European Journal of Operational Research 308 (2023) 568–580

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

Facility location and pricing problem: Discretized mill price and exact algorithms

Yun Hui Lin[a], Qingyun Tian[b,*]

[a] Institute of High Performance Computing, Agency for Science, Technology and Research (A*STAR), Singapore
[b] School of Civil and Environmental Engineering, Nanyang Technological University, Singapore

ARTICLE INFO

Article history:
Received 2 November 2021
Accepted 29 November 2022
Available online 9 December 2022

Keywords:
Facilities planning and design
Location
Pricing
Leader-follower
Branch-and-cut

ABSTRACT

The joint optimization of facility location and service charge arises in many industrial and business contexts. This paper investigates a facility location and mill pricing problem (FLMPr), where a company aims to maximize its profit by locating service facilities and setting appropriate service charges to customers. For each facility, the number of pricing levels are finite, and the company will select exactly one level for each facility if it is open. We visualize the problem from a fully decentralized perspective, i.e., each customer acts as an independent decision-maker. Under mill pricing, customers visiting the same facility encounter the same service charge. The problem is formulated as a bilevel program, in which the company makes location and pricing decisions at the upper level, and customers decide whether to seek the service from a facility at the lower level. To solve FLMPr, we leverage three types of closest assignment constraints to reformulate the problem as mixed-integer linear programs (MILPs), which can be directly solved by modern solvers. However, this approach suffers from a time-consuming solver compiling process and cannot handle large-scale instances effectively. This observation motivates us to design a branch-and-cut algorithm by exploring the bilevel structure and deriving a feasibility cut to efficiently eliminate bilevel infeasible solutions. Our extensive experiments reveal that the proposed algorithm can solve large-scale FLMPr satisfactorily and outperforms the MILP approach by a large margin. Finally, we conduct sensitivity analysis and draw interesting observations.

1. Introduction

This paper studies a facility location and pricing problem, in which a company plans to introduce a service to customers, and the goal is to maximize the profit by locating its service facilities and setting an appropriate pricing level for each facility. The problem is visualized from a decentralized perspective, i.e., each customer acts as an independent decision-maker that decides whether to use the service from a facility based on his/her perceived cost. For the company, pricing decisions are important for its revenue. Setting a higher price allows for generating a higher unit revenue for serving a customer; however, due to the increased charge, customers may decide not to use the service, resulting in potential demand losses.

In the literature, facility pricing can be divided into two classes, depending on how transportation costs or traveling costs are passed on to customers. The first class is *delivered pricing*, under which the company bears part of or all transportation costs and sets a pairwise charge between a facility and a customer. As a result, each facility can have different prices for different customers. Based on policy types, there are three variants of delivered pricing, i.e., spatial discriminatory pricing, uniform pricing, and zone pricing. The discriminatory pricing determines an optimal price for each facility-customer pair (Hansen & Thisse, 1977). On the contrary, all customers are charged the same price under the uniform pricing policy (Hansen, Thisse, & Hanjoul, 1981). As an intermediate approach, zone pricing, originally investigated by Hansen, Peeters, & Thisse (1997), requires the company to first cluster customers into several zones and then select a price for each facility-zone pair. This policy is inherently more general than the above two. If customers are clustered into a single zone, then zone pricing reduces to the uniform pricing; whereas if each customer is treated as a zone, then there will be a specific price for each customer, and we recover the spatial discriminatory pricing. The second class is *mill pricing*, under which the traveling/transportation cost to a facility is borne by the customer (Hanjoul, Hansen, Peeters, & Thisse, 1990), and pricing is imposed on each facility. Customers visiting the same facility will be charged the same price regardless of their original locations. The total cost of a customer

* Corresponding author.
E-mail addresses: liny@ihpc.a-star.edu.sg (Y.H. Lin), qytian@ntu.edu.sg (Q. Tian).

https://doi.org/10.1016/j.ejor.2022.11.052
0377-2217/© 2022 Elsevier B.V. All rights reserved.

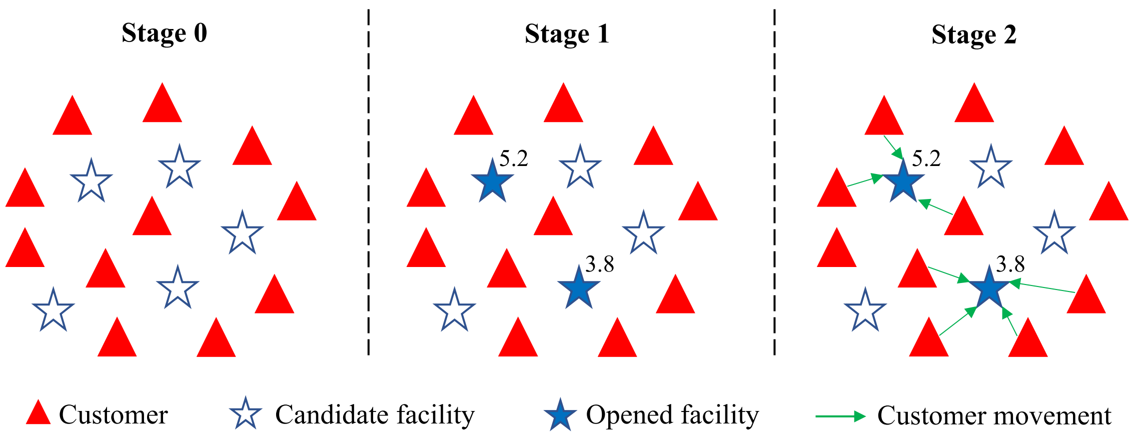


Fig. 1. Sequential decision-making process of the facility location and mill pricing problem. The numbers near the facilities represent the corresponding mill prices.

seeking services from a facility thus consists of the traveling cost and the charge specified by the mill price.

This paper focuses on mill pricing and studies a discrete version of the facility location and mill pricing problem (FLMPr). Existing works on FLMPr allow for unlimited variations in the possible value of prices by assuming that the related pricing variables are continuous. However, this assumption may lead to "ill-defined" decisions that are not realistic to implement. For example, it is certainly impractical to charge a customer with a monetary cost of 3.14159; instead, a charge of either 3.1 or 3.2 appears to be more reasonable, implying that the pricing variable should be discretized. In line with this implication, this paper considers finite numbers of pricing levels, and the company will select exactly one for each opened facility as its mill price.

Essentially, FLMPr is a "leader-follower" game where the company and the customer are the leader and the follower, respectively. Fig. 1 depicts the sequential decision process. At the initial stage, customers are assumed to be aggregated at discrete nodes with known demands. There are candidate facility sites that are available to the company. At Stage 1, the company makes "here-and-now" decisions, including both the locations of the facilities and the mill prices. In this example, two facilities are open, and the mill prices are 5.2 and 3.8. Observing the company's decision, each individual customer then determines whether to use the service from a facility based on the total cost, which consists of an accessing cost (that represents the cost of traveling to the facility) and the corresponding mill price set by the company. We assume the rational behavior of the customer, i.e., the customer will prefer the facility from which seeking the service incurs the lowest cost. Moreover, each customer has a budget. When the total cost exceeds the budget, the customer will decide not to use the service from the company. In Fig. 1, a green arrow pointing from a customer to a facility indicates that the customer decides to patronize that facility. If there is no arrow originating from a customer, then it means that this customer leaves the system or turns to other service providers. In a nutshell, a customer at Stage 2 makes a "wait-and-see" decision to minimize his/her own cost, subject to some budget limitation.

When making decisions at Stage 1, the company needs to anticipate how customers select facilities at Stage 2 because the charges can vary across facilities, resulting in different revenues for serving a customer at different sites. When a customer goes for other service providers, the related demand is lost. Therefore, the company's revenue depends on how many customers use the service (i.e., the captured demand) and how these customers are charged. Clearly, if the company opens a large number of facilities, then it can reduce the accessing cost of customers due to the increased proximity. As a result, more customers are expected to use the service, meaning that there is a trade-off between facility investment and revenue. Furthermore, lowering the mill price allows for attracting additional customers; however, the unit revenue decreases. A trade-off between pricing and captured demand thus arises. In this setting, the decision-making problem faced by the company is how to balance these trade-offs by jointly determining the facility locations and the mill prices, upon anticipating the rational behavior of the customer.

We formulate this problem as a mixed-integer bilevel program (MIBP). Owing to the bilevel structure, the resultant model is computationally challenging. In particular, exact solution approaches are not immediately available. Therefore, this paper aims to develop algorithms to solve the problem exactly and efficiently so that the proposed modeling framework can be practically implementable.

1.1. Literature review

The facility location and pricing problem arises in many industrial and business applications such as postal service industry, chained retail store, and distribution systems for fruits, leather, and other products. In the literature, the pricing strategy can be divided into two classes: delivered pricing and mill pricing (Hansen et al., 1997).

Delivered pricing consists of three policy types. Among them, uniform pricing is the most simple one as it determines a unified charge for all customers. Aboolian, Berman, & Krass (2008) optimized the location and uniform pricing decisions for a company operating in a competitive environment, where there exist competing facilities, and customers will patronize the facility with the highest utility. A mixed-integer nonlinear program was proposed and solved by a specialized linearization approach. Berger, Grigoriev, Panin, & Winokurow (2017) studied the uniform pricing on the Euclidean plane and investigated the computational complexity. Discriminatory pricing, as an extreme case of delivered pricing, requires the company to set a price for each facility-customer pair. Such a strategy can generate the highest possible profit for the company but, in some sense, violates the anti-trust legislation and harms the interest of customers. The related research under discriminatory pricing typically assumed that there exist one or several competing companies. Pelegrín, Fernández, Suárez, & García (2006) investigated a single facility location and pricing problem on a network and explored the Nash equilibrium in price. Pelegrín, Fernández, Pérez, & Hernández (2012) studied a chain expansion problem and incorporated the cannibalization effect resulting from the opening of new facilities. They showed that the chain can achieve profit maximization by locating facilities at the so-called "iso-marginal" points, i.e., points from which the marginal delivered cost is the same as the minimum marginal delivered cost from the existing facilities. Using this finding, they reduced the problem to a discrete optimization problem that can be formulated as a mixed-integer linear program (MILP). Fernández, Salhi, Boglárka et al. (2014) considered the equilibria for a competitive single facility location problem on the plane. Two companies exist in a market, and each company is to locate one facility and determine its discriminatory pricing. For small- and medium-scale problems, an exact interval branch-and-bound algorithm was proposed to solve them optimally; whereas for large-scale problems, an alternating Weiszfeld-like heuristic was used to quickly generate quality solutions. In a similar setting, Panin, Pashchenko, & Plyasunov (2014) investigated a multiple-facility problem and formulated a bilevel program that was solved by approximation algorithms leveraging alternating heuristics and local search. As a general policy, zone pricing has been investigated by Hansen et al. (1997) under the assumption of price-sensitive demand. The resulting model was mixed-integer nonlinear program (MINLP) and tackled optimally by a branch-and-bound algorithm, derived by projecting the original problem onto the pricing space. Recently, facility location and pricing problems under delivered pricing have been extended to consider other issues such as (i) different demand splitting rules between competing companies (Kononov, Panin, & Plyasunov, 2018; 2019) and (ii) routing decisions at the operational level, leading to joint location-routing-pricing problems (Ahmadi-Javid, Amiri, & Meskar, 2018; Dastaki, Setak, & Karimi, 2021).

