j_2} \forall i,j_1,j_2 \in \mathcal{I} : d(i,j_1) > d(i,j_2)$ have appeared in the literature under the name of *closest assignment constraints* in facility location problems. Various reformulations of these constraints have been proposed. The version proposed above was introduced by Dobson and Karmarkar (1987). Particularly, the reformulations proposed by Wagner and Falkson (1975) and Cánovas et al. (2007) are of interest in our context. They are given in (WF) and (CGLM), respectively.

$$\sum_{k:d(i,k)>d(i,j)} a_{i,k} + a_{j,j} \leq 1 \qquad \forall i,j \in \mathcal{I} \qquad \text{(WF)}$$

$$\sum_{k:d(i_1,k)>d(i_1,j)} a_{i_1,k} + \sum_{k:d(i_2,k)>d(i_2,j),d(i_1,k)\leq d(i_1,j)} a_{i_2,k} + a_{j,j} \leq 1 \qquad \forall i_1, i_2, j \in \mathcal{I} \qquad \text{(CGLM)}$$

Constraints (WF) state that station i can be assigned to a generator k further away than j only if j is closed (i.e., $a_{j,j} = 0$). Constraints (CGLM) state that when j is not chosen as the generator, once one of the stations, say i_1, has be assigned to a certain generator k that is further away than j, then i_2 cannot be assigned to a generator that is closer to i_1 than node j. We introduce $\mathcal{T}^{WF}$ as the set obtained by replacing the Dobson and Karmarkar (1987)constraints by the (WF) constraints in $\mathcal{T}$. In a similar manner we obtain $\mathcal{T}^{CGLM}$, respectively.

It is easy to see that constraints (WF) and (CGLM) *dominate* the Dobson and Karmarkar (1987) constraints. These results are summarized and proven in Espejo et al. (2012). The relationship we obtain is the following

$$LP(\mathcal{T}^{CGLM}) \subseteq LP(\mathcal{T}^{WF}) \subseteq LP(\mathcal{T})$$

where $LP(\cdot)$ is the set obtained by relaxing integrality restrictions. Finally, observe that the size of $\mathcal{T}^{CGLM}$ and $\mathcal{T}$ is $\mathcal{O}(|\mathcal{I}|^3)$ while the cardinality of $\mathcal{T}^{WF}$ is only $\mathcal{O}(|\mathcal{I}|^2)$. The empirical effect of these results will be further investigated in our experiments.

5.1. Valid inequalities

Given a station i and a generator j define the set $\mathcal{I}^B_{ij} = \{k \in \mathcal{I} | d(i,k) < d(i,j)\}$ to collect the subset of generators that are closer to station i than generator j. The following result from Cánovas et al. (2007) introduces the first valid inequality.

Proposition 5.1 (Cánovas et al. (2007))**.** *Let $i_1, i_2, j \in \mathcal{I}$. If $\mathcal{I}^B_{i_2 j} \subseteq \mathcal{I}^B_{i_1 j}$, then it holds that*

$$a_{i_1 j} \leq a_{i_2 j} \qquad \text{(VI-1)}$$

Proof. Assume (VI-1) violated, hence $a_{i_1 j} = 1$ and $a_{i_2 j} = 0$. We then have $a_{jj} = 1$, and $a_{kk} = 0, \forall\ k \in \mathcal{I}^B_{i_1 j}$. In turn $a_{jj} = 1$ and $a_{i_2 j} = 0$ implies that there exists one generator $k' \in \mathcal{I}^B_{i_2 j}$ such that $a_{k'k'} = 1$. However, since $k' \in \mathcal{I}^B_{i_2 j} \subseteq \mathcal{I}^B_{i_1 j}$ and $a_{kk} = 0, \forall\ k \in \mathcal{I}^B_{i_1 j}$, this leads to a contradiction and proves the statement in Proposition 5.1. □

Proposition 5.1 gives rise to $\mathcal{O}(|\mathcal{I}|^3)$ possible valid inequalities.

Additional valid inequalities can be obtained by examining the distance between each pair of stations. Particularly, for each $i \in \mathcal{I}$, let $d(i, j_i^{(1)}) \leq d(i, j_i^{(2)}) \leq \cdots \leq d(i, j_i^{(|\mathcal{I}|-1)})$ so that $j_i^{(n)} \in \mathcal{I}$ with $n = 1, \ldots, |\mathcal{I}| - 1$ is the index of the n-th furthest station from i (assume ties are broken arbitrarily). Define, for all $i \in \mathcal{I}$, the set

$$\mathcal{J}_i := \left\{ j_i^{(|\mathcal{I}|-S+1)}, \ldots, j_i^{(|\mathcal{I}|-1)} \right\}$$

Hence, $\mathcal{J}_i$ contains the indices of the $S-1$ stations further from i.

Proposition 5.2. *Let $a \in \mathcal{T}$ and $\mathcal{J}_i^C$ be the complement of $\mathcal{J}_i$. Then $a_{ij} = 1$ for some $j \in \mathcal{J}_i^C$.*

Proof. Assume $a_{ij} = 0$ for all $j \in \mathcal{J}_i^C$. Hence $a_{ij} = 1$ for some $j \in \mathcal{J}_i$. Since $|\mathcal{J}_i| = S - 1$ there exists some $\bar{j} \in \mathcal{J}_i^C$ with $a_{\bar{j}\bar{j}} = 1$. By definition of $\mathcal{J}_i$ we have that $d(i, \bar{j}) \leq d(i, j)$ for all $j \in \mathcal{J}_i$. Hence, solution $a \notin \mathcal{T}$. □

Hence it is possible to set

$$a_{ij} = 0 \qquad \forall i \in \mathcal{I}, j \in \mathcal{J}_i \tag{VI-2}$$

in practice removing all variables a_{ij} for $j \in \mathcal{J}_i$.

6. Tailored Integer Benders Decomposition

We propose a tailored integer Benders decomposition to obtain exact solutions to problem (3). The method is based on the following assumption.

A1 $Q(a, \lambda, \alpha)$ can be computed for all a, λ and α.

Observe that we do not make any restriction regarding the functional form of $Q(a, \lambda, \alpha)$. The method is thus general in the specification of the performance measure. In our experiments in Section 7 we compute $Q(a, \lambda, \alpha)$ by solving a MILP problem.

We start by reformulating problem (3) as follows:

$$\max_{a \in \mathcal{T}, \lambda, \alpha \in \{0,1\}^{|\mathcal{I}| \times |\mathcal{I}| \times |\mathcal{L}|}, \phi} \{\phi | \phi \leq Q(a, \lambda, \alpha), (3b) - (3f)\} \tag{5}$$

This, in turn, allows us to relax constraints $\phi \leq Q(a, \lambda, \alpha)$ and work with the following *relaxed master problem* (RMP):

$$\max_{a \in \mathcal{T}, \lambda, \alpha \in \{0,1\}^{|\mathcal{I}| \times |\mathcal{I}| \times |\mathcal{L}|}, \phi} \{\phi | (3b) - (3f)\} \tag{RMP}$$

In (RMP), ϕ overestimates the value of $Q(a, \lambda, \alpha)$. The overestimation is corrected by means of the addition of optimality cuts. We devise integer optimality cuts of the type introduced in Laporte and Louveaux (1993). Let U be a constant such that

$$\infty > U \geq \max_{a \in \mathcal{T}, \lambda, \alpha \in \{0,1\}^{|\mathcal{I}| \times |\mathcal{I}| \times |\mathcal{L}|}} Q(a, \lambda, \alpha)$$

Given the k-th feasible solution $(a^k, \lambda^k, \alpha^k)$, let $\mathcal{A}_k^+ \subseteq \mathcal{I} \times \mathcal{I}$ and $\mathcal{A}_k^- \subseteq \mathcal{I} \times \mathcal{I}$ be the sets containing the (i, j) pairs such that $a_{ij}^k = 1$ and $a_{ij}^k = 0$, respectively. Similarly, let $\Lambda_k^+ \subseteq \mathcal{I} \times \mathcal{I} \times \mathcal{L}$ and $\Lambda_k^- \subseteq \mathcal{I} \times \mathcal{I} \times \mathcal{L}$ be the sets of tuples (i, j, l) for which $\lambda_{ijl}^k = 1$ and $\lambda_{ijl}^k = 0$, respectively. Finally,

let $\Delta_k^+ \subseteq \mathcal{I}\times\mathcal{I}\times\mathcal{L}$ and $\Delta_k^- \subseteq \mathcal{I}\times\mathcal{I}\times\mathcal{L}$ be the sets of tuples (i,j,l) for which $\alpha_{ijl}^k = 1$ and $\alpha_{ijl}^k = 0$, respectively. Proposition 6.1 defines the optimality cuts.

Proposition 6.1. *Let $(a^k, \lambda^k, \alpha^k)$ be the k-th feasible solution to constraints* (3b) - (3h)*. The set of cuts*

$$\begin{aligned}\phi \leq \big(Q(a^k,\lambda^k,\alpha^k) - U\big)\Bigg(&\sum_{(i,j)\in\mathcal{A}_k^+} a_{ij} - \sum_{(i,j)\in\mathcal{A}_k^-} a_{ij} + \sum_{(i,j,l)\in\Lambda_k^+} \lambda_{ijl} - \sum_{(i,j,l)\in\Lambda_k^-} \lambda_{ijl} \\ + \sum_{(i,j,l)\in\Delta_k^+} \alpha_{ijl} - &\sum_{(i,j,l)\in\Delta_k^-} \alpha_{ijl}\Bigg) - \big(Q(a^k,\lambda^k,\alpha^k) - U\big)\big(|\mathcal{A}_k^+| + |\Lambda_k^+| + |\Delta_k^+| - 1\big) + U\end{aligned} \tag{6}$$

defined for all feasible $(a^k, \lambda^k, \alpha^k)$ solutions to (3b) - (3h) *makes* (RMP) *a reformulation of* (5)*.*

Proof. Observe that, since a, λ and α are binary, the number of feasible solutions to (3b) - (3h) is finite, and so is the number of cuts (6). Then, it is easy to see that when (a, λ, α) is the k-th feasible solution the quantity

$$\Bigg(\sum_{(i,j)\in\mathcal{A}_k^+} a_{ij} - \sum_{(i,j)\in\mathcal{A}_k^-} a_{ij} + \sum_{(i,j,l)\in\Lambda_k^+} \lambda_{ijl} - \sum_{(i,j,l)\in\Lambda_k^-} \lambda_{ijl} + \sum_{(i,j,l)\in\Delta_k^+} \alpha_{ijl} - \sum_{(i,j,l)\in\Delta_k^-} \alpha_{ijl}\Bigg)$$

reduces to

$$|\mathcal{A}_k^+| + |\Lambda_k^+| + |\Delta_k^+|$$

and (6) reduces to

$$\phi \leq Q(a^k, \lambda^k, \alpha^k)$$

Otherwise it reduces to a generally valid upper bound larger than U. □

Thus, optimality cuts (6) are violated by (and thus cut off) solutions $(a^k, \lambda^k, \alpha^k, \phi^k)$ for which $\phi > Q(a^k, \lambda^k, \alpha^k)$. They are however redundant, and thus safe, for solutions other than $(a^k, \lambda^k, \alpha^k, \phi)$ for any ϕ. Convergence of the algorithm is then granted by the existence of only finitely many such cuts.

