Singapore Management University

Institutional Knowledge at Singapore Management University

Research Collection School Of Computing and Information Systems | School of Computing and Information Systems

8-2020

A multicut outer-approximation approach for competitive facility location under random utilities

Tien MAI
Singapore Management University, atmai@smu.edu.sg

Andrea LODI

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

Part of the Artificial Intelligence and Robotics Commons, and the OS and Networks Commons

Citation

1

This Journal Article is brought to you for free and open access by the School of Computing and Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Computing and Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email cherylds@smu.edu.sg.

European Journal of Operational Research 284 (2020) 874–881

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Production, Manufacturing, Transportation and Logistics

A multicut outer-approximation approach for competitive facility location under random utilities

Tien Mai[a], Andrea Lodi[b,*]

[a] Singapore-MIT Alliance for Research and Technology (SMART), 1 Create Way, Singapore
[b] Canada Excellence Research Chair in "Data Science for Real-time Decision-making", Department of Mathematics and Industrial Engineering - École Polytechnique de Montréal, Pavillon André Aisenstadt, 2920 Chemin de la Tour, Montréal, Canada

ARTICLE INFO

Article history:
Received 7 August 2017
Accepted 13 January 2020
Available online 17 January 2020

Keywords:
Facilities planning and design
Maximum capture
Multinomial logit
Mixed multinomial logit
Multicut outer-approximation

ABSTRACT

This work concerns the maximum capture facility location problem with random utilities, i.e., the problem of seeking to locate new facilities in a competitive market such that the captured demand of users is maximized, assuming that each individual chooses among all available facilities according to a random utility maximization model. The main challenge lies in the nonlinearity of the objective function. Motivated by the convexity and separable structure of such an objective function, we propose an enhanced implementation of the outer approximation scheme. Our algorithm works in a cutting plane fashion and allows to separate the objective function into a number of sub-functions and create linear cuts for each sub-function at each outer-approximation iteration. We compare our approach with the state-of-the-art method and, for the first time in an extensive way, with other existing nonlinear solvers using three data sets from recent literature. Our experiments show the robustness of our approach, especially on large instances, in terms of both computing time and number instances solved to optimality.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

We consider a facility location problem in a competitive market, a problem that has been receiving a growing attention in the last decade. The problem concerns how to locate new facilities in a competitive market such that the captured demand of users is maximized, assuming that each individual chooses among all available facilities according to a random utility maximization model. In this problem, two aspects are taken into account, namely, the demand of customers and the competitors in the market. For the latter, the companies that would like to locate new facilities have to compete for their market share. To address these aspects, several competitive facility location models have been proposed in the literature. In general, these models are based on the assumption that customers choose among different facilities based on a given utility that they assign for each location. Such utilities are typically functions of facility attributes/features, e.g., distances, prices and transportation costs.

There are basically two main modeling approaches for the problem. The first approach, which we refer to as the *deterministic approach*, is based on the assumption that customers choose a facility in a deterministic way. For example, ReVelle (1986) proposes a model in which customers choose the closest facility among different competitors. This model, therefore, implies that all the demand of a zone is assigned entirely to a facility, which is not realistic. An alternative approach is the model proposed in Huff (1964), in which the demand captured by a facility is proportional to the attractiveness of the facility and inversely proportional to the distance. The reader can consult Berman, Drezner, Drezner, and Krass (2009) for a review.

The second modeling approach is referred to as the *probabilistic approach*, in which the demand of customers is captured by a probabilistic model, i.e., a model that can assign probabilities to the facilities. The random utility maximization framework; see for instance Ben-Akiva and Lerman (1985) or McFadden (1973), is popular in this context. Under this framework, we assume that there is a random utility associated with each facility, and it is determined by the attributes/features of the facility. Under the "utility maximization" assumption, this way of modeling allows us to compute the probability that a customer chooses a facility versus other facilities. Then, the facility location problem can be described as follows: How to locate facilities in a competitive market such that the expected market share captured by the new facilities is maximized (so, the problem is also called as the "maximum capture" problem).