To date, the facility location and mill pricing problem (FLMPr) has received considerable research attention. From the computational point of view, FLMPr is generally more challenging than its counterpart under delivered pricing. In one particular research stream, random utility theory is employed to predict customer behaviors. More specifically, the choices of customers (regarding whether to use the service and which facility to seek the service from) are interpreted probabilistically and forecasted using discrete choice models such as the multinominal logit model (MNL). It is assumed that the pricing decision affects the utility of the customer and thus the choice probability (Kress & Pesch, 2016; Zambrano-Rey, López-Ospina, & Pérez, 2019; Zhang, 2015). For example, Zhang (2015) studied the network design problem of a chained retail store in a competitive environment. The utility of a facility to a customer was mainly determined by the mill price and the traveling cost. The MNL was applied to estimate the flows of customers traveling to stores. The problem was formulated as a MINLP and solved by a two-phase heuristic. Kress & Pesch (2016) visualized a location and pricing problem on networks as a "location-then-price" game under the MNL and considered Nash equilibrium in prices among competing companies. A fixed-point iteration approach was used to quickly search for local price equilibria. Recently, we have seen new FMLPr models incorporating the queueing or congestion effects (Dan, Lodi, & Marcotte, 2020; Hajipour, Farahani, & Fattahi, 2016) and considering more levels of interactions between competing companies and customers (Arbib, Pınar, & Tonelli, 2020).

In the other research stream, customer choices are considered to be deterministic and follow the "all-or-nothing" choice rule. Such a rule postulates the rational behavior of customers, i.e., each customer will prefer the facility from which seeking the service incurs the lowest total cost. Our paper belongs to this stream. As such, the most relevant works are Diakova & Kochetov (2012) and Kochetov, Panin, & Plyasunov (2015), where bilevel optimization models were proposed to reflect the sequential decision-making process of the company and the customer. Unfortunately, it is computationally challenging to solve the bilevel models. In effect, even when the location decision has been made, determining the optimal mill price on each facility remains NP-hard in general (Plyasunov & Panin, 2013b). As a consequence, the existing literature relied heavily on heuristic approaches. For example, Diakova & Kochetov (2012) proposed a double variable neighborhood search (VNS) heuristic to solve a FLMPr problem. This heuristic was later used as a benchmark method to a standard testbed for *Facility Location and Pricing* in the *Discrete Location Problems Library*, together with a hybrid heuristic that combines VNS and simulated annealing. In Kochetov et al. (2015), several hybrid algorithms based on local search and metaheuristics were presented to compare with the computational results of CPLEX. Our work is also closely related to Ahmadi-Javid and Ghandali (2014), in which discretized pricing was considered under facility capacity restriction. The problem was formulated as a MILP and solved by a Lagrangian relaxation algorithm to obtain quality bounds and solutions. However, exact solution approaches that guarantee finding the optimal solution have not been well-investigated. Motivated by this observation, we focus on developing exact algorithms that are capable of solving industrial scale instances optimally in a reasonable computational time.

1.2. Contribution and structure

Positioning within the literature of FLMPr, our contributions are as follows. Firstly, we study a FLMPr problem, where for each opened facility, there are a finite number of candidate pricing levels, and exactly one level should be selected by the company as the corresponding mill price. Under the rational customer behavior assumption, we formulate the problem as a bilevel program. To the best of our knowledge, discrete mill pricing has not been studied thus far, despite that it is practically reasonable. Secondly, we provide straightforward and easy-to-implement methods to reformulate the problem as three MILP models, which can be directly solved by commercial solvers to obtain the proven optimal solution. However, the MILP approaches suffer from a dramatic increase in the formulation size, rendering them computational prohibitive even when the problem scale is only moderate. Since the scale of real industrial problems faced by practitioners could be large, more efficient solution approaches are needed. Our third contribution thus consists of an efficient branch-and-cut algorithm, which leverages a specialized bilevel feasibility cut. Through extensive computational experiments, we demonstrate that the proposed algorithm is significantly more efficient than the MILP approaches and is capable of solving large-scale instances satisfactorily. Finally, we conduct sensitivity analysis and provide managerial implications.

The remaining of the paper is structured as follows. We introduce the problem as well as the formulation for FLMPr in Section 2. Subsequently, we present the MILP reformulation approaches in Section 3. Section 4 develops an efficient branch-and-cut algorithm. We then conduct extensive numerical experiments in Section 5. Finally, Section 7 concludes the paper and points out potential future research.

2. Problem description and formulation

In this section, we present the problem description and the model formulation for the facility location and mill pricing problem. We summarize the main notations in Table 1, and additional notations will be introduced whenever necessary.

Table 1
Notations.

Sets		
I	:	set of customers.
J	:	set of candidate facilities.
K_j	:	set of candidate pricing levels for facility j, $\forall j \in J$.
Γ	:	set of indices for all location-pricing pairs. $\Gamma = \{(j,k) \mid j \in J, k \in K_j\}$
Parameters		
b_i	:	budget of customer i, $\forall i \in I$.
d_i	:	demand of customer i, $\forall i \in I$.
f_j	:	setup cost of facility j, $\forall j \in J$.
p_{jk}	:	price (charge) at pricing level k on facility j, $\forall j \in J, k \in K_j$.
c_{ij}	:	accessing cost of customer i to facility j, $\forall i \in I, j \in J$.
θ_{ijk}	:	total cost of customer i visiting facility j at pricing level k. $\theta_{ijk} = c_{ij} + p_{jk}, \forall i \in I, j \in J, k \in K_j$.
π_{ijk}	:	surplus of customer i visiting facility j at pricing level k. $\pi_{ijk} = b_i - \theta_{ijk}, \forall i \in I, j \in J, k \in K_j$.
Variables		
w_j	:	binary. 1, if facility j is open; 0, otherwise, $\forall j \in J$.
x_{jk}	:	binary. 1, if facility j is open at pricing level k, 0, otherwise, $\forall j \in J, k \in K_j$.
y_{ijk}	:	binary. 1, if customer i patronizes facility j at pricing level k; 0, otherwise, $\forall i \in I, j \in J, k \in K_j$.

2.1. Decision stage

Fig. 1 explains the sequential decision process of the company and the customer. In brief, at Stage 1, the company makes "here-and-now" location and pricing decisions to maximize the profit, anticipating that customers will minimize their own cost by selecting the best "wait-and-see" decision at Stage 2. We use set I, set J, set K_j to denote the sets of customers, facilities, and pricing levels for facility j, respectively. For ease of exposition in our later discussion, we define an additional set Γ as the set of indices for all location-pricing pairs, i.e., $\Gamma = \{(j,k) \mid j \in J, k \in K_j\}$.

Stage 1. The company decides the location of facilities and the charges to customers for using the services. To model the location decision, we define a binary variable $w_j, \forall j \in J$, which is 1 if facility j is open; 0, otherwise. There are finite numbers of mill pricing levels for each facility. Specifically, we use a parameter p_{jk} to reflect the charge on facility j at pricing level k, $\forall j \in J, k \in K_j$. Define a binary variable x_{jk} such that it is 1 if facility j is open at pricing level k; 0, otherwise. Then the actual mill price for facility j is $\sum_{k \in K_j} p_{jk} x_{jk}$. Meanwhile, the location and pricing variables should satisfy

$$\sum_{k \in K_j} x_{jk} = w_j \quad \forall j \in J \tag{1}$$

which enforces that when a facility is opened, one pricing level should be selected as the mill price.

Stage 2. After facilities are built and the mill prices are set at Stage 1, customers then determine whether to use the service from the company and, subsequently, which facility to patronize at Stage 2. Following the standard assumption in the literature of facility mill pricing problems (Diakova & Kochetov, 2012; Kochetov et al., 2015; Plyasunov & Panin, 2013a), we assume that customer i has a cost budget b_i, and if customer i uses facility j, then he/she incurs an accessing cost c_{ij} that represents the cost of customer i traveling to facility j. Together with the mill price, the total cost of customer i seeking services from facility j at pricing level k is given by $\theta_{ijk} = c_{ij} + p_{jk}$, i.e., the summation of the accessing cost and the charge.

When multiple facilities are open, customer i will only consider the one with the lowest total cost. Moreover, if the resultant lowest cost exceeds budget b_i, then customer i will not use the service from the company. Define a binary variable y_{ijk} such that $y_{ijk} =$ 1 if customer i uses facility j at pricing level k, $\forall i \in I, (j,k) \in \Gamma$. Now, given the location and pricing decisions (w,x), the choices of customers can be modeled by

$$\text{argmin}_y \sum_{i \in I} \sum_{(j,k) \in \Gamma} (\theta_{ijk} - b_i) y_{ijk} \tag{2a}$$

$$\text{st.} \sum_{(j,k) \in \Gamma} y_{ijk} \le 1 \qquad \forall i \in I \tag{2b}$$

$$y_{ijk} \le x_{jk} \qquad \forall i \in I, (j,k) \in \Gamma \tag{2c}$$

$$y_{ijk} \in \{0,1\} \qquad \forall i \in I, (j,k) \in \Gamma \tag{2d}$$

where Constraint (2b) enforces that each customer will visit at most one facility. Constraint (2c) is a logic constraint imposing that y_{ijk} can be 1 only if facility j is opened at pricing level k. In the objective function, when $\theta_{ijk} - b_i > 0$, we will have $y_{ijk} = 0$ in the optimal solution since we are minimizing the objective and should avoid adding a positive value to it. Therefore, y_{ijk} can take the value of 1 only if the total cost of customer i visiting facility j at pricing level k is not more than his/her budget b_i, i.e., $\theta_{ijk} - b_i \le 0$. Moreover, if there exist facilities that could be selected by customers, then the facility with the lowest total cost to customer i will be chosen and the corresponding y variable will be 1 in the optimal solution.