6.1. Valid inequalities for integer Benders decomposition

For improving the computational efficiency of the integer Benders decomposition method developed, additional valid inequalities are constructed using the upper-bounds on the performance measure Q.

Such performance upper bounds can be evaluated on a zonification basis, i.e., for each possible set of generators $\mathcal{G} \in \mathcal{G}^S$. Let $\overline{P}_\mathcal{G}$ represent the performance upper bound for the tessellation induced by $\mathcal{G}$, i.e., when $a_{ii} = 1$ for all $i \in \mathcal{G}$. To compute $\overline{P}_\mathcal{G}$ observe that, given $\mathcal{G}$, the allocation

of stations to zones (i.e., a specification of a) can be found in $\mathcal{O}(|\mathcal{I}|^2)$ operations which assign each station to the closest generator $i \in \mathcal{G}$. Given which, the following valid inequalities can be added.

$$\phi \leq \overline{P}_{\mathcal{G}} + \left(U - \overline{P}_{\mathcal{G}}\right)\left(S - \sum_{j \in \mathcal{G}} a_{jj}\right) \quad \forall \mathcal{G} \in \mathcal{G}^S \tag{7}$$

Observe that, when a given $\mathcal{G}$ of generators is enforced we have $\sum_{j \in \mathcal{G}} a_{jj} = S$. Thus ϕ is bounded by the corresponding $\overline{P}_{\mathcal{G}}$. When $\mathcal{G}$ is not enforced we have $\sum_{j \in \mathcal{G}} a_{jj} < S$ and the left-hand side of (7) reduces to a value that is greater than U. Finally, observe that it is possible to generate up to $\binom{|\mathcal{I}|}{S}$ valid inequalities (7).

One can also evaluate the performance upper bounds on the customer O-D basis, i.e., for each station pair (i, j). Let $\overline{P}_{ijl}$ indicate the performance upper bound for station O-D pair (i, j) at pricing level l. In practice, $\overline{P}_{ijl}$ can be easily obtained through multiplying the maximal serviceable demand from station i to j by the per-customer return (e.g., profit) at pricing level l. These upper bounds can be computed in $\mathcal{O}(|\mathcal{I}|^2|\mathcal{L}|)$ operations. Based on which, another valid inequality can be added as follows.

$$\phi \leq \sum_{i,j \in \mathcal{I}} \sum_{l \in \mathcal{L}} \alpha_{ijl} \overline{P}_{ijl} \tag{8}$$

Observe that only one pricing level is applied to each station pair (i, j), i.e., $\sum_{l \in \mathcal{L}} \alpha_{ijl} = 1$. Consequently, the right-hand side of (8) ends up with the accumulation of the performance upper bounds for all station pairs at their respective pricing levels induced by a pricing decision α.

In Section 7.2 we explain how upper bounds $\overline{P}_{\mathcal{G}}$ and $\overline{P}_{ijl}$ can be computed for our specification of $Q(a, \lambda, \alpha)$.

6.2. Outline of the algorithm

The solution procedure is described in Algorithm 1. Observe that (RMP) is a MILP. For this we solve it in a branch-and-bound (B&B) procedure. Optimality cuts are separated only upon finding nodes in the tree with an integer solution, see Laporte and Louveaux (1993). This strategy is generally preferable to building a new tree each time (RMP) is solved.

Algorithm 1 is essentially a B&B procedure with the addition of optimality cuts added every time an integer feasible solution is found. The algorithm starts from selecting a pendant node and solves the linear programming (LP) relaxation of (RMP) corresponding to that node (i.e., with the addition of all branching constraints). If the LP relaxation to (RMP) is infeasible or it returns an objective value not larger than `best_objective`, the current node node is pruned (see, line 9 and line 13). If the solution returned by the LP relaxation of (RMP) is fractional, two branches are created and added to the set $\mathcal{P}$. Otherwise (i.e., the solution is integer), the algorithm identifies any violated optimality cuts (see, from line 20). The relaxation of (RMP) for the same node is then solved again.

Compared to an ordinary B&B method, nodes are not pruned by integrality. On the contrary, integrality triggers the separation of optimality cuts, see Algorithm 1 at line 20. Furthermore, the algorithm may return to the same node after adding optimality cuts, see Line 23. The algorithm leaves a node either when the corresponding (RMP) (i) is infeasible, (ii) provides a bound worse than the incumbent, (iii) returns a fractional solution or (iv) provides an integer solution which satisfies all optimality cuts.

The algorithm terminates when either there is no pendant node in the B&B tree, or when the optimality gap is within the required tolerance `TOL`. The optimality gap is computed as $|$`best_bound-objective_value`$|/(\epsilon + |$`objective_value`$|)$ where ϵ is a sufficiently small constant, e.g., 10^{-10}.

Algorithm 1 Pseudocode of the algorithm for solving problem (3).

1: `Input TOL`: optimality tolerance
2: `Initialize` $k \leftarrow 0$, `best_bound` $\leftarrow \infty$, `best_objective` $\leftarrow -\infty$
3: $\mathcal{P} \leftarrow \{P^0\}$ ▷ $\mathcal{P}$ is the set of open nodes, P^0 is to the initial (RMP).
4: **while** $\mathcal{P} \neq \emptyset$ or ($|$`best_bound-objective_value`$|/(\epsilon + |$`objective_value`$|) >$ `TOL` **do**
5: Select a pendant node from set $\mathcal{P}$
6: $k \leftarrow k + 1$
7: Solve the linear programming relaxation of (RMP)
8: **if** (RMP) is infeasible **then**
9: Remove the current node from $\mathcal{P}$ and return to Line 5 ▷ Prune by infeasibility.
10: **else**
11: Let $(a^k, \lambda^k, \alpha^k, \phi^k)$ be the optimal solution to (RMP)
12: **if** $\phi^k \leq$ `best_objective` **then**
13: Remove the current node from $\mathcal{P}$ and return to Line 5 ▷ Prune by bound.
14: **else**
15: `best_bound` $\leftarrow \min\{$`best_bound`$, \phi^k\}$ ▷ Update the global upper bound.
16: **if** the solution $(a^k, \lambda^k, \alpha^k)$ is not integer **then**
17: Branch on a fractional variable, produce two new nodes and add them to $\mathcal{P}$
18: Remove the current node from $\mathcal{P}$ and return to Line 5
19: **else**
20: Compute $Q(a^k, \lambda^k, \alpha^k)$
21: `best_objective` $\leftarrow \max\{Q(a^k, \lambda^k, \alpha^k),$ `best_objective`$\}$ ▷ Update incumbent.
22: **if** $\phi^k > Q(a^k, \lambda^k, \alpha^k)$ **then** ▷ Optimality condition is violated.
23: Add an optimality cut of form (6) to (RMP) and return to Line 6
24: **else**
25: Remove the current node from $\mathcal{P}$ and return to Line 5 ▷ Prune by optimality.
26: **end if**
27: **end if**
28: **end if**
29: **end if**
30: **end while**

7. Numerical Experiments

We perform experiments on instances based on a real carsharing service in the city of Copenhagen, Denmark. The scope of the experiments is twofold. First, we provide empirical evidence on the impact of the different formulations and valid inequalities described in Section 5 and on the performance of the the decomposition method described in Section 6. Second, we discuss the practical effect and potential benefits of joint zonification and pricing decisions.

In the remainder of this section, we begin with Section 7.1 by explaining how we model $Q(a, \lambda, \alpha)$, the rental profits occurred as a consequence of zonification and pricing decisions. In

Section 7.2 we provide additional efficiency measures for accelerating the integer Benders decomposition based on the specific model of $Q(a, \lambda, \alpha)$. Following, in section 7.3 we introduce the instances we solve. Finally, in Section 7.4 we provide empirical evidence on the performance of the solution method and in Section 7.5 we discuss the impact of joint zonification and pricing decisions.

7.1. Model of rental profits

In this section we provide a potential specification $Q(a, \alpha, \lambda)$ based on a "first-come-first-served" demand fulfillment model, see e.g., Pantuso (2022). In particular, we assume that for the target period of time (e.g., 14:00-15:00) the carsharing operator has full knowledge of the set of potential customers in the system, denoted by $\mathcal{K}$. This set is, for example, the set of all registered users on the platform. Furthermore, we assume the operator has a model of how users choose a transportation service, given a list of their attributes, including price. We also assume that there are a number of transport services that compete with carsharing, such as public transport and bicycle. Zonification and pricing decisions are made before the beginning of the target period. Once decisions have been made, each potential customer $k \in \mathcal{K}$ decides, according to the choice model, whether or not to use the carsharing service. This entails that the actual rental demand is determined by zonification and pricing decisions. During the target period, the carsharing system operator fulfills rental requests in the order of their arrival time, subject to the availability of shared vehicles.

To model customer choices in response to pricing and zonification decisions we use the discrete choice model provided by Zheng et al. (2024). In this model, each potential customer $k \in \mathcal{K}$ chooses a transportation service by minimizing a generalized transport cost which is derived from travel time and travel fares. Particularly, the *Value of Time* (VOT) is used to convert travel time into an equivalent amount of money. VOT can vary across customers and transport mode, see e.g., Rossetti et al. (2023). In general, it is higher for the time spent on walking and waiting than that for the time spent on a given transport means. Let $\mathcal{M}$ be the set of available transport modes (e.g., public transit, carsharing and taxi). For simplicity, let $m_0 \in \mathcal{M}$ be the carsharing mode. Let $\mathcal{K}$ be the set of customers. Let μ^V_{km} be the in-vehicle VOT of customer k taking mode m and μ^W_k be the walking-and-waiting VOT of customer k, respectively. Correspondingly, the travel time of customer k with mode m is divided into in-vehicle time indicated by T^V_{km} and walking-and-waiting time indicated by T^W_{km}. The walking-and-waiting time includes the time to reach the transport mode (e.g., station or carsharing station), to commute and possibly wait for the connection, and to reach the final destination (e.g., walking from the station to the final destination). The generalized transport cost can be expressed as

$$c_{km} = p_{km} + \mu^V_{km} T^V_{km} + \mu^W_k T^W_{km}, \quad \forall k \in \mathcal{K}, \forall m \in \mathcal{M}$$

Here p_{km} is the transport fare associated with each customer $k \in \mathcal{K}$ by taking mode $m \in \mathcal{M}$. For

each $k \in \mathcal{K}$ this is given as

$$p_{km} = \begin{cases} P_{km}, & \forall m \in \mathcal{M} \setminus \{m_0\}, \\ P^{CS} T^V_{km} + \sum_{l \in \mathcal{L}} L_l \alpha_{i(k),j(k),l}, & m = m_0. \end{cases}$$

where, for all modes other than carsharing P_{km} is a fixed known fare. For carsharing ($m = m_0$), the first term of the fare is the per-minute fee P^{CS} paid for the duration of the ride T^V_{km}, while the second term is the drop-off fee applied between the customer's origin and destination station, $i(k)$ and $j(k)$, respectively, where L_l is the fee at level $l \in \mathcal{L}$.