Among the random utility maximization models in the literature, the multinomial logit (MNL) is widely used due to its

* Corresponding author.
E-mail address: andrea.lodi@polymtl.ca (A. Lodi).

https://doi.org/10.1016/j.ejor.2020.01.020
0377-2217/© 2020 Elsevier B.V. All rights reserved.

simple structure. The first MNL-based facility location approach was introduced by Benati and Hansen (2002) and had several applications afterwards, e.g., locating schools (Haase & Müller, 2013a), preventive health-care facilities (Haase & Müller, 2015) and siting park-and-ride facilities (Aros-Vera, Marianov, & Mitchell, 2013). The advantage of this approach, compared to the deterministic one, is that the probabilistic models allow consideration of the characteristics of the facilities and customers to model the choice decisions, and the choice model can be trained/estimated using supervised data of customer choice decisions, so the demand of customers can be predicted more accurately. The challenge, however, lies in the fact that the corresponding data-driven discrete optimization problems are nonlinear, thus they are typically difficult to solve. Existing approaches address this challenge by reformulating the problems into Mixed-Integer Linear Programing (MILP) models, which are convenient to solve. Different MILP models have been proposed by Benati and Hansen (2002), Hasse (2009) and Zhang, Berman, and Verter (2012). These reformulations have been evaluated and compared by Haase and Müller (2014) and they concluded that the one proposed in Hasse (2009) is the most efficient. This MILP reformulation has been further strengthened by Freire, Moreno, and Yushimito (2016) using tighter coefficients in some inequalities. Recently, Ljubić and Moreno (2018) proposed a Branch-and-Cut algorithm based on a multicut implementation of outer-approximation plus specifically-derived sub-modular cuts. This approach is currently the state-of-the-art for the maximum capture facility location problem under the MNL.

Although the MNL model is popular for the maximum capture facility location problem, it is important to note that there are more flexible choice models that can be used. One of the most preferable models in the demand modeling literature is the mixed MNL (MMNL), which is fully flexible for approximating any random utility maximization model (McFadden & Train, 2000). This model has been also considered in some facility location studies, e.g., Müller, Haase, and Kless (2009) and Haase and Müller (2013a). It is worth noting that the above studies solve the MMNL-based problems using MILP approaches, i.e., linearizing objective functions and using MILP solvers, which have been shown to be dominated by the Branch-and-Cut procedure proposed by Ljubić and Moreno (2018).

Our contribution: In this paper, we exploit the convexity and separability of the objective function and propose an enhanced implementation of the multicut outer-approximation algorithm. Our algorithm allows to create a set of piecewise linear functions that outer-approximate separated parts of the objective function by opportunely clustering the customers. This is based on the outer-approximation scheme (Duran & Grossmann, 1986), i.e., it works in a cutting plane fashion by solving a MILP at every iteration, but it allows to generate several cuts per iteration instead of one per iteration as in the original framework. On the other hand, our algorithm differs from the "multicut" Branch-and-Cut procedure of Ljubić and Moreno (2018) by the fact that it generates cuts for groups of demand points instead of cuts for every demand point and it is a Cutting Plane approach instead of a Branch and Cut. From a computational standpoint, this enhanced implementation is compared with the state-of-the-art one proposed by Ljubić and Moreno (2018) and, for the first time, with two outer-approximation based mixed-integer nonlinear programming (MINLP) solvers from the BONMIN package (Bonami et al., 2008) using the three data sets from recent literature. Our computational experiments show that our approach is more robust and more efficient, especially with the real-life large-scale instances from a park-and-ride location problem in New York City, where the number of demand points is huge. This more detailed comparison highlights the strength of each of the different algorithmic ingredients, shedding light on the trade-off between single vs multi cut approaches and between Cutting Plane vs Branch-and-Cut algorithms.

The paper is structured as follows. Section 2 presents the maximum capture problem under the random utility maximization framework. Our algorithm is presented in Section 3. Section 4 reports the computational results comparing the performance of our approach with other exact approaches in the literature. Finally, Section 5 concludes.

2. Maximum capture facility location under random utilities

We are interested in a situation where a firm wants to locate new facilities in a market in which customers are already served by existing competitors. To capture the customers' demand, we assume that a customer selects a facility in the market according to a random utility maximization model. Such a model associates a decision-maker/customer with a random utility and we assume that the customer chooses a facility by maximizing his/her utility. Once a choice model is specified, the firm can select a set of locations to open new facilities to maximize their expected market share given by the choice model.