In a nutshell, Problem (2) effectively models the choice of the customer at Stage 2. We refer to Problem (2) as the *lower-level problem*.

2.2. Model formulation

Given a solution y from Problem (2), the company computes the revenue by $\sum_{i \in I} \sum_{(j,k) \in \Gamma} d_i p_{jk} y_{ijk}$. The ultimate goal is to maximize the profit, accounting for the revenue and the cost of facilities; therefore, the complete decision problem faced by the company is

$$\max \sum_{i \in I} \sum_{(j,k) \in \Gamma} d_i p_{jk} y_{ijk} - \sum_{j \in J} f_j w_j \tag{3a}$$

$$\textbf{[FLMPr]} \quad \text{st.} \sum_{k \in K_j} x_{jk} = w_j \qquad \forall j \in J \tag{3b}$$

$$w_j \in \{0,1\} \qquad \forall j \in J \tag{3c}$$

$$x_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K_j \tag{3d}$$

$$y \in (2) \tag{3e}$$

where Constraint (3e) expresses y as an optimal solution to Problem (2). We refer to the above problem as *Facility Location and Mill Pricing Problem*, abbreviated as [FLMPr]. Indeed, [FLMPr] is a *bilevel program*, where the location and pricing decisions of the company (x,w) are the upper-level variables, and the customer choice y is the lower-level variable. Since all decision variables are binary, an optimal solution to this bilevel program exists and is attainable (Vicente, Savard, & Judice, 1996).

To simplify the formulation in our later discussion, we define the following sets

$$\Omega^L = \{(x,y) \mid (2b) - (2d)\}$$
$$\Omega = \{(w,x,y) \mid (2b) - (2d), (3b) - (3d)\}$$

where Ω^L is the feasible region of the lower-level variables. Ω contains all constraints in [FLMPr] except for the embedded minimization in the lower-level problem (2); therefore, Ω is the joint decision space of the company and the customer.

In the context of bilevel optimization, we assume the *optimistic* strategy of the customer, namely, given a location and pricing decisions of the company (w, x), if there exists more than one optimal solution to the lower-level problem (2), then the customer will select the one that benefits the company, which is a common assumption in facility pricing problems (Arbib et al., 2020; Diakova & Kochetov, 2012; Kochetov et al., 2015; Plyasunov & Panin, 2013a). As an illustration, assume that there is only 1 pricing level, and facility j and facility l have the same total cost, i.e., $p_j + c_{ij} = p_l + c_{il}$. If $p_j > p_l$, then customer i will prefer facility j. In this setting, customers are implicitly assumed to be more sensitive to the accessing cost (note that $c_{ij} < c_{il}$) and are willing to bear a higher service charge to reduce it. Moreover, if $b_i = p_j + c_{ij}$, i.e., the cost of choosing facility j equates to the budget, then the customer will prefer using the service from the company rather than going for the outside option.

3. Mixed-integer linear programing approach

Due to the bilevel structure, exact solution approaches for [FLMPr] are not immediately available. Fortunately, under the optimistic strategy, there exist several single-level reformulation methods that can subtly recast [FLMPr] into mixed-integer linear programs (MILPs). The idea is to replace the minimization of (2a) in the lower-level problem by modified *closest assignment constraints* (CACs). These constraints have been employed in the facility location literature to guarantee that customers will be "assigned" to the option with the lowest cost (Berman, Drezner, Tamir, & Wesolowsky, 2009; Espejo, Marín, & Rodríguez-Chía, 2012). In this paper, we discuss three variants of CACs, with which we derive three MILP formulations for [FLMPr].

3.1. MILP formulation 1

We first look at the following CAC.

$$\sum_{(m,n)\in\Gamma} \theta_{imn} y_{imn} \le \theta_{ijk} x_{jk} + b_i(1 - x_{jk}) \quad \forall i \in I, (j,k) \in \Gamma \tag{4}$$

which is equivalent to (2a). To see why this idea works, suppose that not all facilities will be open in the optimal solution (in the case where all facilities are open, the location decision itself is trivial). If $x_{jk} = 0$, then the right hand side of (4) become b_i, and we have $\sum_{(m,n)\in\Gamma} \theta_{imn} y_{imn} \le b_i$. In this case, for facility m at pricing level n such that $\theta_{imn} > b_i$, y_{imn} will be equal to 0 so as to satisfy the constraint. On the other hand, if $x_{jk} = 1$, then we have $\sum_{(m,n)\in\Gamma} \theta_{imn} y_{imn} \le \theta_{ijk}$. This constraint stipulates that if $\theta_{imn} > \theta_{ijk}$, then $y_{imn} = 0$, and that y_{imn} can be 1 only if $\theta_{imn} \le \theta_{ijk}$. Combining these cases, Eq. (4), together with $y \in \Omega^L$, represents the choice of the customer as described in the lower-level problem (2).

With the above inequality, we can reformulate [FLMPr] as the following MILP

$$\max \sum_{i\in I} \sum_{(j,k)\in\Gamma} d_i p_{jk} y_{ijk} - \sum_{j\in J} f_j w_j \tag{5a}$$

[MILP-1]

$$\text{st.} \sum_{(m,n)\in\Gamma} \theta_{imn} y_{imn} \le \theta_{ijk} x_{jk} + b_i(1 - x_{jk}) \quad \forall i \in I, (j,k) \in \Gamma \tag{5b}$$

$$(w, x, y) \in \Omega \tag{5c}$$

which is ready to be solved using off-the-shelf MILP solvers such as Gurobi.

3.2. MILP formulation 2

We then discuss another CAC. Specifically, consider the following MILP

$$\max \sum_{i\in I} \sum_{(j,k)\in\Gamma} d_i p_{jk} y_{ijk} - \sum_{j\in J} f_j w_j \tag{6a}$$

[MILP-2]

$$\text{st. } y_{imn} \le 1 - x_{jk} \quad \text{if } \theta_{imn} > \theta_{ijk} \quad \forall i \in I, (m,n) \in \Gamma, (j,k) \in \Gamma \tag{6b}$$

$$y_{ijk} = 0 \quad \text{if } \theta_{ijk} > b_i \quad \forall i \in I, (j,k) \in \Gamma \tag{6c}$$

$$(w, x, y) \in \Omega \tag{6d}$$

in which the minimization of (2a) in the lower-level problem is restated as Constraints (6b) and (6c). Essentially, Constraint (6b) imposes that for customer i, if the cost of an option (m, n) is higher than another option (j, k), then this option will never be chosen by customer i when we open facility j with pricing level k. Therefore, customer i will only consider the available option with the lowest total cost. Moreover, when an option results in a total cost that exceeds the customer's budget (i.e., $\theta_{ijk} > b_i$), this option will not be chosen no matter whether it is available or not. This condition is imposed in Constraint (6c). As a result, we conclude that Constraints (6b) and (6c) are equivalent to Constraint (5b).

3.3. MILP formulation 3

One potential issue of [MILP-2] is that (6b) contains up to $O(|I| \cdot |\Gamma| \cdot |\Gamma|)$ numbers of constraints. The formulation size thus increases dramatically, which could significantly impede the solution process. One way to mitigate this issue is to use a more compact inequality to replace (6b), giving rise to our third MILP formulation.

$$\max \sum_{i\in I} \sum_{(j,k)\in\Gamma} d_i p_{jk} y_{ijk} - \sum_{j\in J} f_j w_j \tag{7a}$$

[MILP-3]

$$\text{st.} \sum_{(m,n):\theta_{imn}>\theta_{ijk}} y_{imn} \le 1 - x_{jk} \quad \forall i \in I, (j,k) \in \Gamma \tag{7b}$$

$$y_{ijk} = 0 \quad \text{if } \theta_{ijk} > b_i \quad \forall i \in I, (j,k) \in \Gamma \tag{7c}$$

$$(w, x, y) \in \Omega \tag{7d}$$

where Constraint (7b) imposes that if option (j, k) is available (i.e., $x_{jk} = 1$), then all other options that incur higher costs will not be considered by the customer (i.e., the left hand side must be zero). This recovers the logic that customers will minimize their costs. Eventually, only the option with the lowest cost among all available options will possibly be chosen by the customer as inferred by Constraint (7b).

Remark. The above approaches find the optimal solution to [FLMPr], provided that the resultant MILP models can be solved optimally. However, these models themselves are challenging optimization problems. The reformulation process introduces a large number of constraints. When the problem scale is large, the formulation size could become too large to be effectively handled. It turns out that when we apply Gurobi to solve median-scale instances (e.g., 100 customers, 50 candidate facilities, and 20 candidate pricing levels for each facility), it may take several minutes for the solver under the Python-API to compile the model before the internal process solution starts. In particular, our preliminary experiment shows that the solver compiling time can exceed 2 hours in some of our testbeds.

4. Bilevel branch-and-cut algorithm

Noticing the drawbacks of the MILP approaches, this section presents an efficient algorithm to expedite solving [FLMPr]. This algorithm explores the bilevel structure of the problem and derives a specialized feasibility cut that can be easily implemented within the branch-and-cut framework of modern solvers.

4.1. Overall framework

We first look at the following result.

Lemma 1. *If $(w, x, y) \in \Omega$, then the lower-level problem (2) can be rewritten as*

$$\text{argmax}_y \sum_{i\in I} \sum_{(j,k)\in\Gamma} \pi_{ijk} y_{ijk} x_{jk} \tag{8a}$$

$$\text{st.} \sum_{(j,k)\in\Gamma} y_{ijk} \le 1 \qquad \forall i \in I \tag{8b}$$

$$y_{ijk} \in \{0, 1\} \qquad \forall i \in I, (j, k) \in \Gamma \tag{8c}$$

where $\pi_{ijk} = b_i - \theta_{ijk}, \forall i \in I, (j, k) \in \Gamma$.