Then, a potential customer is identified as a request if there exists a price level $l \in \mathcal{L}$ which makes his/her transport cost for the carsharing service be the smallest among the transport modes. Henceforth, we define the set of requests as follows

$$\mathcal{R} = \{k \in \mathcal{K} : \exists l \in \mathcal{L},\ s.t.\ c_{km_0} \leq c_{km}, \forall m \in \mathcal{M} \setminus \{m_0\}\}$$

For each $r \in \mathcal{R}$ we let $i(r)$ and $j(r)$ be the origin and destination station, respectively, of the customer denoted by $k(r)$. Let $l(r)$ denote the highest acceptable price level of request r. A potential customer $k(r)$ is accounted for as an actual request only when the price level offered on the pair $(i(r), j(r))$ is not larger than level $l(r)$. Furthermore, we let $\mathcal{L}_r = \{l \in \mathcal{L} : L_l \leq L_{l(r)}\}$ denote the set of all acceptable price levels for request r.

Given a set of requests, the allocation of vehicles to requests is done according to a first-come-first-served principle. For each request $r \in \mathcal{R}$, we let $\mathcal{R}_r = \{q \in \mathcal{R} : i(q) = i(r), k(q) < k(r)\}$ be the set of requests from the same station which have priority over request r. We assume that the indices k of the customers represent the order of arrival at the carsharing station (i.e., customer k arrives before customer $k+1$). This is without loss of generality, as the ordering of the customers can be arbitrary and represent different priority relationships.

Let $\mathcal{V}$ represent the set of available shared vehicles. For all $r \in \mathcal{R}$, $v \in \mathcal{V}$, and $l \in \mathcal{L}$, binary variable y_{vrl} indicates whether request r is served by vehicle v at pricing level l. Then, the profit obtained as a result of zonification and pricing decisions is obtained by solving the following MILP:

$$Q(a, \lambda, \alpha) = \max \sum_{r \in \mathcal{R}} \sum_{v \in \mathcal{V}} \sum_{l \in \mathcal{L}_r} R^N_{rl} y_{vrl} \tag{9a}$$

$$\sum_{v \in \mathcal{V}} \sum_{l \in \mathcal{L}_r} y_{vrl} \leq 1, \qquad \forall r \in \mathcal{R} \tag{9b}$$

$$\sum_{r \in \mathcal{R}} \sum_{l \in \mathcal{L}_r} y_{vrl} \leq 1, \qquad \forall v \in \mathcal{V} \tag{9c}$$

$$\sum_{v \in \mathcal{V}} y_{vrl} \leq \alpha_{i(r),j(r),l}, \qquad \forall r \in \mathcal{R}, l \in \mathcal{L}_r \tag{9d}$$

$$\sum_{l\in\mathcal{L}_r} y_{vrl} + \sum_{r_1\in\mathcal{R}_r}\sum_{l_1\in\mathcal{L}_{r_1}} y_{v,r_1,l} \le G_{v,i(r)}, \qquad \forall r\in\mathcal{R}, v\in\mathcal{V} \quad (9e)$$

$$y_{vrl} + \sum_{r_1\in\mathcal{R}_r}\sum_{l_1\in\mathcal{L}_{r_1}} y_{v,r_1,l_1} + \sum_{v_1\in\mathcal{V}\setminus\{v\}} y_{v_1,r,l} \ge \alpha_{i(r),j(r),l} + G_{v,i(r)} - 1, \quad \forall r\in\mathcal{R}, v\in\mathcal{V}, l\in\mathcal{L}_r \quad (9f)$$

$$y_{vrl} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{R}|\times|\mathcal{L}_r|} \quad (9g)$$

In objective function (9a), R^N_{rl} represents the net revenue obtained from serving request r at price level l. It is calculated as

$$R^N_{rl} = c_{k(r),m_0} - C^U_{i(r),j(r)} = P^{CS}T^V_{k(r),m_0} + L_l - C^U_{i(r),j(r)}$$

with $C^U_{i(r),j(r)}$ indicating the operating cost born by the carsharing operator when a vehicle is rented between stations $i(r)$ and $j(r)$. Constraints (9b) ensure that each request is satisfied at most once, while constraints (9c) ensure that each shared vehicle satisfies at most one request. Constraints (9d) state that a given request r can be satisfied at a price level l only if the same price level is set between its origin and destination. Constraints (9e) ensure that a shared vehicle v can be used to satisfy request r only if it was not occupied by any other customers arriving at the same station earlier. Here, parameter G_{vi} takes value 1 if vehicle v is at station i, 0, otherwise. Finally, constraints (9f) will force y_{vrl} to take value 1 if (i) the vehicle v has not been rented by any customer with higher priority (i.e., the second term on the left-hand side is equal to 0), (ii) the request r has not been satisfied by any other shared vehicles (i.e., the third term on the left-hand side is equal to 0), (iii) the price level l is offered to all trips between stations $i(r)$ and $j(r)$ (i.e., $\alpha_{i(r),j(r),l} = 1$), and (iv) vehicle v is available at station $i(r)$ (i.e., $G_{v,i(r)} = 1$).

Observe that problem (9) is always feasible. Its optimal solution can be found in $\mathcal{O}(|\mathcal{R}||\mathcal{V}|)$ operations by the algorithm provided in Appendix A. Given this specification of $Q(a,\lambda,\alpha)$, model (3) becomes a MILP.

We stress model (9) represents only a proxy of the actual performance measure $Q(a,\lambda,\alpha)$ used for illustrative purposes. The model necessarily contains simplifications. As an example, shared vehicles can only be assigned once during the target period. Nevertheless, model (3) and the solution algorithm are general and can be used under more extensive specifications of the performance measure $Q(a,\lambda,\alpha)$.

7.2. Further efficiency measures

In our implementation of the integer Benders decomposition algorithm, (RMP) is solved in a branch-and-bound framework where optimality cuts (6) are identified and added upon reaching integer feasible nodes in the tree.

Valid inequalities (VI-1), (VI-2), (7) and (8) are added statically to (RMP) already at the root node. For valid inequalities (7) and (8) we compute the necessary upper bounds $\overline{P}_{\mathcal{G}}$ and $\overline{P}_{ijl}$ as

follows. Let $\mathcal{V}(\mathcal{G})$ be the tessellation induced by $\mathcal{G}$, see Definition 3.1. Observe that, given our definition of R^N_{rl} in Section 7.1, the profit for each request r depends solely on the origin and destination of the request, $i(r)$, $j(r)$, as well as on l. That is, it is independent on the customers. Let us refer to R^N_{rl} as R^N_{ijl}. Accordingly, the upper bound $\overline{P}_{ijl}$ in (8) can be derived as follows.

$$\overline{P}_{ijl} = R^N_{ijl} \min\left\{\left|\{r \in \mathcal{R} : i(r) = i, j(r) = j, l(r) \geq l\}\right|, \left|\mathcal{V}_i\right|\right\}, \forall i, j \in \mathcal{I}, l \in \mathcal{L}$$

That is, the upper bound is given by the profit obtained by all requests going from station i to j, provided that there are sufficient vehicles at the origin station (i.e., limited by the number of vehicles $|\mathcal{V}_i|$).

Then, for each pair of zones $\mathcal{V}$,$\mathcal{U} \in \mathcal{V}(\mathcal{G})$ and for each price level $l \in \mathcal{L}$, an upper bound on the profit between such zones at price level l can be computed by taking the sum of the maximal profits obtained by serving requests from stations in $\mathcal{V}$ to stations in $\mathcal{U}$, that is $\sum_{(i,j)\in\mathcal{V}\times\mathcal{U}} \overline{P}_{ijl}$. We identify the price level which gives the highest profit upper bound for zone pair ($\mathcal{V}$,$\mathcal{U}$) as $l^*_{\mathcal{V},\mathcal{U}} = \text{argmax}_{l\in\mathcal{L}} \left\{\sum_{(i,j)\in\mathcal{V}\times\mathcal{U}} \overline{P}_{ijl}\right\}$. Then $\overline{P}_{\mathcal{G}}$ is obtained by summing up the profit upper bounds from all pairs of zones given the identified pricing levels, as illustrated in the following.

$$\overline{P}_{\mathcal{G}} = \sum_{\mathcal{V},\mathcal{U}\in\mathcal{V}(\mathcal{G})} \sum_{(i,j)\in\mathcal{V}\times\mathcal{U}} R^N_{ijl^*_{\mathcal{V},\mathcal{U}}} \overline{P}_{ijl^*_{\mathcal{V},\mathcal{U}}}, \forall \mathcal{G} \in \mathcal{G}^S$$

In addition, we add classical duality-based Benders decomposition cuts (10) generated from the LP relaxation of problem (9).

$$\begin{aligned}\phi \leq & \sum_{r\in\mathcal{R}} \pi^A_{rk} + \sum_{v\in\mathcal{V}} \pi^B_{vk} + \sum_{r\in\mathcal{R}} \sum_{l\in\mathcal{L}_r} \alpha_{i(r),j(r),l} \pi^C_{rlk} + \sum_{r\in\mathcal{R}} \sum_{v\in\mathcal{V}} G_{v,i(r)} \pi^D_{vrk} \\ & + \sum_{v\in\mathcal{V}} \sum_{r\in\mathcal{R}} \sum_{l\in\mathcal{L}_r} (\alpha_{i(r),j(r),l} + G_{v,i(r)} - 1) \pi^E_{vrlk} \end{aligned} \tag{10}$$

where, π^A_{rk}, π^B_{vk}, π^C_{rlk}, π^D_{vrk}, and π^E_{vrlk} are the optimal dual solutions at iteration k corresponding to constraints (9b), (9c), (9d), (9e), and (9f), respectively. Relaxation cuts define a non-trivial upper bound on $Q(a, \lambda, \alpha)$. They are generated and added to (RMP) once an integer feasible solution $(a^k, \lambda^k, \alpha^k, \phi^k)$ to (RMP) violates the optimality condition $\phi^k \leq Q(a^k, \lambda^k, \alpha^k)$.