To describe the problem in detail, we assume that, in the market, there are $\mathcal{M} = \{1, \dots, m\}$ available locations and we denote by $Y \subset \mathcal{M}$ the set of locations that have facilities of the competitor company. Let I be the set of zones where customers are located and q_i be the number of customers located in zone $i \in I$, where a zone can be defined as a geographical area. We can also view I as a set of groups of customers. The objective is to maximize the expected number of customers by locating facilities in a subset of locations $X \subset \mathcal{M}$. Note that X and Y are not necessarily disjoint, i.e., the firm can consider to open a new facility at a location where there are already facilities from the competitor. We denote by $R(X)$ the expected number of customers given by facilities in X. Therefore, $R(X)$ can be computed as

$$R(X) = \sum_{i \in I} q_i \sum_{j \in X} P(i, j|X, Y),$$

where $P(i, j|X, Y)$ is the probability that a customer located in zone i selects facility $j \in X$. As mentioned, a random utility maximization model associates each pair of location $j \in X \cup Y$ and zone $i \in I$ with a random utility u_{ij}, which is typically a sum of two parts, i.e., $u_{ij} = v_{ij} + \epsilon_{ij}$, where v_{ij} refers to the deterministic part of the utility and often contains observed attributes/features of location j and zone i, and ϵ_{ij} is a random term that is unknown to the analyst. The random utility maximization framework assumes that the customer selects a facility by maximizing the associated random utility. More precisely, the framework allows to compute the probability that a customer i selects a facility located at j as

$$P(i, j|X, Y) = P(u_{ij} \geq u_{ij'}, \ \forall j' \in X \cup Y).$$

If the MNL model is used to predict the choice probabilities of customers, then $R(X)$ can be computed as

$$R(X) = \sum_{i \in I} q_i \frac{\sum_{j \in X} e^{v_{ij}}}{\sum_{j \in X} e^{v_{ij}} + \sum_{j \in Y} e^{v_{ij}}}, \tag{1}$$

where $v_{ij} = (\beta^*)^{\mathrm{T}} a_{ij}$ is the utility associated with location j and a customer located in zone i, β^* are the parameters of the MNL model and a_{ij} is the vector of features/attributes associated with location j and customers at zone i. For notational simplicity, we denote $V_{ij} = e^{v_{ij}}$.

Then, the maximum capture problem under the MNL model can be written as

$$\max_{X \subset \mathcal{M}} \sum_{i \in I} q_i \frac{\sum_{j \in X} V_{ij}}{\sum_{j \in X} V_{ij} + U_Y^i}, \tag{2}$$

where $U_Y^i = \sum_{j \in Y} e^{v_{ij}}$, which is a constant in the optimization problem. We can also formulate (2) as a MINLP model as

$$\max_{\substack{x_j \in \{0,1\} \\ j \in \{1,2,\ldots,m\}}} \quad \sum_{i \in I} q_i \left(\frac{\sum_{j=1}^m x_j V_{ij}}{\sum_{j=1}^m x_j V_{ij} + U_Y^i} \right), \tag{FL-MNL}$$

where x_j, $j \in \mathcal{M}$, is equal to 1 if location j is selected and $x_j = 0$ otherwise.

Even though the MNL is widely used to model discrete choice behaviors due to its simple structure, it is well-known that the model exhibits the independence of irrelevant alternatives (IIA) property. In this context, the property implies that for two facilities, the ratio of the choice probabilities is the same no matter what other facilities are available or what the attributes/features of the other facilities are. This is, in general, not reasonable and to relax the IIA property, several choice models have been proposed. Among them, the mixed MNL model is one of the most preferable (Train, 2003). In the MMNL model, it is assumed that the utilities v_{ij} are no-longer deterministic, but contain some random components. To approximate the choice probabilities as well as the expected number of customers, one needs to sample over the randomness of the utilities. Formally speaking, the maximum capture problem under the MMNL model can be formulated as (Haase, Müller, Krohn, & Hensher, 2016)

$$\max_{\substack{x_j \in \{0,1\} \\ j \in \{1,2,\ldots,m\}}} \quad \frac{1}{N} \sum_{n=1}^{N} \sum_{i \in I} q_i \left(\frac{\sum_{j=1}^m x_j V_{ij}^n}{\sum_{j=1}^m x_j V_{ij}^n + U_Y^{in}} \right), \tag{FL-MMNL}$$

where $\{V_{ij}^1, \ldots, V_{ij}^N\}$ and $\{U_Y^{i1}, \ldots, U_Y^{iN}\}$ are N realizations of V_{ij} and U_Y^i, sampled over the randomness of the utilities v_{ij}. Indeed, FL-MMNL shares the same structure of the FL-MNL, meaning that, in general, any algorithm being able to solve FL-MNL can be used to solve FL-MMNL (Haase & Müller, 2013b; Haase et al., 2016; Müller et al., 2009). Note that we write FL-MNL and FL-MMNL in their simplest forms and different business constraints can be included, e.g., a constraint on the number of facilities that the firm would like to open or constraints on the budget the firm has to open facilities.