Proof. Consider the objective function (8a). When $x_{jk} = 0$, y_{ijk} will be 0 because the condition $y_{ijk} \le x_{jk}$ is included in Ω. When $x_{jk} = 1$, y_{ijk} can be 1 only if $\pi_{ijk} \ge 0$, i.e., $\theta_{ijk} \le b_i$. To maximize the objective, the y_{ijk} corresponding to the largest π_{ijk} will be 1 in the optimal solution, meaning that the option with the lowest cost (i.e., the smallest θ_{ijk}) will be selected by the customer, which is exactly the lower-level problem (2) □

Essentially, π_{ijk} can be seen as the "surplus" of facility j at pricing level k over the outside option, and Problem (8) can be interpreted as that *customers will maximize the surplus through selecting the option with the lowest cost*. The objective function is in a bilinear form with the term $y_{ijk} x_{jk}$, which will be useful to derive a specialized bilevel cut. To facilitate our discussion, define set $\widehat{\Omega}^L$ as the feasible region of Problem (8), i.e.,

$$\widehat{\Omega}^L = \{y \mid ((8b) - (8c)\}$$

which is independent of the upper-level variables x and w.

To proceed, we call a solution $(w, x, y) \in \Omega$ *bilevel feasible* if given a company's decision (w, x), y is the optimal solution to Problem (8). That is, y reflects the best choices of the customers under the location and pricing decisions (w, x). Based on this definition, a solution is feasible for [FLMPr] if and only if it is bilevel feasible. For the company, to maximize the profit is to search for a bilevel feasible solution that maximizes the objective function in [FLMPr].

To this end, we explicitly give a condition for the bilevel feasibility.

Lemma 2. *A solution $(w, x, y) \in \Omega$ is bilevel feasible if and only if*

$$\sum_{(j,k)\in\Gamma} \pi_{ijk} y_{ijk} \ge \sum_{(j,k)\in\Gamma} \pi_{ijk} y^s_{ijk} x_{jk} \quad \forall i \in I, y^s \in \widehat{\Omega}^L \tag{9}$$

Proof. Based on Ye (2006), a bilevel feasible condition for a solution $(w, x, y) \in \Omega$ to [FLMPr] is

$$\sum_{i\in I} \sum_{(j,k)\in\Gamma} \pi_{ijk} y_{ijk} x_{jk} \ge \sum_{i\in I} \sum_{(j,k)\in\Gamma} \pi_{ijk} y^s_{ijk} x_{jk} \quad \forall y^s \in \widehat{\Omega}^L \tag{10}$$

That is, for every $y^s \in \widehat{\Omega}^L$, if the above condition holds, then y leads to the maximum value of the objective given the location and pricing decisions (w, x). Therefore, y is an optimal solution to Problem (8) and thus, (w, x, y) is bilevel feasible. Moreover, in the left hand side of (10), the bilinear term $y_{ijk} x_{jk}$ can be replaced by x_{jk} because the condition $y_{ijk} \le x_{jk}$ is included in Ω, which has prevent y_{ijk} from taking the value of 1 if $x_{jk} = 0$. Finally, both inequality (10) and $\widehat{\Omega}^L$ are separate for each customer i; therefore, we can impose up to $|I|$ constraints (i.e., one for each customer) at once for each y^s, giving rise to the formulation in (9). □

For each y^s, Constraint (9) can be seen as a disaggregated *bilevel feasible cut*, which cuts off the bilevel infeasible solutions to recover bilevel feasibility. With the cut, [FLMPr] can be equivalently restated as the following single-level problem

$$\max \sum_{i\in I} \sum_{(j,k)\in\Gamma} d_i p_{jk} y_{ijk} - \sum_{j\in J} f_j w_j \tag{11a}$$

$$\text{st.} \sum_{(j,k)\in\Gamma} \pi_{ijk} y_{ijk} \ge \sum_{(j,k)\in\Gamma} \pi_{ijk} y^s_{ijk} x_{jk} \quad \forall i \in I, y^s \in \widehat{\Omega}^L \tag{11b}$$

$$(w, x, y) \in \Omega \tag{11c}$$

Theoretically, solving Problem (11) yields an optimal solution to [FLMPr]. However, we need to enumerate all feasible points in $\widehat{\Omega}^L$ (and thus all bilevel feasible cuts) as indicated in (11b). The size of $\widehat{\Omega}^L$ is exponentially large; therefore, it can be computationally prohibitive to directly solve Problem (11) even for medium-scale problems.

To circumvent handling the large constraint size, we design a mechanism to maintain only a manageable number of bilevel feasible cuts at the beginning of the solution process and then generate y^s on-the-fly to allow for adding back bilevel feasible cuts as needed. Specifically, we consider a relaxation of Problem (11) by choosing set Ξ such that $\Xi \subset \widehat{\Omega}^L$ and use the following relaxed problem

$$\max \sum_{i\in I} \sum_{(j,k)\in\Gamma} d_i p_{jk} y_{ijk} - \sum_{j\in J} f_j w_j \tag{12a}$$

$$\text{st.} \sum_{(j,k)\in\Gamma} \pi_{ijk} y_{ijk} \ge \sum_{(j,k)\in\Gamma} \pi_{ijk} y^s_{ijk} x_{jk} \quad \forall i \in I, y^s \in \Xi \tag{12b}$$

$$(w, x, y) \in \Omega \tag{12c}$$

Since y^s is not fully enumerated, Problem (12) is a relaxation of Problem (11) and thus, solving Problem (12) gives us an upper bound to [FLMPr]. Moreover, given any (w, x), a new y^s can be generated by solving the equivalent lower-level problem (8). This y^s then defines a set of bilevel feasible cuts, which can be inserted into Problem (12) to cut off bilevel infeasible solutions and tighten the bound. Through subsequently generating the location and pricing decisions (w, x) and the customer choice y^s, we can obtain a sufficient number of bilevel feasible cuts to guarantee that we select the optimal bilevel feasible solution (w^*, x^*, y^*) to [FLMPr].

4.2. Generating bilevel feasible cut

Central to the above idea is how we can efficiently generate bilevel feasible cuts. This involves solving Problem (8), under the condition that the customer follows the optimistic strategy, to obtain the actual choice of the customer y^s. Note that customers will choose the option with the lowest cost among all available alternatives (i.e., all opened facilities plus the outside option). We can thus solve Problem (8) analytically. Specifically, given $\bar{x}$, we define the set $\bar{\Psi}$ such that

$$\bar{\Psi} = \{(j,k) \mid \bar{x}_{jk} = 1, \forall (j,k) \in \Gamma\} \tag{13}$$

For each customer i, we use (m_i, n_i) to represent *the index of the most preferable facility with its pricing level of customer i*. Two cases arise.

Case 1. Customer i has a unique optimal choice that minimizes his/her total cost. In this case, the index (m_i, n_i) is such that $\theta_{im_in_i} < \theta_{ijk}, \forall (j,k) \in \bar{\Psi} \setminus \{(m_i, n_i)\}$. In other words, (m_i, n_i) is the unique index of the lowest-cost facility and its mill price, i.e.,

$$(m_i, n_i) = argmin_{(j,k)\in\bar{\Psi}} \ \theta_{ijk} \tag{14}$$

Case 2. Customer i has multiple optimal choices that minimize his/her total cost. In this case, there exist multiple indices such that, for an index (M_i, N_i), we have $\theta_{iM_iN_i} \le \theta_{ijk}, \forall (j,k) \in \bar{\Psi} \setminus \{(M_i, N_i)\}$. Let $\bar{O}_i$ be the set of indices included in the optimal solutions. Assuming the optimistic bilevel formulation, we have

$$(m_i, n_i) = argmax_{(j,k)\in\bar{O}_i} \ p_{jk} \tag{15}$$

i.e., the option with the highest charge is preferred by customer i.

Given (m_i, n_i) from these two cases, y^s can be obtained by

$$y^s_{ijk} = \begin{cases} 1 & \text{if } (j,k) = (m_i, n_i) \text{ and } \theta_{ijk} \le b_i \\ 0 & \text{otherwise} \end{cases} \quad \forall (j,k) \in \Gamma \tag{16}$$

We can then generate bilevel feasible cuts as in (12b).

4.3. Preprocessing

Similar to Constraint (6c) in [MILP-2], if $\theta_{ijk} > b_i$ (or equivalently, $\pi_{ijk} < 0$), then the option (j,k) will definitely not be chosen by customer i. We can thus impose this constraint so that the solver can remove redundant variables by its internal presolve function. In many cases, this can significantly reduce the problem size and the number of bilevel feasible cuts required to reach optimality.

Moreover, unlike the MILP approach, the binary variable y in Problem (12) can be relaxed to $y \in \mathbb{R}_+$ since the formulation has automatically enforced the integrality of y_{ijk}. In our experiment, we observe that relaxing y in general leads to faster computation.

Finally, we add the following restriction to the model

$$\sum_{j\in J} w_j \ge 1 \tag{17}$$

which enforces that at least one facility is opened to avoid the empty set of $\bar{\Psi}$. The case where $\sum_{j\in J} w_j = 0$ simply means that no facility is open, and the company eventually decides not to provide any service. Such a scenario can be easily identified if the resulting profit is negative after adding (17).

4.4. Summary of implementation

We now describe how we implement the above idea using the branch-and-cut framework within modern MILP solvers. In this paper, we use Gurobi 9.1.2 to design our algorithm. We rely on the default branching rule and the internal cutting planes of Gurobi. We initialize Problem (12) without any bilevel feasible cut, i.e., $\Xi = \emptyset$, and start the branch-and-cut algorithm with the preprocessing technique in Section 4.3. In the searching tree, when the location and pricing decisions $(\bar{w}, \bar{x})$ from the current LP relaxation are integer (i.e., when the solver visits integer nodes), we read the solution $(\bar{w}, \bar{x}, \bar{y})$. Then, we solve Problem (8) to obtain the optimal customer choices y^s under $\bar{x}$ to generate bilevel feasible cuts as described in Section 4.2. Now, if the current solution $(\bar{w}, \bar{x}, \bar{y})$ violates the resultant bilevel feasible cut for some customer i, i.e.,

$$\sum_{(j,k)\in\Gamma} \pi_{ijk}\bar{y}_{ijk} < \sum_{(j,k)\in\Gamma} \pi_{ijk}y^s_{ijk}\bar{x}_{jk} \tag{18}$$

then it means that this solution is not bilevel feasible since $\bar{y}$ does not lead to the maximum objective of Problem (8) under $(\bar{w}, \bar{x})$. In this case, we add the corresponding cut into the searching tree to refine bilevel feasibility using the *lazy-cut callback* function of Gurobi.