Finally, since in our instances the fleet is homogeneous, a valid upper bound U for the expression of the optimality cuts (6) can be set as

$$U = \sum_{r\in\mathcal{R}} \max\{R^N_{r,l(r)}, 0\}$$

It entails that all the requests contributing to a positive revenue are served, and only those, and that these are served at the respective highest price level.

7.3. Instances

We build instances based on a real-world carsharing service operating in Copenhagen, Denmark. The instances comprise the twenty stations illustrated in Figure 2. The transport modes available in the city (set $\mathcal{M}$) comprise public transport (a service offering busses, metro and superficial trains) and taxi, in addition to carsharing.

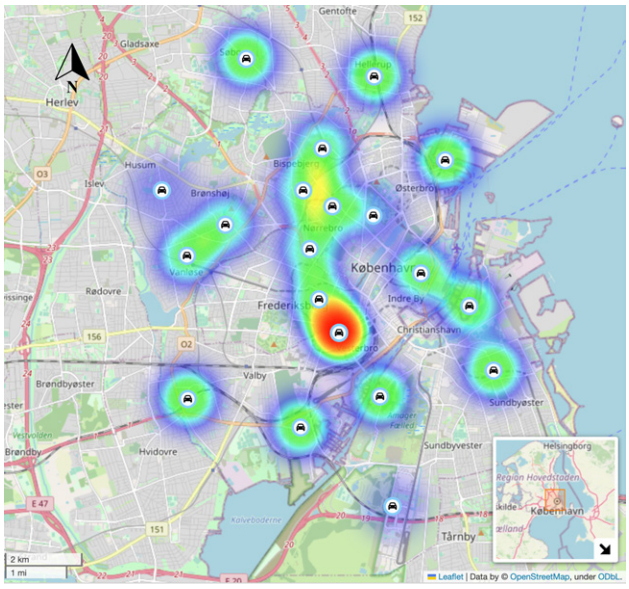


Figure 2: Location of carsharing stations and distribution of POIs.

For the carsharing service we set five drop-off fees, ranging from Euro -2 to 2 with Euro 1 intervals according to the values used in (Pantuso, 2022). The per-minute fee is set at Euro 0.30 consistently with ongoing practice. The tariffs for taxis and public transit are likewise set according to current tariffs in Copenhagen. These are summarized in Table 2.

Customers are created based on the points of interest (POIs) located in an area of radius 800 meters centered around each station. POIs are locations such as schools, hospitals and museums and we assume they represent the origins and destinations of the customers. Figure 2 provides a heatmap of the POIs considered, thus of the origins and destinations of the customers. Particularly, each customer is created by randomly sampling, without replication, an origin and destination pair from the POIs.

For any two POIs, we obtain real-world walking times and travel times and for the different transport modes using Google Maps APIs. For the carsharing mode, the waiting time is set to zero. For public transit, Google Maps APIs return the overall travel time including both walking-and-waiting time and in-vehicle time. Thus, we assume the walking-and-waiting time is uniformly distributed between 4 and 15 minutes and is subtracted from the overall travel time. This represents the time necessary to walk to the a nearby public transit station, to switch between public transit services (e.g., between bus and metro), to reach the next public transit station, and finally to walk

Table 2: Transport prices.

Parameter	Values
Carsharing drop-off fee	-2, -1, 0, 1, 2 Euro for each pricing level
Carsharing per-minute fee	0.30 Euro
Taxi pick-up fee	3.89 Euro
Taxi per-minute fee	2.55 Euro
Public transit ticket fee	3.22 Euro

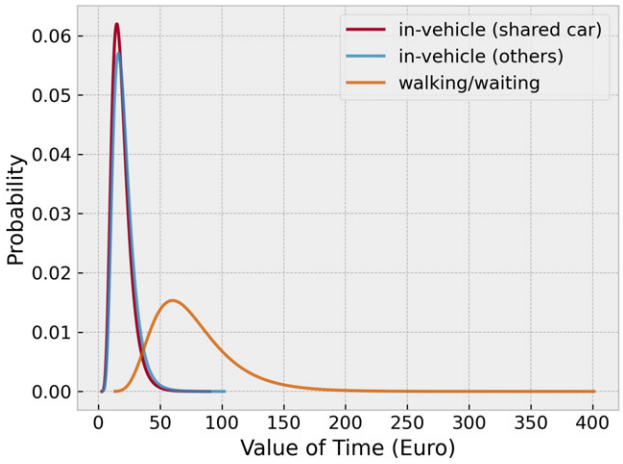


Figure 3: Log-normal distributions of VOT values for different transport modes.

to the destination POI. Finally, the waiting time for a taxi is uniformly distributed between 4 and 8 minutes while the walking time is zero.

VOTs depend on the transport mode. Following Rossetti et al. (2023), we assume the VOTs are distributed according to a log-normal distribution with standard deviation Euro 0.4 and different mean values. Particularly, the mean values are Euro 2.86 for the in-shared-vehicle time, Euro 2.94 for the in-vehicle time of other transport modes, and Euro 4.25 for the walking-and-waiting time, see Figure 3. Consequently, for each customer, we randomly sample a realization of the VOTs from the distributions above.

We build two groups of instances. The first group comprises small instances with 10 stations. The second group comprises larger instances with 20 stations. We control the parameters of number of zones, customers and shared vehicles for generating instances with different problem sizes. In both instance groups, we consider a number of zones $S \in \{3, 4, 5\}$ and varying vehicle-to-customer ratios ranging from 1 : 4 to 1 : 2. Particularly, small instances are generated by considering the following combinations of number of customers and number of vehicles, $(|\mathcal{K}|, |\mathcal{V}|) \in \{(100, 25), (100, 40), (100, 50), (200, 50), (200, 75), (200, 100), (300, 75), (300, 100), (300, 150)\}$. Larger instances are likewise created from the following configurations $(|\mathcal{K}|, |\mathcal{V}|) \in \{(400, 100), (400, 150), (400, 200), (600, 150), (600, 200), (600, 300), (800, 200), (800, 300), (800, 400)\}$. The locations of shared vehicles are randomly generated according to the densities of

customers across all origin stations. Given the randomness on generating customer O-Ds and the vehicle locations, we set 5 different random seeds for generating multiple instances with the same problem size. In overall, we create 270 instances, each of which is named based on the values taken for control parameters. For example, K100V40I10S3-r0 indicates an instance with 100 customers, 40 vehicles, 10 stations, 3 zones, and generated at random seed 0. For the sake of replicability, all generated instances and relevant data are made available in a public repository with link `https://github.com/AlexDeng1/Instances_PricingZonification`.

7.4. Computational performance

In this section we start by investigating the computational efficiency of the monolithic formulations induced by different reformulations of the $\mathcal{T}$ set and valid inequalities (VIs) introduced in section 5 for problem (3). The considered versions of (3) are named "Original", "WF" and "CGLM" and constructed with using $\mathcal{T}$, $\mathcal{T}^{WF}$ and $\mathcal{T}^{CGLM}$, respectively. All formulations give rise to MILP problems. The extensive formulation of the "Original" formulation is provided as an example in Appendix B.

On the basis of the most efficient formulation, we then report on the performance of the integer Benders decomposition algorithm (BD in what follows). Such performance is compared to that of commercial solver Gurobi 11.0.0 (the solver, in what follows) employed to solve the same monolithic formulation of problem (3). The BD algorithm is implemented in Python 3.11 using Gurobi's callable libraries. The monolithic formulation is likewise solved using Gurobi's Python libraries. In all tests we set a time limit of 1800 seconds. All tests were executed on machines equipped with 2.10GHz Intel(R) Xeon(R) Gold 6230 40 core CPU and 240 Gb RAM.

7.4.1. Evaluation of monolithic formulations and valid inequalities

The three monolithic formulations, with and without valid inequalities, are compared on small instances with 10 stations. The results are summarized in Table 3. For each instance group, we report the average solution time over five randomly generated instances.

Table 3 shows that the CGLM formulation outperforms the other two formulations. Specifically, the CGLM formulation requires the least solution time in 19 out of 27 instance groups, and on average, it reduces computational time by 8.66% and 6.02% compared to the Original and the WF formulation, respectively. The inclusion of both (VI-1) and (VI-2) yields a sensible reduction of the solution time for all formulations. The average reduction in computational time can be quantified in 17.90% for the Original formulation, 11.32% for the WF formulation and 12.75% for the CGLM formulation. Overall, the best performance is delivered by the CGLM formulation with (VI-1) and (VI-2). Therefore, we adopt this formulation in our examination of the BD method.

7.4.2. Performance of the BD algortihm

Table 4 compares the performance of the BD algorithm (henceforth BD) to that of the monolithic CGLM formulation on small instances. Valid inequalities (VI-1) and (VI-2) are always added

Table 3: Average solution time in seconds on small instances for the Original, WF and CGLM formulations with/without the additions of valid inequalities.

Instance group	Original				WF				CGLM			
	No VIs	VI-1	VI-2	VI-1&2	No VIs	VI-1	VI-2	VI-1&2	No VIs	VI-1	VI-2	VI-1&2
K100V25S3	4.94	4.29	4.63	3.56	4.42	4.67	4.30	3.91	3.48	3.78	3.50	3.22
K100V40S3	5.24	5.26	5.11	4.50	5.07	4.77	4.64	3.96	4.20	4.17	3.98	3.76
K100V50S3	4.88	4.79	4.71	4.46	5.20	4.78	5.05	4.31	4.24	3.74	4.74	3.49
K200V50S3	20.42	16.39	23.10	16.94	22.29	17.60	20.07	20.93	19.58	23.08	17.32	11.84
K200V75S3	9.88	9.31	9.17	8.52	16.91	8.86	14.66	8.07	8.98	8.64	8.36	9.50
K200V100S3	9.67	12.91	10.48	10.15	11.36	12.76	10.21	11.22	10.46	8.10	9.94	9.47
K300V75S3	33.99	30.15	39.86	23.77	31.88	33.16	33.85	31.40	30.99	37.40	30.61	28.47
K300V100S3	26.21	17.35	19.36	19.93	29.06	15.64	20.07	15.96	22.16	23.15	21.00	17.16
K300V150S3	16.54	16.71	18.98	13.88	17.99	15.54	13.76	14.04	20.34	23.27	21.66	15.43
S=3 Average	**14.64**	**13.02**	**15.04**	**11.75**	**16.02**	**13.09**	**14.07**	**12.64**	**13.83**	**15.04**	**13.46**	**11.37**
K100V25S4	3.92	4.16	3.25	3.70	4.14	3.45	3.48	3.44	3.50	2.65	2.73	2.70
K100V40S4	4.43	4.20	3.72	3.57	4.72	3.57	3.77	3.46	3.28	3.07	3.60	3.42
K100V50S4	3.47	4.37	3.84	3.58	4.18	4.85	3.59	3.34	3.25	3.69	3.08	3.15
K200V50S4	20.54	18.62	16.79	14.57	19.13	17.11	19.40	14.15	16.15	13.75	17.14	11.37
K200V75S4	13.46	12.02	9.21	9.33	10.41	11.02	11.86	13.41	9.19	9.96	8.87	9.29
K200V100S4	12.80	13.93	11.92	13.83	9.23	15.35	11.15	8.78	10.46	11.97	12.26	12.82
K300V75S4	30.97	33.03	31.52	30.05	32.33	31.07	30.53	31.39	31.30	22.06	23.77	20.62
K300V100S4	28.00	24.72	30.13	21.33	26.25	29.35	27.60	29.55	31.01	21.32	31.48	28.88
K300V150S4	33.32	19.57	31.99	23.10	23.21	19.93	24.18	19.69	22.18	22.65	23.01	24.63
S=4 Average	**16.77**	**14.96**	**15.82**	**13.67**	**14.84**	**15.08**	**15.06**	**14.13**	**14.48**	**12.35**	**13.99**	**12.99**
K100V25S5	1.92	2.04	1.40	1.70	1.96	2.00	1.35	1.57	1.67	1.65	1.24	1.29
K100V40S5	2.54	2.20	1.64	1.82	2.07	2.12	1.65	1.72	1.95	1.99	1.62	1.51
K100V50S5	2.28	2.21	1.66	1.92	2.34	2.11	1.84	1.64	2.04	2.05	1.74	1.50
K200V50S5	11.15	8.22	10.05	5.06	9.78	8.04	7.94	9.36	7.76	6.33	6.35	7.92
K200V75S5	6.12	6.71	5.87	6.03	6.81	8.15	5.15	5.49	5.28	5.53	5.02	6.07
K200V100S5	7.06	9.08	6.35	8.26	9.35	7.92	8.80	6.76	8.56	6.70	7.66	7.33
K300V75S5	21.30	19.95	20.80	23.26	16.64	16.12	11.80	24.77	19.36	21.24	20.12	21.55
K300V100S5	19.85	17.35	14.60	14.78	17.74	12.43	22.94	15.49	15.79	13.61	13.79	15.51
K300V150S5	20.74	20.40	21.37	16.70	20.62	17.60	17.24	16.05	25.93	15.06	14.06	17.49
S=5 Average	**10.33**	**9.80**	**9.30**	**8.84**	**9.70**	**8.50**	**8.75**	**9.20**	**9.82**	**8.24**	**7.96**	**8.91**
Overall Average	**13.91**	**12.59**	**13.39**	**11.42**	**13.52**	**12.22**	**12.63**	**11.99**	**12.71**	**11.87**	**11.80**	**11.09**

both in BD and in the monolithic formulation. For the BD algorithm we report the results with and without the addition of BD-tailored VIs (i.e., (7) and (8)). Valid inequalities (7) are added for all $\binom{|\mathcal{I}|}{S}$ possible sets of generators. In both BD with and without VIs we add relaxation cuts as explained in Section 7.2. Table 4 does not report optimality gaps as all instances are solved to Gurobi's default precision (10^{-4}).

The first observation is that BD-specific VIs are effective. They decrease the solution time by 55.19% with $S = 5$, 46.68% with $S = 4$, and 42.35% with $S = 3$. On the instances with $S = 3, 4$ BD, with the addition of VIs, significantly outperforms the solver both in terms of average solution time and in terms of the number of instances solved faster. The results are essentially reversed when $S = 5$. The average solution time with the solver decreases as the number of zones increases. When increasing the number of zones we progressively relax the feasible region by including a larger number of partitions (observe that $\binom{|\mathcal{I}|}{S} < \binom{|\mathcal{I}|}{S+1}$ always holds in our instances and in general when $|\mathcal{I}| > 2S + 1$). At the same time the number of decision variables and constraints is unaffected.

Table 5 reports additional results on larger instances with 20 stations. Particularly, it reports average optimality gap aggregated by instance group and number of zones. For BD we use BD-specific VIs as they were proven effective on the small instances, see Section 7.4.2.

We observe that on these instances BD significantly outperforms the solver. The average optimality gap of BD is 2.40% while that of the solver is 18.24%. The optimality gap reduction is evident in all instance groups and becomes more pronounced when S decreases, thus when the

Table 4: Solution time (in seconds) for the solver and the BD algorithms. Column "# Wins" reports the number of instances that are solved faster by BD with VIs than the solver.

Instance group	BD without VIs	BD with VIs	Solver	# Wins
K100V25S3	5.23	4.05	3.22	2/5
K100V40S3	5.68	3.10	3.76	3/5
K100V50S3	3.73	2.93	3.49	3/5
K200V50S3	11.72	5.45	11.84	5/5
K200V75S3	9.32	3.85	9.50	5/5
K200V100S3	8.36	3.16	9.47	5/5
K300V75S3	36.63	25.29	28.47	3/5
K300V100S3	44.45	30.96	17.16	4/5
K300V150S3	17.84	3.60	15.43	5/5
S=3 Average	**15.89**	**9.16**	**11.37**	**35/45**
K100V25S4	6.58	4.42	2.70	1/5
K100V40S4	3.34	2.52	3.42	3/5
K100V50S4	3.33	2.50	3.15	3/5
K200V50S4	10.28	8.34	11.37	2/5
K200V75S4	8.94	3.42	9.29	5/5
K200V100S4	8.22	3.19	12.82	5/5
K300V75S4	52.40	31.76	20.62	2/5
K300V100S4	18.19	8.94	28.88	5/5
K300V150S4	17.68	3.63	24.63	5/5
S=4 Average	**14.33**	**7.64**	**12.99**	**31/45**
K100V25S5	5.92	5.57	1.29	0/5
K100V40S5	2.66	2.28	1.51	1/5
K100V50S5	2.73	2.28	1.50	0/5
K200V50S5	8.24	6.36	7.92	2/5
K200V75S5	5.77	3.72	6.07	3/5
K200V100S5	6.62	2.81	7.33	5/5
K300V75S5	596.70	260.50	21.55	2/5
K300V100S5	14.16	8.97	15.51	4/5
K300V150S5	17.62	3.44	17.49	5/5
S=5 Average	**73.38**	**32.88**	**8.91**	**22/45**

feasible region is smaller. In particular, the optimality gap is decreased by 95.54% with $S = 3$, 87.00% with $S = 4$, and 81.28% with $S = 5$. Furthermore, BD successfully closes the optimality gap in 95 out of 135 instances, while only 25 of these instances are solved to optimality by the solver. For the instances that can be solved both by BD and the solver, the solution time is significantly smaller with BD. On average, BD takes 707.44 seconds to solve the 95 instances to optimality, while the solver takes 1495.46 seconds for the respective 25 instances.

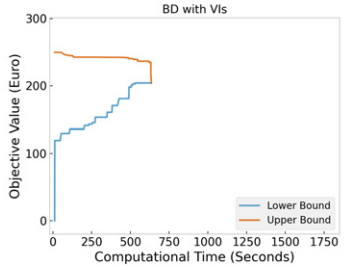
Figure 4 reports the progression of upper and lower bounds for both BD with VIs and the solver on three large instances with $S = 3$, where BD closes the optimality gap. In Figures 4a to 4c we observe a steady and rapid progression of both upper (dual) and lower (primal) bounds when using BD. At the same time, Figures 4d to 4f illustrate that, for the same instances, the solver is able to improve the primal bound but fails to improve the dual bound. Particularly, the solver finds primal solutions of quality comparable to those found by BD. Nevertheless, BD typically obtains such solutions faster. We further observe that the optimality gap of the solver tends to increase with the number of customers. This indicates that part of the explanation of the gap is to be found in the size of the LP relaxation. This further justifies the use of a decomposition method.

7.5. Analysis of the result and discussion

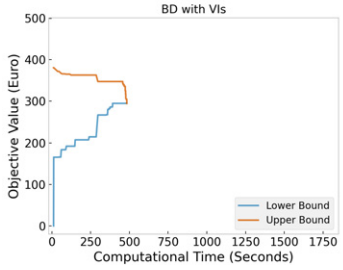
In this section, we provide evidence on how different parameters, such as the number of zones, customers and fleet size affect profits. Furthermore, we compare the proposed optimal partition

Table 5: Average optimality gap after 1800 seconds. Optimality gaps are computed as |`best_bound`-`objective_value`|/(ϵ+|`objective_value`|). The column "Reduction" reports the difference in optimality gap between BD and the solver as a percentage. "Instances solved" reports the number of instances that are solved to a target 10^{-4} optimality gap, and "Solution time" reports the average solution time of the instances solved to optimality.

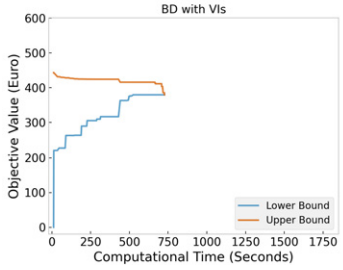
Instance group	**Gap (%)**			**Instance Solved**		**Solution time (s)**	
	Solver	BD	Reduction (%)	Solver	BD	Solver	BD
K400V100S3	13.72	0.00	100.00	2/5	5/5	1629.05	779.45
K400V150S3	14.70	0.00	100.00	2/5	5/5	952.13	457.02
K400V200S3	15.64	0.00	100.00	2/5	5/5	1459.55	325.11
K600V150S3	31.34	6.20	80.20	0/5	2/5	-	738.90
K600V200S3	30.23	0.00	100.00	0/5	5/5	-	654.54
K600V300S3	6.05	0.00	100.00	4/5	5/5	1492.00	354.69
K800V200S3	32.21	6.54	79.70	0/5	1/5	-	1368.79
K800V300S3	12.65	0.00	100.00	3/5	5/5	1607.00	591.90
K800V400S3	14.35	0.00	100.00	3/5	5/5	1551.36	323.55
S=3 Average	**18.99**	**1.42**	**95.54**	**16/45**	**38/45**	**1448.52**	**621.55**
K400V100S4	13.79	6.48	52.99	1/5	2/5	1720.65	1016.65
K400V150S4	19.35	0.00	100.00	0/5	5/5	-	739.24
K400V200S4	20.40	0.00	100.00	0/5	5/5	-	465.84
K600V150S4	24.56	7.05	71.28	0/5	1/5	-	778.49
K600V200S4	24.99	0.00	100.00	0/5	5/5	-	814.03
K600V300S4	8.55	0.00	100.00	3/5	5/5	1473.66	392.57
K800V200S4	26.37	9.29	64.79	0/5	0/5	-	-
K800V300S4	10.21	0.62	93.94	2/5	3/5	1443.32	1016.74
K800V400S4	9.89	0.00	100.00	3/5	5/5	1431.97	619.26
S=4 Average	**17.57**	**2.60**	**87.00**	**9/45**	**31/45**	**1517.40**	**730.35**
K400V100S5	12.27	9.05	26.24	0/5	0/5	-	-
K400V150S5	13.79	0.20	98.56	0/5	4/5	-	886.65
K400V200S5	15.00	0.00	100.00	0/5	5/5	-	501.27
K600V150S5	18.47	7.38	60.04	0/5	0/5	-	-
K600V200S5	18.42	1.10	94.03	0/5	3/5	-	1112.87
K600V300S5	19.04	0.00	100.00	0/5	5/5	-	668.54
K800V200S5	22.89	10.34	54.84	0/5	0/5	-	-
K800V300S5	21.44	0.47	97.80	0/5	4/5	-	842.40
K800V400S5	22.12	0.00	100.00	0/5	5/5	-	610.83
S=5 Average	**18.16**	**3.17**	**81.28**	**0/45**	**26/45**	-	770.43
Overall Average	**18.24**	**2.40**	**87.94**	**25/135**	**95/135**	**1482.96**	**707.44**