5. Numerical study

This section presents computational studies to demonstrate the efficiency of the proposed algorithms as well as conducts sensitivity analysis to draw insightful observations.

5.1. Computational experiment

We refer to the bilevel branch-and-cut algorithm proposed in Section 4 as BBC. We also test the performance of the MILP approaches, where the MILP formulations presented in Section 3 are directly solved by Gurobi. We refer to them as MILP-1, MILP-2, and MILP-3. Moreover, we also add Constraint (6c) to MILP-1. All experiments are programmed using Python on a 16 GB memory macOS computer with a 2.6 GHz Intel Core i7 processor. Gurobi version is 9.1.2, and the solver parameters are under default settings, except that, for BBC, *lazyConstraints* is set to 1 to activate the callback function. Finally, for parameter simplicity and without loss of generality, we assume that $K = K_j, \forall j \in J$, and $f = f_j, \forall j \in J$.

Below show the main notations that will be used in our discussion.

- $ST[s]$: the solver computational time in seconds, which stands for the actual time that Gurobi solves an instance.
- $TT[s]$: the total run time in seconds, which consists of the solver setup time and the solver computational time $ST[s]$. The solver setup time represents the compiling time of the input model to Gurobi. It turns out that when the MILP approaches are applied to solve [FLMPr], Gurobi may require a significant time to compile the model before starting the solution process; therefore, the model may remain unsolved even when the time limit has been reached.
- $rgap$[%]: the relative optimality gap in percentages, computed by $|zbb - zopt|/|zbb| \times 100$, where $zopt$ is the value of the optimal integer solution found at termination, and zbb is the current best bound. An instance is solved optimally if we have $zbb = zopt$ or $rgap < 0.01\%$.

5.2. FLPr dataset

We start by testing the performance of the solution approaches using a standard testbed for *Facility Location and Pricing Problem* (*FLPr*) from *Discrete Location Problems Library*[1]. There are 20 structured instances, labeled as $A - B - C$, where A and B present the number of customers and the number of facilities, respectively, and C is the instance number. For example, $100 - 40 - 4$ means the 4th instance in the dataset with 100 customers and 40 facilities. Furthermore, in these problems, exactly 5 facilities are to be opened (i.e., the problems in the library consider locating 5 facilities and determining their optimal mill prices to maximize the revenue); therefore, we slightly modify our model by removing $-\sum_{j\in J} f_j w_j$ from the objective function and adding $\sum_{j\in J} w_j = 5$ to

[1] We recode the dataset using Python. Files can be provided upon request.

Table 2
Computational results of FLPr instances. Time limit is 7200 seconds.

$\|K\|$	instance	MILP-1			MILP-2			MILP-3			BBC		
		TT[s]	ST[s]	rgap[%]	TT[s]	ST[s]	rgap[%]	TT[s]	ST[s]	rgap[%]	TT[s]	ST[s]	rgap[%]
10	100-40-01	515.7	116.0	0	233.8	150.2	0	78.3	62.8	0	47.3	45.1	0
	100-40-02	526.7	151.3	0	282.7	200.4	0	60.1	46.0	0	134.0	131.6	0
	100-40-03	462.2	91.6	0	140.4	57.8	0	92.6	78.1	0	41.7	39.4	0
	100-40-04	436.4	47.7	0	122.4	37.3	0	34.8	20.6	0	22.0	19.8	0
	100-40-05	546.2	169.9	0	409.5	329.1	0	207.9	193.8	0	163.3	161.1	0
	100-40-06	444.4	74.4	0	187.2	108.0	0	51.6	37.7	0	38.1	35.8	0
	100-40-07	510.0	133.0	0	379.8	300.6	0	57.0	43.1	0	108.5	106.3	0
	100-40-08	532.1	145.6	0	377.3	298.2	0	189.2	175.4	0	136.2	133.6	0
	100-40-09	506.2	103.0	0	301.2	217.8	0	71.0	57.3	0	53.9	51.6	0
	100-40-10	536.6	137.0	0	341.2	256.9	0	156.4	142.4	0	219.6	217.4	0
	Average	**501.7**	**117.0**	**0**	**277.5**	**195.6**	**0**	**99.9**	**85.7**	**0**	**96.5**	**94.2**	**0**
20	100-40-01	1809.3	294.3	0	1267.0	956.8	0	267.4	209.0	0	288.6	284.2	0
	100-40-02	2088.7	520.7	0	4262.7	3929.3	0	363.2	314.7	0	395.2	390.6	0
	100-40-03	1917.3	344.7	0	1605.5	1273.5	0	124.0	73.1	0	85.0	80.6	0
	100-40-04	1866.2	299.5	0	1111.9	770.8	0	232.2	181.9	0	160.4	155.8	0
	100-40-05	2347.7	764.0	0	2915.7	2583.1	0	2113.2	2061.3	0	1048.0	1043.6	0
	100-40-06	2089.4	558.5	0	1632.3	1312.5	0	354.9	305.2	0	279.2	274.8	0
	100-40-07	1822.2	330.4	0	2561.3	2226.4	0	171.6	116.1	0	285.2	280.9	0
	100-40-08	1933.7	443.3	0	3225.3	2904.0	0	718.7	665.1	0	399.7	395.2	0
	100-40-09	1801.6	312.1	0	2122.4	1770.2	0	370.6	320.3	0	239.4	234.9	0
	100-40-10	1907.6	418.2	0	1156.6	845.6	0	292.3	242.2	0	223.3	218.9	0
	Average	**1958.4**	**428.6**	**0**	**2186.1**	**1857.2**	**0**	**500.8**	**448.9**	**0**	**340.4**	**336.0**	**0**
10	100-100-01	4119.4	1700.3	0	7200.0	6724.7	4.64	2054.3	1981.1	0	2200.5	2194.8	0
	100-100-02	3418.8	1101.5	0	6331.5	5725.6	0	298.8	225.1	0	691.3	686.0	0
	100-100-03	3294.0	1005.5	0	4288.7	3754.0	0	395.3	317.6	0	1427.8	1421.9	0
	100-100-04	3217.0	926.6	0	7200.0	6623.1	4.61	1367.1	1294.0	0	672.6	667.0	0
	100-100-05	3215.8	915.9	0	7200.0	6673.9	3.03	1105.8	1031.2	0	1191.5	1186.2	0
	100-100-06	3359.9	1078.7	0	6492.8	5986.7	0	1101.9	1027.5	0	508.0	502.6	0
	100-100-07	2521.4	225.4	0	1669.0	1208.9	0	236.4	163.1	0	109.0	103.5	0
	100-100-08	3223.5	930.1	0	5013.1	4425.0	0	1097.2	1019.2	0	596.6	590.9	0
	100-100-09	3163.8	716.2	0	4259.0	3759.2	0	347.4	273.3	0	514.9	509.6	0
	100-100-10	3424.0	1098.1	0	4223.0	3662.4	0	778.6	699.7	0	522.1	516.7	0
	Average	**3295.8**	**969.8**	**0**	**5387.7**	**4854.4**	**1.23**	**878.3**	**803.2**	**0**	**843.4**	**837.9**	**0**
20	100-100-01	7200.0	n.a.	n.a.	7200.0	4224.4	22.75	1203.1	928.8	0	1695.1	1684.0	0
	100-100-02	7200.0	n.a.	n.a.	7200.0	4480.2	11.76	6257.6	5958.0	0	3475.8	3464.8	0
	100-100-03	7200.0	n.a.	n.a.	7200.0	4535.5	266.84	3986.3	3711.3	0	5441.6	5430.6	0
	100-100-04	7200.0	n.a.	n.a.	7200.0	4590.1	133.33	4499.9	4229.2	0	2229.9	2219.0	0
	100-100-05	7200.0	n.a.	n.a.	7200.0	4428.4	16.37	7200.0	6908.1	4.78	2023.1	2012.6	0
	100-100-06	7200.0	n.a.	n.a.	7200.0	4564.9	7.73	7200.0	6874.1	3.81	2361.8	2350.9	0
	100-100-07	7200.0	n.a.	n.a.	7200.0	4561.5	9.83	1161.9	876.6	0	1012.9	1002.0	0
	100-100-08	7200.0	n.a.	n.a.	7200.0	4469.7	21.86	6261.1	5975.8	0	4971.6	4961.0	0
	100-100-09	7200.0	n.a.	n.a.	7200.0	4265.8	33.42	1098.6	792.6	0	1357.2	1346.5	0
	100-100-10	7200.0	n.a.	n.a.	7200.0	4554.5	14.01	7200.0	6901.9	4.17	3354.6	3343.8	0
	Average	**7200.0**	**n.a.**	**n.a.**	**7200.0**	**4467.5**	**53.79**	**4606.9**	**4315.6**	**1.28**	**2792.4**	**2781.5**	**0**

the constraints. This modification will not alter the design of our algorithm.

5.2.1. *Computational time comparison*

In our first experiment, we compare the computational time of all approaches. We generate p_{jk} using the following equation $p_{jk} = 60 \cdot k/|K|, \forall j \in J, k = 1, \dots, |K|$, where $|K|$ is the number of candidate pricing levels.

Table 2 shows the result of 40 instances for $|K| = \{10, 20\}$. In terms of the total run time (TT), MILP-3 and BBC are significantly more efficient than MILP-1 and MILP-2. For the first 30 instances, the TT of MILP-1 is on average around 5 times longer than those of MILP-3 and BBC. Notably, the drawback of MILP-1 is owing to the difficulty in compiling the model. As shown in Table 2, the solver computational time (ST) in the first 30 instances takes on average only 20–30% of the total run time. This indicates that 70–80% of the run time is dedicated to setting up the model before Gurobi starts the solution process. When the problem size increases to 100 customers, 100 facilities, and 20 pricing levels, i.e., the last 10 instances in the table, MILP-1 fails to finish the solver setup process within 7200 seconds. Consequently, these 10 instances are left unsolved with unknown ST and $rgap$ (labeled with n.a.). Clearly, MILP-1 is not suitable for solving large-scale problems.