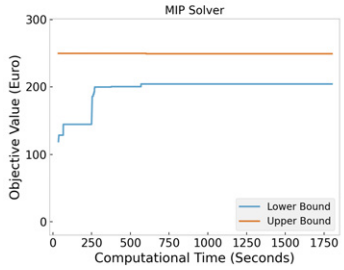
(a) K400V200S3-r3



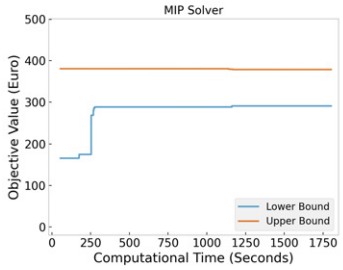
(b) K600V400S3-r3



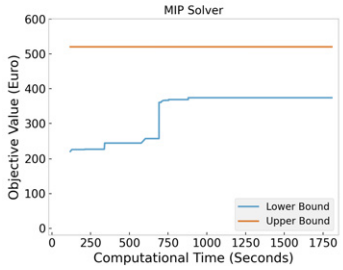
(c) K800V400S3-r3



(d) K400V200S3-r3



(e) K600V300S3-r3

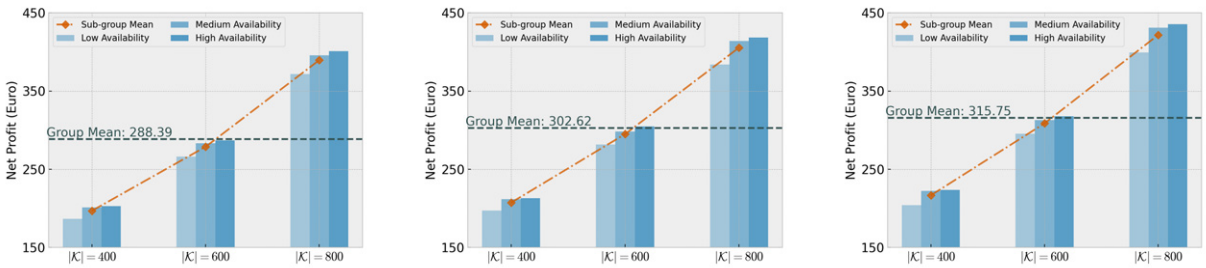


(f) K800V400S3-r3

Figure 4: Progression of upper and lower bounds for the large instances with three zones.

strategy against a predefined zip-code based partition and against a system without zones. Finally, we discuss the impact of the number of zones.

We begin by assessing how profits change with the number of customers and vehicles. The results are obtained by solving the largest instances with 20 stations. Figure 5 reports the average profit as a function of the number of customers and fleet size. Particularly, for $|\mathcal{K}| = 400$ we consider three fleet sizes, namely $(100, 150, 200)$ which we refer to a low, medium and how availability. Similarly, for $|\mathcal{K}| = 600$, the three fleet sizes are $(150, 200, 300)$ and for $|\mathcal{K}| = 800$ the three fleet sizes are $(200, 300, 400)$. Figure 5 illustrates that the profit grows linearly with the number of customers, regardless of the number of zones S. When the customers volume doubles from 400 to 800, the profit increases by 97.79%, 95.58%, and 94.60% for the 3-, 4-, and 5-zone partitions, respectively. Profits grow also with the fleet size, though the pattern is less regular. In general, a medium level of vehicle availability suffices to secure average profit returns, given a certain customer volume. Finally, we observe that a finer partition of the carsharing stations (i.e., larger S) yields an increase of the average profit. Technically, the average profit grows by 9.49% when a 5-zone partition is applied compared to a 3-zone partition. A finer partition allows the service provider to better adapt prices to the specific characteristics of the customers in different parts of the city.



(a) $S = 3$ (b) $S = 4$ (c) $S = 5$

Figure 5: Carsharing system profits under various scales of zone numbers, customer volume, and vehicle availability.

In order to show the efficacy of the proposed model, we evaluate the performance of the carsharing system when using the optimal zonification determined by model (3) (we assume $S = 3$ zones) compared to two benchmarks. The first benchmark is obtained by partitioning the carsharing stations according to the zip codes and keeping that partition fixed when optimally setting prices. Also in this case we use $S = 3$ zones. We refer to this benchmark as the *zip-code partition* benchmark. The second benchmark consists of not partitioning the carsharing stations and thus optimally choosing the one and only drop-off fee that must apply between all pairs of carsharing stations. This benchmark is referred to as the *no-partition* benchmark. Table 6 reports profits and service rates with 400, 600 and 800 customers assuming a fleet size of 200, 300, and 400, respectively. Service rates are calculated as the number of customers served over the total number of requests $|\mathcal{R}|$.

Table 6: Service rates and profits for the optimal partition compared to the zip-code partition and no-partition benchmarks. The results are obtained with $S = 3$ zones.

$\|\mathcal{K}\|$	Optimal Partition		Zip-code Partition		No-Partition	
	Service Rate (%)	Profit (Euro)	Service Rate (%)	Profit (Euro)	Service Rate (%)	Profit (Euro)
400	86.41	202.74	79.41	187.74	72.53	165.41
600	80.85	286.92	78.57	272.91	76.62	239.61
800	79.10	401.03	81.03	386.38	79.71	363.00
Avg.	82.12	296.90	79.67	282.34	76.29	256.01

The results illustrate that by implementing an optimal partition, and thus ensuring co-optimized zones and prices, the profit increases substantially both with respect to the zip-code partition and to the no-partition benchmark. Specifically, when compared to the zip-code partition benchmark, the operating profit with the optimal partition increases by 7.99%, 5.13%, and 3.79% for cases with 400, 600, and 800 customers, respectively. The improvement becomes even more pronounced when compared with the no-partition scenario. Particularly, the profit increases by 22.57%, 19.74%, and 10.48% for three different customer volumes.

When using the optimal partition, the average service rate is 82.12%, much higher than both the service rate of the zip-code partition benchmark and that of the no-partition benchmark. We recall that model (3) is designed to maximize profits, while the service rate is simultaneously ensured with flexible pricing zone partition decisions.

Finally, we analyze the impact of the number of zones. For this purpose, we test the zonification and pricing model with the number of zones S set to values chosen from the set $\{1, 2, 3, 4, 5, |\mathcal{I}|\}$. The scenario with $S = 1$ corresponds to a system where the business area is not partitioned (i.e., the "No-Partition" setting in Table 6). Here, the same, optimized, price applies to all pairs of stations. The scenario with $|\mathcal{I}|$ zones is the most flexible. Here, the drop-off fee can optimally adapt to each specific station pair. This scenario necessarily yields the highest profit.

We introduce the metric δ_{ub} computed as `profit/profit upper bound` $\times 100$. Here `profit` is the profit obtained by means of an optimal pricing and zonification into $S \in \{1, 2, 3, 4, 5\}$ zones, while `profit upper bound` is the profit obtained with $S = |\mathcal{I}|$. Hence, δ_{ub} determines the share of the largest possible profit obtained by means of a zonification into S zones. Furthermore, we analyze profitability rate, which is calculated as `profit/cost` for different values of S. In addition, to assess the effect of zonification decisions on the customer side, we analyze the average drop-off fee and rental fee. These statistics are reported in Table 7 for instances with 20 stations, 400, 600 and 800 customers, and 200, 300, and 400 vehicles.

We observe that, in our instances, $S = 3$ zones are typically sufficient to capture up to approximately three-quarters of the largest possible profit. With $S = 5$, the δ_{ub} values further increase to 87.05%, 83.70%, and 81.26% for $|\mathcal{K}| = 400$, 600, and 800, respectively. In addition, we observe that the growth in terms of profitability slows down significantly as S increases. As offering more flexible prices (i.e., with larger S) tends to induce more rentals from the request set $\mathcal{R}$, this in turn

Table 7: Economic effects of increased zonification granularity for carsharing service providers and customers.

		Operator side				**Customer side**	
$\|\mathcal{K}\|$	S	Profit (Euro)	δ_{ub} (%)	Cost (Euro)	Profitability rate	Avg. Drop-off fee (Euro)	Rental fee (Euro)
400	1	165.41	64.43	68.47	2.42	-0.15	4.45
	2	189.70	73.89	78.50	2.42	-0.12	4.53
	3	202.74	78.97	82.12	2.47	-0.07	4.58
	4	212.89	82.92	80.23	2.65	0.17	4.87
	5	223.50	87.05	84.30	2.65	0.16	4.82
	$\|\mathcal{I}\| = 20$	256.73	-	90.73	2.83	0.40	4.96
600	1	239.61	63.14	106.41	2.25	-0.29	4.33
	2	268.88	70.86	110.32	2.44	-0.03	4.69
	3	286.92	75.61	114.78	2.50	0.06	4.79
	4	304.82	80.33	115.40	2.64	0.23	5.03
	5	317.63	83.70	120.31	2.64	0.23	4.99
	$\|\mathcal{I}\| = 20$	379.48	-	134.90	2.81	0.48	5.12
800	1	363.00	67.77	158.96	2.28	-0.26	4.35
	2	386.87	72.22	161.61	2.39	-0.12	4.53
	3	401.03	74.87	161.39	2.48	0.02	4.73
	4	418.58	78.14	162.20	2.58	0.12	4.83
	5	435.27	81.26	169.91	2.56	0.12	4.81
	$\|\mathcal{I}\| = 20$	535.67	-	192.33	2.79	0.43	5.06

imposes higher operation costs on the service provider.

From the customer's point of view, we observe that finer partitions and more flexible pricing strategies generally lead to higher carsharing costs. Specifically, the average drop-off fee increases by approximately 1.5, 1.1, and 2.6 times when comparing the fully-partitioned setup to the 5-zone partition in scenarios with 400, 600, and 800 potential customers, respectively. Similarly, the total rental fee rises by 2.90%, 2.61%, and 5.20% in these scenarios.

Therefore, it emerges that while optimized station-to-station prices are necessarily the best strategy in terms of profits, they tend to generate higher costs for the customers. However, a zonification into a few zones is typically sufficient to capture a large share of the potential profits without excessively compromising profitability.

8. Concluding remarks

In this paper, we studied the problem of jointly deciding pricing zones and trip prices in a carsharing service. To partition the carsharing stations into distinct pricing zones which should be "visually disjoint" we introduced a special type of tessellation. We proved that such tessellation ensures that the resulting partition generates mutually non-overlapping convex hulls. This geometric property facilitates the design of user-friendly pricing strategies that ease price communication towards customers. The problem of choosing such a tessellation, and optimally assigning prices has been formulated as a binary integer programming problem for which three formulations have been proposed and compared. To solve the problem we designed a tailored integer Benders decomposition, which incorporates a number of problem-specific improvements.

We performed extensive experiments on instances based on a carsharing system operated in Copenhagen. The results illustrate that the proposed Benders decomposition significantly outperforms a state-of-the-art solver on instances of size comparable with business practices. Particularly, our method reports an average 2.40% optimality gap, which is significantly lower to that reported by the solver (18.24%). Furthermore, our method closes the optimality gap in 95 out of 135 large instances while only 25 instances in the same group are solved to optimality by the solver.

Furthermore, our results illustrate that the optimal zonification yields profits that are 5.16% higher (on average) compared to the zonification determined by zip codes and 15.97% higher compared to the prevailing practice of having only one pricing zone. Notably, the optimal zonification yields also the highest service rates, by serving 82.12% of the customers, on average. When analyzing the economic impact of the number of zones, we find that a few zones are sufficient to capture a large share of the potential profit that can be obtained under the most flexible setup (i.e., with station-to-station prices). At the same time, the rental cost born by customers remains significantly lower compared to a fully optimized pricing scheme. These observations offer strategic insight for carsharing operators on the effect of establishing optimized zone prices.

This research is not free from limitations, which leave room for future extensions. In our problem, we assume that price affects carsharing demand in a deterministic way. In practice, carsharing demand can be both stochastic and price-dependent. This gives rise to stochastic optimization problem with decision-dependent uncertainty for which, to the best of our knowledge, there are no general purpose solution methods. Moreover, given the novelty of the proposed problem, we aimed to provide an exact algorithm in order to have both primal and dual bounds. The development of heuristic algorithms may help provide primal solutions in shorter solution times, if needed. In addition, other operational decisions, primarily staff-based relocations, might have a significant impact on the managerial insights emerged from our study. The inclusion of such decisions into the model would likely represent an improvement of our work. Finally, the proposed methodology has been developed with a station-based system in mind. Clearly, similar arguments would motivate the application of joint zonification and pricing in free-floating systems. The model and method provided may directly apply provided that carsharing stations are replaced by suitable demand points in a free-floating system.

CRediT authorship contribution statement

Jiali Deng: Conceptualization, Methodology, Software, Formal analysis, Investigation, Visualization, Writing. **Giovanni Pantuso:** Conceptualization, Methodology, Supervision, Writing - Review & Editing.

Acknowledgement

This research is supported by the research project "Shared mobility: Towards sustainable urban transport" (grant no. 1127-00176B) funded by Danmarks Frie Forskningsfond (DFF). This work has been performed using the Danish National Life Science Supercomputing Center, Computerome.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Angelopoulos, A., Gavalas, D., Konstantopoulos, C., Kypriadis, D., Pantziou, G., 2018. Incentivized vehicle relocation in vehicle sharing systems. Transportation Research Part C: Emerging Technologies 97, 175–193.

Bard, J.F., Jarrah, A.I., 2009. Large-scale constrained clustering for rationalizing pickup and delivery operations. Transportation Research Part B: Methodological 43, 542–561.

Bender, M., Meyer, A., Kalcsics, J., Nickel, S., 2016. The multi-period service territory design problem–an introduction, a model and a heuristic approach. Transportation Research Part E: Logistics and Transportation Review 96, 135–157.

Bergey, P.K., Ragsdale, C.T., Hoskote, M., 2003a. A decision support system for the electrical power districting problem. Decision Support Systems 36, 1–17.

Bergey, P.K., Ragsdale, C.T., Hoskote, M., 2003b. A simulated annealing genetic algorithm for the electrical power districting problem. Annals of Operations Research 121, 33–55.

Boyacı, B., Zografos, K.G., 2019. Investigating the effect of temporal and spatial flexibility on the performance of one-way electric carsharing systems. Transportation Research Part B: Methodological 129, 244–272.

Bozkaya, B., Erkut, E., Laporte, G., 2003. A tabu search heuristic and adaptive memory procedure for political districting. European journal of operational research 144, 12–26.

Cánovas, L., García, S., Labbé, M., Marín, A., 2007. A strengthened formulation for the simple plant location problem with order. Operations Research Letters 35, 141–150.

Carlsson, J.G., Delage, E., 2013. Robust partitioning for stochastic multivehicle routing. Operations research 61, 727–744.

Di Febbraro, A., Sacco, N., Saeednia, M., 2012. One-way carsharing: solving the relocation problem. Transportation research record 2319, 113–120.

Dobson, G., Karmarkar, U.S., 1987. Competitive location on a network. Operations Research 35, 565–574.

Duque, J.C., Anselin, L., Rey, S.J., 2012. The max-p-regions problem. Journal of Regional Science 52, 397–419.

Eilertsen, U., Falck-Pedersen, O.M., Henriksen, J.V., Fagerholt, K., Pantuso, G., 2024. Joint relocation and pricing in electric car-sharing systems. European Journal of Operational Research 315, 553–566.

Espejo, I., Marín, A., Rodríguez-Chía, A.M., 2012. Closest assignment constraints in discrete location problems. European Journal of Operational Research 219, 49–58.

Free2move, 2025. What is a drop off fee? URL: `https://www.free2move.com/it/en/faq/car-sharing/rate-and-fees/#what-is-a-drop-off-fee`.

Galvao, L.C., Novaes, A.G., De Cursi, J.S., Souza, J.C., 2006. A multiplicatively-weighted voronoi diagram approach to logistics districting. Computers & Operations Research 33, 93–114.

Golalikhani, M., Oliveira, B.B., de Almeida Correia, G.H., Oliveira, J.F., Carravilla, M.A., 2024. Optimizing multi-attribute pricing plans with time-and location-dependent rates for different carsharing user profiles. Transportation Research Part E: Logistics and Transportation Review 192, 103760.

GreenMobility, 2025. The greenmobility zone. URL: https://www.greenmobility.com/dk/en/the-zone/.

Hansen, R.G., Pantuso, G., 2018. Pricing car-sharing services in multi-modal transportation systems: An analysis of the cases of copenhagen and milan, in: Computational Logistics: 9th International Conference, ICCL 2018, Vietri sul Mare, Italy, October 1–3, 2018, Proceedings 9, Springer. pp. 344–359.

Hess, S.W., Weaver, J., Siegfeldt, H., Whelan, J., Zitlau, P., 1965. Nonpartisan political redistricting by computer. Operations Research 13, 998–1006.

Huang, K., An, K., Rich, J., Ma, W., 2020. Vehicle relocation in one-way station-based electric carsharing systems: A comparative study of operator-based and user-based methods. Transportation Research Part E: Logistics and Transportation Review 142, 102081.

Illgen, S., Höck, M., 2019. Literature review of the vehicle relocation problem in one-way car sharing networks. Transportation Research Part B: Methodological 120, 193–204.

Jarrah, A.I., Bard, J.F., 2012. Large-scale pickup and delivery work area design. Computers & operations research 39, 3102–3118.

Jorge, D., Molnar, G., de Almeida Correia, G.H., 2015. Trip pricing of one-way station-based carsharing networks with zone and time of day price variations. Transportation Research Part B: Methodological 81, 461–482.

Kalcsics, J., Ríos-Mercado, R.Z., 2019. Districting problems. Location science , 705–743.

Laporte, G., Louveaux, F.V., 1993. The integer l-shaped method for stochastic integer programs with complete recourse. Operations research letters 13, 133–142.

Li, H., Hu, L., Jiang, Y., 2022. Dynamic pricing, vehicle relocation and staff rebalancing for station-based one-way electric carsharing systems considering nonlinear charging profile. Transportation Letters , 1–26.

Lu, R., Correia, G.H.d.A., Zhao, X., Liang, X., Lv, Y., 2021. Performance of one-way carsharing systems under combined strategy of pricing and relocations. Transportmetrica B: Transport Dynamics 9, 134–152.

Mehrotra, A., Johnson, E.L., Nemhauser, G.L., 1998. An optimization based heuristic for political districting. Management Science 44, 1100–1114.

Müller, C., Gönsch, J., Soppert, M., Steinhardt, C., 2023. Customer-centric dynamic pricing for free-floating vehicle sharing systems. Transportation Science 57, 1406–1432.

Novaes, A.G., de Cursi, J.S., da Silva, A.C., Souza, J.C., 2009. Solving continuous location–districting problems with voronoi diagrams. Computers & operations research 36, 40–59.

Okabe, A., Boots, B., Sugihara, K., Chiu, S.N., 2000. Spatial tessellations: concepts and applications of voronoi diagrams second edition.

Okabe, A., Suzuki, A., 1997. Locational optimization problems solved through voronoi diagrams. European journal of operational research 98, 445–456.

Pantuso, G., 2020. Formulations of a carsharing pricing and relocation problem, in: Computational Logistics: 11th International Conference, ICCL 2020, Enschede, The Netherlands, September 28–30, 2020, Proceedings 11, Springer. pp. 295–310.

Pantuso, G., 2022. Exact solutions to a carsharing pricing and relocation problem under uncertainty. Computers & Operations Research 144, 105802.

Ren, S., Luo, F., Lin, L., Hsu, S.C., Li, X.I., 2019. A novel dynamic pricing scheme for a large-scale electric vehicle sharing network considering vehicle relocation and vehicle-grid-integration. International Journal of Production Economics 218, 339–351.

Rossetti, T., Broaddus, A., Ruhl, M., Daziano, R., 2023. Commuter preferences for a first-mile/last-mile microtransit service in the united states. Transportation research part A: policy and practice 167, 103549.

Salazar-Aguilar, M.A., Ríos-Mercado, R.Z., Cabrera-Ríos, M., 2011. New models for commercial territory design.