Despite that MILP-2 compiles significantly faster MILP-1, due to the large number of constraint sizes in (6b), MILP-2 is computationally more challenging. That is, we observe that the ST of MILP-2 is substantially longer than that of MILP-1 in general. In particular, for the 10 instances with 100 customers, 100 facilities, and 10 pricing levels, MILP-2 fails to solve 3 instances, whereas MILP-1 successfully solves all.

Fortunately, the drawbacks of MILP-1 and MILP-2 can be largely mitigated by employing MILP-3 and BBC. According to Table 2, both MILP-3 and BBC compile faster than MILP-1. Impressively, the solver setup process of BBC takes only a few seconds even for the largest-scale instances. Moreover, their STs are significantly shorter than that of MILP-2.

5.2.2. *Maximal attainable profit*

In our second experiment, we intend to obtain the maximal attainable profit of the FLPr instances when the candidate mill prices are as all integer numbers between 20 and 80. We are motivated by that c and b in all instances take integer values, and thus, the optimal price will take an integer value as well.

Table 3
Profit and the optimal price of FLPr instances. The candidate prices are all integers between 20 and 80.

instance	Best-known Profit	BBC		
		Optimal Profit	Optimal Price	TT[s]
100-40-01	2245	2245	{40, 43, 39, 49, 38}	757.2
100-40-02	2259	2259	{51, 71, 50, 39, 57}	1397.2
100-40-03	2019	2019	{39, 37, 35, 33, 32}	402.2
100-40-04	1533	1533	{46, 24, 31, 42, 42}	980.5
100-40-05	2386	2386	{45, 58, 46, 69, 47}	2392.1
100-40-06	1960	1960	{38, 35, 48, 38, 41}	974.4
100-40-07	2179	2179	{43, 57, 42, 55, 51}	2503.5
100-40-08	2139	2139	{47, 46, 50, 46, 52}	3319.4
100-40-09	1895	**1904**	{52, 51, 46, 39, 47}	2044.0
100-40-10	2209	2209	{38, 57, 38, 58, 45}	1172.4
100-100-01	2235	2235	{38, 57, 52, 45, 32}	5083.8
100-100-02	2240	2240	{49, 69, 51, 51, 48}	2118.5
100-100-03	1923	1923	{41, 54, 52, 42, 34}	11161.0
100-100-04	2133	2133	{61, 51, 38, 38, 55}	3251.3
100-100-05	2099	2099	{53, 72, 50, 38, 64}	21349.9
100-100-06	2237	2237	{55, 52, 40, 54, 57}	4143.5
100-100-07	1888	**1894**	{42, 45, 50, 35, 36}	6416.0
100-100-08	1825	1825	{44, 45, 46, 60, 40}	12919.1
100-100-09	1767	1767	{51, 50, 28, 42, 35}	10429.5
100-100-10	2368	2368	{55, 44, 60, 37, 66}	3361.8

Table 3 reports the results using our proposed BBC algorithm, where the column “Best-known Profit” is the current best-known objective provided by the benchmark library; the columns “Optimal Profit” and “Optimal Price” stand for, respectively, the objective and the pricing decisions on 5 facilities. As shown in the table, “Optimal Profit” is greater or equal to “Best-known Profit” for all instances. Meanwhile, we successfully improve the objective of 2 instances, i.e., 100-40-09 and 100-100-07 (the corresponding profit values are highlighted in boldface). The experiment thus indicates that our proposed modeling framework and algorithm are reasonable and practically implementable.

5.3. RND dataset

To proceed, we utilize three randomly generated datasets[2]. The location of facilities and customers are on the plane and drawn from 2-dimensional uniform distribution $[0, 100]^2$. We compute $c_{ij} = l_{ij}/2$ where l_{ij} is the Euclidean distance between customer i and facility j. Demand of each customer zone d_i is drawn from uniform distribution [0,100]. Moreover, we set

$$b_i = \bar{c}_i \cdot \lambda \tag{19}$$

where $\bar{c}_i = \sum_{j\in J} c_{ij}/|J|$ (the average value of vector $c_{i.}$) and λ is a positive parameter. A large value of λ indicates that the budget of the customer is larger. Moreover, we consider instances with $|K| = \{20, 40\}$, i.e., 20 pricing levels and 40 pricing levels. The maximum mill price is set at 20. When $|K| = 20$, the candidate pricing levels are {1.0, 2.0, 3.0,..., 20.0}. When $|K| = 40$, the candidate pricing levels are {0.5, 1.0, 1.5,..., 20.0}.

5.3.1. RND-M

In the context of the above setup, we generate medium-scale instances as follows.

- **RND-M**. Instances with 100 customer zones. We consider the following combinations of instance sizes and param eters: $|J| = \{50, 100\}$, $|K| = \{20, 40\}$, $f = \{500, 1000, 2000, 3000\}$, and $\lambda = \{0.5, 0.6, 0.7\}$. In total, the number of instances is $2 \times 2 \times 4 \times 3 = 48$.

[2] All datasets are generated using Python with fixed random seeds and can be provided upon request.

We summarize the computational results on RND-M under a 2-hour time limit in Fig. 2. This figure shows the percentage of instances that are solved optimally (% instances solved) versus the total run time (TT). A point in the figure with coordinates (m, n) indicates that for n% of the instances, the required TT to solve them to optimality is less than m seconds. According to Fig. 2, BBC significantly outperforms the MILP approaches because BBC quickly solves all instances optimally.

In total, MILP-1 can only solve 25% of the instances. In effect, except for the “smallest-scale” instances with $|J| = 50$ and $|K| = 20$ (whose solver compiling times are more than 2000 seconds), MILP-1 fails to compile within 2 hours for all other instances. This result, once again, suggests that the performance of MILP-1 is rather limited and is largely impeded by the bottleneck in the solver setup process. For MILP-2, 25% of the instances are unsolved due to the same reason. Finally, despite that MILP-3 solves all 48 instances, we start to observe long solver compiling times for MILP-3. That is, for the instances with $|J| = 100$ and $|K| = 40$, the average compiling time has exceeded 1000 seconds, which can be inferred from the long flat curve of MILP-3 between 1000 and 2000-second TT in Fig. 2. This signals a potential problem of MILP-3 when it is applied to larger-scale instances.

5.3.2. RND-L

Since both MILP-3 and BBC performs well on the above medium-scale instances, we further compare their performance using large-scale instances, which are described as follows.

- **RND-L**: Instances with $|K| = 20$. We consider the following combinations of instance sizes and parameters: $(|I|, |J|) = \{(500, 100), (500, 200), (600, 200)\}$, $f = \{3000, 5000, 10000\}$, and $\lambda = \{0.5, 0.6, 0.7\}$.

Table 4 shows the results under a 2-hour time limit. We observe that although the problem scale is large, BBC manages to solve 22 instances out of 27 within 2 hours. For these 5 unsolved instances, the average *rgap* is only 0.98% (with a maximum value of 1.09%). Such gaps are small enough to ensure that proven high-quality solutions to these instances are found upon termination. This observation justifies the capability of BBC in solving large-scale [FLMPr].

On the contrary, the performance of MILP-3 on these instances is not satisfactory. In particular, when $|J| = 200$, all 18 instances cannot be compiled within 2 hours. This indicates that the MILP approaches are practically ineffective for these instances owing to the time-consuming solver compiling process.

5.3.3. RND-LR

Based on the above results, BBC is the most effective approach, particularly when the instance scale is large. In our last computational experiment, we explore the capability of BBC using even larger-scale instances, which are described as follows.

- **RND-LR**: Instances with $|J| = 200$. Consider the following combinations of instance sizes and parameters: $|I| = \{800, 1000\}$, $|K| = \{20, 40\}$, $f = \{3000, 10000\}$, and $\lambda = \{0.5, 0.6, 0.7\}$.

Table 5 reports the results under an 8-hour time limit. In general, when the number of pricing level is 20, the performance of BBC is acceptable: half of the instances are solved optimally, and for the unsolved instances, the worst *rgap* is 2.88%. In particular, BBC can handle instances with 800 customers reasonably well.

However, under 40 pricing levels, BBC may terminate with a *rgap* up to 5.46% when there are 1000 customers. In fact, for those unsolved instances, *rgap* moves very slowly. As an example, for the instance with $(|I|, |J|, |K|, \lambda, f) = (1000, 200, 40, 0.5, 3000)$, the corresponding *rgap*s when run times are 4 hours, 6 hours, and 8

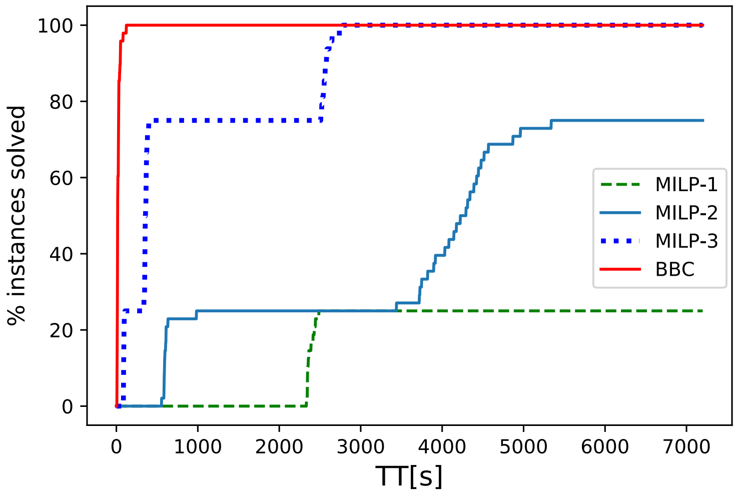


Fig. 2. Computational results of RND-M instances.

Table 4
Computational results of MILP-3 and BBC on RND-L instances with $|K| = 20$. #F and $\bar{p}$ represent the number of opened facilities and the average mill price on the opened facilities, respectively.