Networks and Spatial Economics 11, 487–507.

ShareNow, 2025. What is a drop off zone? URL: https://www.share-now.com/hu/en/faq/parking/#what-is-a-drop-off-zone.

Shi, B., Lu, Y., Cao, Z., 2024. A dynamic region-division based pricing strategy in ride-hailing. Applied Intelligence 54, 11267–11280.

Soppert, M., Steinhardt, C., Müller, C., Gönsch, J., 2022. Differentiated pricing of shared mobility systems considering network effects. Transportation Science 56, 1279–1303.

Wagner, J., Falkson, L., 1975. The optimal nodal location of public facilities with price-sensitive demand. Geographical Analysis 7, 69–83.

Waserhole, A., Jost, V., 2016. Pricing in vehicle sharing systems: Optimization in queuing networks with product forms. EURO Journal on Transportation and Logistics 5, 293–320.

Xu, M., Meng, Q., Liu, Z., 2018. Electric vehicle fleet size and trip pricing for one-way carsharing services considering vehicle relocation and personnel assignment. Transportation Research Part B: Methodological 111, 60–82.

Zhang, S., Sun, H., Wang, X., Lv, Y., Wu, J., 2022. Optimization of personalized price discounting scheme for one-way station-based carsharing systems. European Journal of Operational Research 303, 220–238.

Zheng, H., Zhang, K., Nie, Y., Yan, P., Qu, Y., 2024. How many are too many? analyzing dockless bike-sharing systems with a parsimonious model. Transportation Science 58, 152–175.

Appendix A. Algorithm for solving the subproblem

We present here Algorithm 2 to solve problem (9) to optimality. Algorithm 2 first initializes

Algorithm 2 Exact computation of $Q(a, \lambda, \alpha)$

Input: a, λ, α
$\mathcal{V}^A \leftarrow \mathcal{V}$ ▷ $\mathcal{V}^A$ is the set of remaining available vehicles.
$y_{vrl} \leftarrow 0, \forall v \in \mathcal{V}, r \in \mathcal{R}, l \in \mathcal{L}_r$
$Q(a, \lambda, \alpha) \leftarrow 0$
Sort the requests $\mathcal{R}$ in non-decreasing order of customer index $k(r)$
for request $r \in \mathcal{R}$ **do**
    $L_{i(r),j(r)} = \sum_{l \in \mathcal{L}} L(l)\alpha_{i(r),j(r),l}$ ▷ The fee applied between $i(r)$ and $j(r)$.
    if $L_{i(r),j(r)} \leq l(r)$ **then**
        for $v \in \mathcal{V}^A$ **do**
            if $G_{v,i(r)} = 1$ **then**
                $\mathcal{V}^A \leftarrow \mathcal{V}^A \setminus \{v\}$ ▷ Vehicle v becomes unavailable.
                $y_{v,r,L_{i(r),j(r)}} \leftarrow 1$
                $Q(a, \lambda, \alpha) \leftarrow Q(a, \lambda, \alpha) + R^N_{r,L_{i(r),j(r)}}$ ▷ $R^N_{r,L_{i(r),j(r)}}$ is the net revenue of r at fee $L_{i(r),j(r)}$.
            end if
        end for
    end if
end for
return $Q(a, \lambda, \alpha)$, and $y_{vrl} \quad \forall v \in \mathcal{V}, r \in \mathcal{R}, l \in \mathcal{L}_r$

the set of available shared vehicles $\mathcal{V}^A$, the solution y_{vrl} and the objective value $Q(a, \lambda, \alpha)$. All requests in the set $\mathcal{R}$ are sorted in a non-decreasing order of customer index $k(r)$. Following the rule of "first-come, first-served", the algorithm iteratively checks whether the fee applied between the origin and destination is accepted and whether there is a vehicle available in the origin zone. If the fee level $L_{i(r),j(r)}$ between request r's origin station $i(r)$ and destination station $j(r)$ is lower than the highest acceptable pricing level $l(r)$ and there is at least one available vehicle v at customer's origin $i(r)$ (i.e., a vehicle with $G_{v,i(r)} = 1$), customer r will be served at pricing level $L_{i(r),j(r)}$. We then remove the used vehicle v from the available vehicles, set the value of $y_{v,r,L_{i(r),j(r)}}$ to 1, and update $Q(a, \lambda, \alpha)$.

Appendix B. Extensive MILP formulation

In this appendix, we provide the monolithic MILP formulation of problem (3) when considering the specific $Q(a, \lambda, \alpha)$ function defined in Section 7.1 as follows

$$\max_{a,\lambda,\alpha} \sum_{r \in \mathcal{R}} \sum_{v \in \mathcal{V}} \sum_{l \in \mathcal{L}_r} R^N_{rl} y_{vrl} \tag{B.1a}$$

$$\sum_{j \in \mathcal{I}} a_{ii} = S \tag{B.1b}$$

$$\sum_{j \in \mathcal{I}} a_{ij} = 1 \qquad \forall i \in \mathcal{I} \tag{B.1c}$$

$$a_{ij} \leq a_{jj} \qquad \forall i, j \in \mathcal{I} \tag{B.1d}$$

$$d(i,j_1)a_{i,j_1} \leq d(i,j_2)a_{j_2,j_2} + d(i,j_1)(1-a_{j_2,j_2}) \qquad \forall i,j_1,j_2 \in \mathcal{I} \quad \text{(B.1e)}$$

$$\sum_{l\in\mathcal{L}} \lambda_{ijl} \geq a_{ii} + a_{jj} - 1 \qquad \forall i,j \in \mathcal{I} \quad \text{(B.1f)}$$

$$\sum_{l\in\mathcal{L}} \lambda_{ijl} \leq a_{ii} \qquad \forall i,j \in \mathcal{I} \quad \text{(B.1g)}$$

$$\sum_{l\in\mathcal{L}} \lambda_{ijl} \leq a_{jj} \qquad \forall i,j \in \mathcal{I} \quad \text{(B.1h)}$$

$$a_{i_1,j_1} + a_{i_2,j_2} + \lambda_{j_1,j_2,l} \leq \alpha_{i_1,i_2,l} + 2 \qquad \forall i_1,i_2,j_1,j_2 \in \mathcal{I}, \forall l \in \mathcal{L} \quad \text{(B.1i)}$$

$$\sum_{l\in\mathcal{L}} \alpha_{ijl} = 1 \qquad \forall i,j \in \mathcal{I} \quad \text{(B.1j)}$$

$$\sum_{v\in\mathcal{V}} \sum_{l\in\mathcal{L}_r} y_{vrl} \leq 1 \qquad \forall r \in \mathcal{R} \quad \text{(B.1k)}$$

$$\sum_{r\in\mathcal{R}} \sum_{l\in\mathcal{L}_r} y_{vrl} \leq 1 \qquad \forall v \in \mathcal{V} \quad \text{(B.1l)}$$

$$\sum_{v\in\mathcal{V}} y_{vrl} \leq \alpha_{i(r),j(r),l} \qquad \forall r \in \mathcal{R}, l \in \mathcal{L}_r \quad \text{(B.1m)}$$

$$\sum_{l\in\mathcal{L}_r} y_{vrl} + \sum_{r_1\in\mathcal{R}_r} \sum_{l_1\in\mathcal{L}_{r_1}} y_{v,r_1,l} \leq G_{v,i(r)} \qquad \forall r \in \mathcal{R}, v \in \mathcal{V} \quad \text{(B.1n)}$$

$$y_{vrl} + \sum_{r_1\in\mathcal{R}_r} \sum_{l_1\in\mathcal{L}_{r_1}} y_{v,r_1,l_1} + \sum_{v_1\in\mathcal{V}\setminus\{v\}} y_{v_1,r,l} \geq \alpha_{i(r),j(r),l} + G_{v,i(r)} - 1 \qquad \forall r \in \mathcal{R}, v \in \mathcal{V}, l \in \mathcal{L}_r \quad \text{(B.1o)}$$

$$a_{ij} \in \{0,1\} \qquad \forall i,j \in \mathcal{I} \quad \text{(B.1p)}$$

$$\lambda_{ijl}, \alpha_{ijl} \in \{0,1\} \qquad \forall i,j \in \mathcal{I}, \forall l \in \mathcal{L} \quad \text{(B.1q)}$$

$$y_{vrl} \in \{0,1\} \qquad \forall r \in \mathcal{R}, v \in \mathcal{V}, l \in \mathcal{L}_r \quad \text{(B.1r)}$$

Appendix C. Notation list

In Table C.1 we summarizes the notation.

Table C.1: Notations in the zonification and pricing problem and extensive MILP formulation

Sets	
$\mathcal{I}$	Set of carsharing stations
$\mathcal{G}$	Set of generators of the tessellation and a subset of stations $\mathcal{G} \subseteq \mathcal{I}$
$\mathcal{V}(\mathcal{G})$	Collection of all subsets of station set $\mathcal{I}$ led by $\mathcal{G}$
$\mathcal{V}$,$\mathcal{U} \in \mathcal{V}(\mathcal{G})$	Subsets of stations
$\mathcal{G}^S$	Set of generators with a certain length S (i.e., the number of zones)
$\mathcal{T}$	Set of feasible discrete tessellation (or partition) solutions
$\mathcal{T}^{CGLM}$	Set of feasible discrete tessellation (or partition) solutions with (CGLM) constraints
$\mathcal{T}^{WF}$	Set of feasible discrete tessellation (or partition) solutions with (WF) constraints
$\mathcal{L}$	Set of pricing levels
$\mathcal{I}_{ij}^B$	Subset of generators that are closer to station i than generator j
$\mathcal{J}_i$	Subset of stations that are further than the $\lvert\mathcal{I}\rvert - S$-th furthest station from i
$\mathcal{K}$	Set of potential customers
$\mathcal{V}$	Set of available shared vehicles
$\mathcal{R}$	Set of requests
$\mathcal{L}_r$	Set of acceptable price levels for request $r \in \mathcal{R}$
Parameters	
$d(i,j)$	Distance between stations i and j
S	Number of zones
R_{rl}^N	Net revenue of serving request r at pricing level l
G_{vi}	Take value 1 if vehicle v is initially located at station i, 0 otherwise
L_l	Drop-off fee at price level $l \in \mathcal{L}$
Decision Variables	
a_{jj}	Binary variable indicating whether element $j \in \mathcal{I}$ is assigned as a generator
a_{ij}	Binary variable indicating whether element $i \in \mathcal{I}$ belongs to the same subset as a generator j
λ_{ijl}	Binary variable indicating whether a pricing level l is applied between generators i and j
α_{ijl}	Binary variable indicating whether a pricing level l is applied between stations i and j
y_{vrl}	Binary variable indicating whether request r is served by vehicle v at price level l