$(\lvert I\rvert, \lvert J\rvert)$	λ	f	MILP-3						BBC					
			TT[s]	ST[s]	rgap[%]	Profit	#F	$\bar{p}$	TT[s]	ST[s]	rgap[%]	Profit	#F	$\bar{p}$
(500,100)	0.5	3000	3986.8	2512.2	0	96114	22	7.5	144.6	90.9	0	96114	22	7.5
		5000	4032.7	2527.4	0	60310	15	6.9	82.4	29.0	0	60310	15	6.9
		10000	3834.3	2420.9	0	15352	4	5.5	59.8	9.1	0	15352	4	5.5
	0.6	3000	3708.2	2277.4	0	156139	25	9.6	409.6	354.9	0	156139	25	9.6
		5000	3778.2	2314.8	0	118376	17	9.2	146.2	93.3	0	118376	17	9.2
		10000	3646.1	2120.4	0	61695	8	7.4	69.5	15.3	0	61695	8	7.4
	0.7	3000	7200.0	5656.8	1.02	217177	24	11.8	7200.0	7148.3	1.09	217177	24	11.8
		5000	7200.0	5735.7	0.68	178344	16	11.5	1336.9	1284.5	0	178344	16	11.5
		10000	3609.9	2104.6	0	117097	10	9.3	105.2	52.3	0	117097	10	9.3
(500,200)	0.5	3000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	263.8	161.0	0	105019	23	7.6
		5000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	178.7	75.7	0	66697	15	7.1
		10000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	118.9	16.3	0	17362	6	5.5
	0.6	3000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	5513.9	5411.2	0	165959	24	10
		5000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	496.5	395.8	0	124829	16	8.9
		10000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	150.6	45.1	0	63870	9	7.2
	0.7	3000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	7200.0	7093.3	0.89	230177	26	12.5
		5000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	4990.5	4886.6	0	187769	17	11.9
		10000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	361.1	260.3	0	123441	10	9.2
(600,200)	0.5	3000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	957.7	829.2	0	134567	24	7.5
		5000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	279.1	155.9	0	90306	20	7.6
		10000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	156.4	32.6	0	31329	8	5.9
	0.6	3000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	7200.0	7088.3	0.93	207307	27	9.9
		5000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	1330.3	1202.2	0	161238	19	9.6
		10000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	290.9	168.5	0	88335	11	7.8
	0.7	3000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	7200.0	7070.3	1.02	283349	27	12.6
		5000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	7200.0	7078.6	0.98	236099	21	12.3
		10000	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	605.0	475.2	0	158756	11	9.6

Table 5
Computational results of BBC on RND-LR instances.

$(\lvert I\rvert, \lvert J\rvert)$	λ	f	$\lvert K\rvert = 20$			$\lvert K\rvert = 40$		
			TT[s]	rgap[%]	Profit	TT[s]	rgap[%]	Profit
(800,200)	0.5	3000	18154.7	0	201933	28800.0	1.28	205358
		10000	265.1	0	66883	594.7	0	68015
	0.6	3000	28800.0	1.47	299429	28800.0	1.73	304784
		10000	619.9	0	153689	6594.7	0	155400
	0.7	3000	28800.0	1.41	402260	28800.0	2.95	397429
		10000	5903.1	0	250744	4046.1	0	255915
(1000,200)	0.5	3000	28800.0	1.29	276706	28800.0	2.40	280671
		10000	408.7	0	116333	934.9	0	117524
	0.6	3000	28800.0	2.88	398026	28800.0	5.46	392931
		10000	17416.7	0	229379	28800.0	2.07	233303
	0.7	3000	28800.0	2.83	514391	28800.0	4.07	522744
		10000	28800.0	2.42	350080	28800.0	2.54	357613

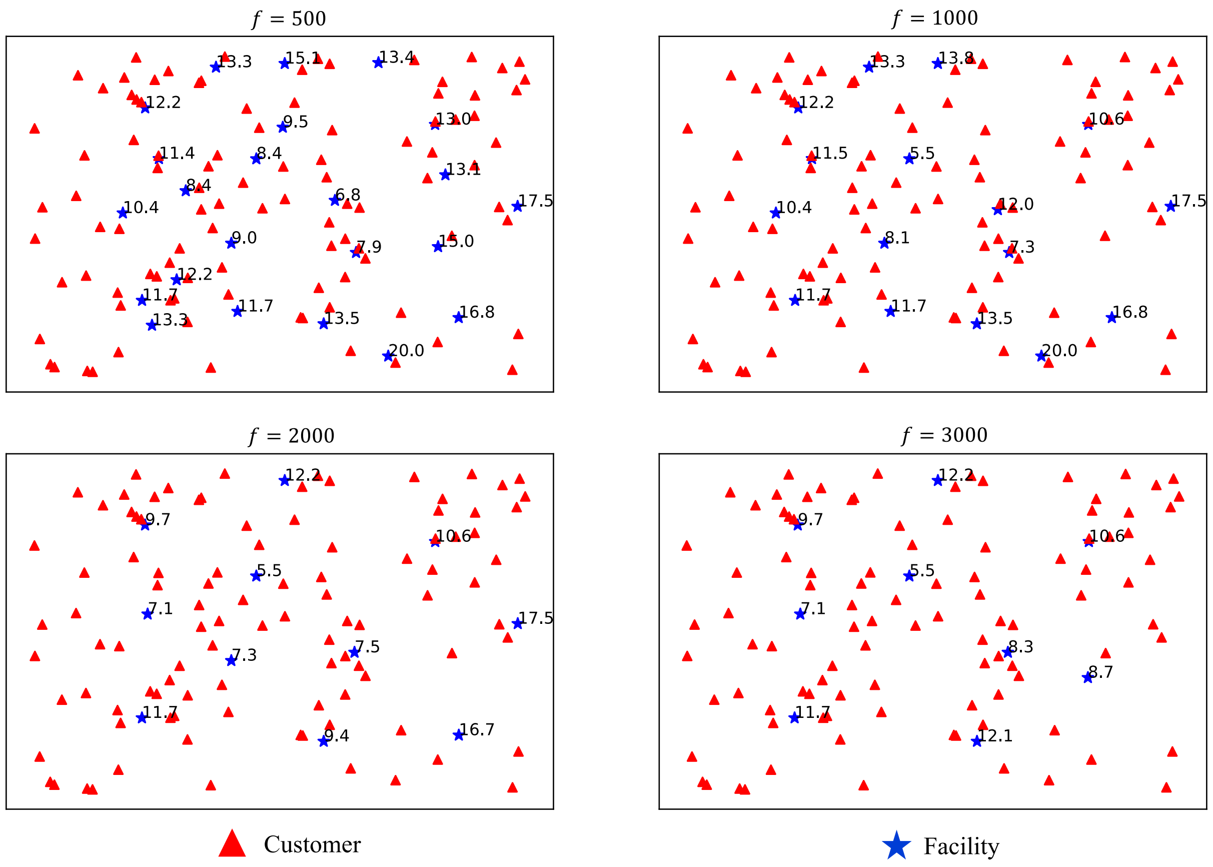


Fig. 3. Geographic locations of customer zones and facilities ($\lambda = 0.7$). The number in the figure indicates the corresponding mill pricing of the facility.

hours are 2.67%, 2.40%, and 2.40%, respectively. Similarly, for the instance with $(|I|, |J|, |K|, \lambda, f) = (1000, 200, 40, 0.7, 10000)$, the corresponding *rgap* reaches 2.54% in the first 3 hours, but such a value remains unchanged until the runtime depletes.

6. Sensitivity analysis and implication

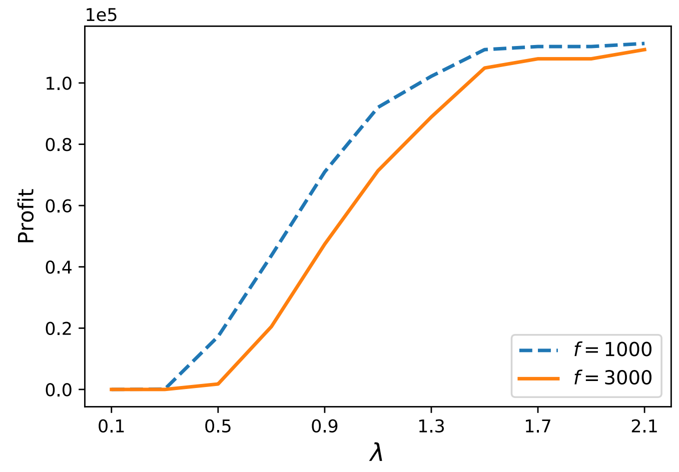
Besides the computational time and *rgap*, Table 4 also provides information about the objectives and solution structures for RND-L instances. The table shows that when the setup cost of facility f is higher, the model suggests opening fewer facilities (see $\#F$), and the average mill price on opened facilities $\bar{p}$ becomes lower. In effect, when the number of opened facilities reduces, customers are on average further away from their nearest facilities (in other words, customers' accessing costs increase). Consequently, customers are less willing to accept high prices since their budgets remain unchanged. In this case, the company has to reduce the charge to attract customers and avoid excessive demand losses. This explains why $\bar{p}$ decreases with f in general as shown in Table 4.

Noting that f and λ significantly affect the location and pricing decisions, our next experiment is to provide sensitivity analysis on the parameters using RND-M with $|J| = 50$ and $|K| = 40$.

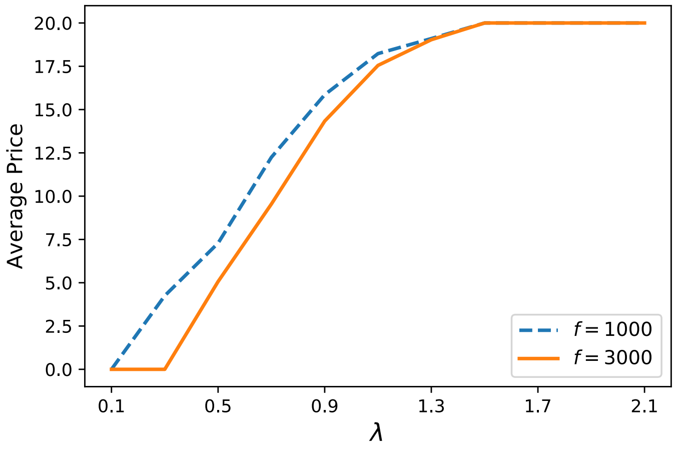
To begin with, we continue the discussion on the impact of f. Fig. 3 depicts the geographic locations of customer zones and opened facilities under 4 different values of f. The red triangular and the blue star present, respectively, the customer zone and the opened facility. Moreover, the number in the figure is the corresponding mill price on the facility. Consistent with the previous result, as f increases, fewer facilities are opened, and the average price decreases. For example, when f is 1000, $\#F$ is 16, and $\bar{p}$ is 12.2. When f goes up to 3000, $\#F$ reduces to 9, and $\bar{p}$ is 9.5. Therefore, the optimal pricing level largely depends on f, which affects the number of opened facilities. In general, opening more facilities increases the proximity of facilities to customers. This in turn allows for setting higher charges to further increase the profit.

Next, we explore the impact of λ by varying λ from 0.1 to 2.1 with step size 0.2 and plot the main results in Fig. 4. Fig. 4(a) shows that the profit increases with λ. Essentially, a large value of λ indicates that customers have larger budgets (or for customers, the costs of seeking services from the outside option are higher) and are more likely to visit the company's facilities and use the services. In this case, the company can boost the profit by charging more. As revealed in Fig. 4(b), there indeed exists an increasing trend of the average price over λ. Therefore, we conclude that when the budgets of customers increase, the optimal pricing levels become higher.

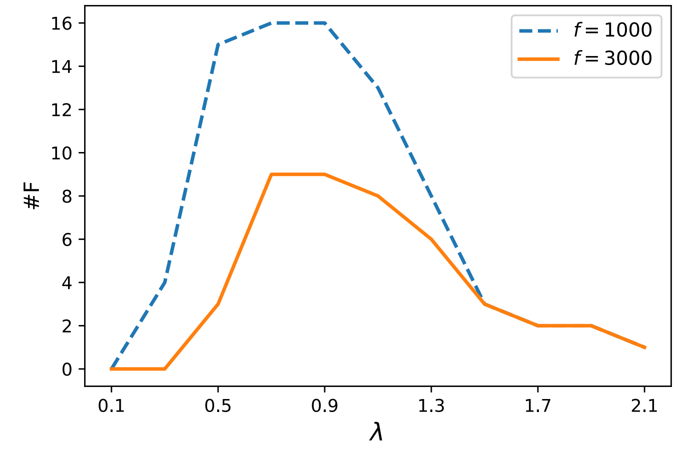
Fig. 4(c) shows that $\#F$ first increases with λ and then decreases. Apparently, when the value of λ is very small (e.g., $\lambda = 0.3$), customers have low budgets. In this case, the optimal price will be low as well (see Fig. 4(b)), and the company cannot collect enough revenue (due to the low charge) to compensate for the cost of opening facilities, leading to the small value of $\#F$. Furthermore, when the value of λ is very large (e.g., $\lambda = 1.9$), customers have high budgets and thus show a high tendency to seek the service from the company. Customers will not be sensitive to both the accessing cost and the charge; therefore, the company does not necessarily need a large number of facilities to gain proximity to customers. Instead, it suffices to maintain a few facilities with high mill prices to achieve profit maximization. On the contrary, when



(a) Profit versus λ



(b) Average Price versus λ



(c) #F versus λ

Fig. 4. Results of the sensitivity analysis.

the value λ is intermediate (e.g., $\lambda = 0.7$ and 0.9), the company has to carefully strike the balance between reducing the accessing cost and controlling the cost of opening facilities. It turns out that the optimal decision is to open more facilities to allow for setting higher mill prices because the increase in the revenue (due to the increased charge) outweighs the cost of additional facilities, which explains the emergence of the bell-shaped curves in Fig. 4(c).

7. Conclusion

In this paper, we investigated a facility location and mill pricing problem, where the number of candidate pricing levels on a facility is finite and the company will select one level for each opened facility to set up the charge. The problem was formulated as a mixed-integer bilevel program, where the company makes location and pricing decisions at the upper level, and customers choose facilities to patronize at the lower level. The bilevel structure of the program rendered the exact solution approaches not intermediate available. To solve the problem exactly, we first proposed methods to reformulate the problem into single-level mixed-integer linear programs and solved the reformulated models using off-the-shelf solvers. However, such an approach suffers from an extremely long solver compiling process and thus cannot solve large-scale instances. Therefore, we further designed a branch-and-cut algorithm leveraging a specialized bilevel feasibility cut to effectively cut off bilevel infeasible solutions. We then conducted extensive computational experiments, and the results revealed that the proposed algorithm significantly outperformed the MILP methods and was capable of handling large-scale instances satisfactorily. Finally, we performed a sensitivity analysis to discuss how parameters affect the profit and the decision of the company and provided interesting observations.

We assumed the optimistic formulation of [FLMPr] in the context of bilevel optimization. Given a location and pricing decisions of the company, if there exists more than one optimal solution to the lower level problem, then the customer will select the one that benefits the company most, namely, the facility with the highest charge. The hidden assumption is that the customer is more sensitive to the accessing cost and is ready to pay more to reduce the accessing cost. We point out that, it is possible to present a *pessimistic* model. In this case, avoiding high charges becomes the priority of the customer when multiple choices leading to the same total cost exist. We would like to leave the discussion for future research. Moreover, we modeled the choice of the customer by a simple rule, i.e., the customer would select the option with the lowest cost. Such a rule stipulates a deterministic “all-or-nothing” choice. In many problems, probabilistic choices are frequently observed. More advanced choice modeling, such as the gravity model and the multinomial logit model based on random utilities, can be introduced to forecast how customers make decisions among alternatives, which will give rise to a class of location pricing problems that are worth investigating from both their problem structures and their exact algorithms.

References

Aboolian, R., Berman, O., & Krass, D. (2008). Optimizing pricing and location decisions for competitive service facilities charging uniform price. *Journal of the Operational Research Society, 59*(11), 1506–1519.
Ahmadi-Javid, A., Amiri, E., & Meskar, M. (2018). A profit-maximization location-routing-pricing problem: A branch-and-price algorithm. *European Journal of Operational Research, 271*(3), 866–881.
Ahmadi-Javid, A., & Ghandali, R. (2014). An efficient optimization procedure for designing a capacitated distribution network with price-sensitive demand. *Optimization and Engineering, 15*(3), 801–817.
Arbib, C., Pınar, M. c., & Tonelli, M. (2020). Competitive location and pricing on a line with metric transportation costs. *European Journal of Operational Research, 282*(1), 188–200.
Berger, A., Grigoriev, A., Panin, A., & Winokurow, A. (2017). Location, pricing and the problem of apollonius. *Optimization Letters, 11*(8), 1797–1805.
Berman, O., Drezner, Z., Tamir, A., & Wesolowsky, G. O. (2009). Optimal location with equitable loads. *Annals of Operations Research, 167*(1), 307–325.
Dan, T., Lodi, A., & Marcotte, P. (2020). Joint location and pricing within a user-optimized environment. *EURO Journal on Computational Optimization, 8*(1), 61–84.

Dastaki, M. S., Setak, M., & Karimi, H. (2021). The multi-zone location–routing problem with pricing: A flow-based formulation and two heuristic approaches. *Soft Computing, 25*(1), 741–769.

Diakova, Z., & Kochetov, Y. (2012). A double VNS heuristic for the facility location and pricing problem. *Electronic Notes in Discrete Mathematics, 39*, 29–34.

Espejo, I., Marín, A., & Rodríguez-Chía, A. M. (2012). Closest assignment constraints in discrete location problems. *European Journal of Operational Research, 219*(1), 49–58.

Fernández, J., Salhi, S., Boglárka, G., et al., (2014). Location equilibria for a continuous competitive facility location problem under delivered pricing. *Computers & Operations Research, 41*, 185–195.

Hajipour, V., Farahani, R. Z., & Fattahi, P. (2016). Bi-objective vibration damping optimization for congested location–pricing problem. *Computers & Operations Research, 70*, 87–100.

Hanjoul, P., Hansen, P., Peeters, D., & Thisse, J.-F. (1990). Uncapacitated plant location under alternative spatial price policies. *Management Science, 36*(1), 41–57.

Hansen, P., Peeters, D., & Thisse, J.-F. (1997). Facility location under zone pricing. *Journal of Regional Science, 37*(1), 1–22.

Hansen, P., & Thisse, J.-F. (1977). Multiplant location for profit maximisation. *Environment and Planning A, 9*(1), 63–73.

Hansen, P., Thisse, J.-F., & Hanjoul, P. (1981). Simple plant location under uniform delivered pricing. *European Journal of Operational Research, 6*(2), 94–103.

Kochetov, Y. A., Panin, A. A., & Plyasunov, A. V. (2015). Comparison of metaheuristics for the bilevel facility location and mill pricing problem. *Journal of Applied and Industrial Mathematics, 9*(3), 392–401.

Kononov, A. V., Panin, A. A., & Plyasunov, A. V. (2018). A new model of competitive location and pricing with the uniform split of the demand. In Proceedings of the international conference on optimization problems and their applications (pp. 16–28). Springer.

Kononov, A. V., Panin, A. A., & Plyasunov, A. V. (2019). A bilevel competitive location and pricing model with nonuniform split of demand. *Journal of Applied and Industrial Mathematics, 13*(3), 500–510.

Kress, D., & Pesch, E. (2016). Competitive location and pricing on networks with random utilities. *Networks and Spatial Economics, 16*(3), 837–863.

Panin, A. A., Pashchenko, M. G., & Plyasunov, A. V. (2014). Bilevel competitive facility location and pricing problems. *Automation and Remote Control, 75*(4), 715–727.

Pelegrín, B., Fernández, P., Pérez, M. D. G., & Hernández, S. C. (2012). On the location of new facilities for chain expansion under delivered pricing. *Omega, 40*(2), 149–158.

Pelegrín, B., Fernández, P., Suárez, R., & García, M. D. (2006). Single facility location on a network under mill and delivered pricing. *IMA Journal of Management Mathematics, 17*(4), 373–385.

Plyasunov, A. V., & Panin, A. A. (2013a). The pricing problem. part i: Exact and approximate algorithms. *Journal of Applied and Industrial Mathematics, 7*(2), 241–251.

Plyasunov, A. V., & Panin, A. A. (2013b). The pricing problem. Part II: Computational complexity. *Journal of Applied and Industrial Mathematics, 7*(3), 420–430.

Vicente, L., Savard, G., & Judice, J. (1996). Discrete linear bilevel programming problem. *Journal of Optimization Theory and Applications, 89*(3), 597–614.

Ye, J. J. (2006). Constraint qualifications and KKT conditions for bilevel programming problems. *Mathematics of Operations Research, 31*(4), 811–824.

Zambrano-Rey, G., López-Ospina, H., & Pérez, J. (2019). Retail store location and pricing within a competitive environment using constrained multinomial logit. *Applied Mathematical Modelling, 75*, 521–534.

Zhang, Y. (2015). Designing a retail store network with strategic pricing in a competitive environment. *International Journal of Production Economics, 159*, 265–273.