It is worth noting that beside the MMNL model, authors in demand modeling also consider the multivariate extreme value (MEV) model (McFadden, 1981) to relax the IIA property of the MNL. However, the main advantage of using either the MNL or the MMNL model is that the resulting objective functions are convex and can be linearized, so the problems under the MNL or MMNL can be solved exactly by MILP solvers. On the contrary, the objective functions given by the MEV are typically nonlinear and nonconvex, making the resulting models way more difficult to solve exactly.

As pointed out, the nonlinearity of the objective functions of both FL-MNL and FL-MMNL are convex. Basically, FL-MNL and FL-MMNL are 0–1 fractional linear programming models, for which it is possible to reformulate the nonlinear models into mixed-integer linear programming ones (Wu, 1997). In the context of competitive facility location, this has been done in some previous studies, e.g., Benati and Hansen (2002), Hasse (2009) and Zhang et al. (2012). It is important to note that the objective functions of FL-MNL and FL-MMNL are concave and continuously differentiable, so, a convex MINLP solver such as the BONMIN (Bonami et al., 2008) can be used to solve the problem. In addition, one can take this advantage to build a Branch-and-Cut procedure with outer-approximation cuts. As already discussed, such an approach is studied in Ljubić and Moreno (2018) and has been shown to achieve the state-of-the-art results for the maximum capture facility location problem under the MNL model. In the remainder of the paper, we concentrate on the competitive facility location under the MNL model because, as previously shown, it has the same theoretical complexity of that under the (more realistic) MMNL model and it is still quite challenging to solve. Nevertheless, we will discuss computational issues related to solve the MMNL version at the end of the computational section.

3. Multicut outer-approximation scheme

In this section, we focus on the use of the outer-approximation scheme to solve the facility location problem under the MNL model. In particular, our approach is motivated by the fact that the objective function can be separated into several sub-functions, each of which is convex and continuously differentiable. This suggests the idea of building an outer-approximation for each sub-function by using subgradient cuts. In the following, we describe the multicut version and show how the new scheme can be applied to the maximum capture problem.

The maximum capture facility location problem can be written in general form as the following integer nonlinear programming problem:

$$\begin{array}{lll} \underset{x}{\text{minimize}} & G(x) & \\ \text{subject to} & Ax \leq b & \\ & x \in \{0,1\}^m, & \end{array} \tag{P1}$$

where $Ax \leq b$ are some linear business constraints, e.g., upper and lower bounds on the number of facilities that the company would like to open. Moreover, in the context of the maximum capture problem under the MNL, one can show that the objective function $G(x)$ is convex and continuously differentiable, which is essential for the use of the outer-approximation scheme.

The general idea of the outer-approximation scheme (Duran & Grossmann, 1986) is to create a piecewise linear and convex function $Q(x)$ that underestimates $G(x)$, i.e., $Q(x) \leq G(x), \forall x$. If this function is tight at every integer point in the feasible set of the problem, i.e., $Q(x) = G(x)$, $\forall x \in \{0,1\}^n$, $Ax \leq b$, then we can find an optimal solution to FL-MNL by solving $\min\{Q(x) | x \in \{0, 1\}^n, Ax \leq b\}$. To this end, one can reformulate (P1) as $\min_{\theta, x}\{\theta | \theta \geq G(x)\}$ (plus the regular constraints). Then, we can relax the constraint $\theta \geq G(x)$ and consider θ as an underestimator of $G(x)$. An outer-approximation algorithm is basically an iterative procedure in which at each iteration we add cuts in (x, θ)-space to approximate the shape of $G(x)$. This is done until we find a solution (x^*, θ^*) such that $\theta^* = G(x^*)$. It is also well-known that the algorithm terminates after a finite number of iterations.

We now discuss an extended version of the outer-approximation algorithm proposed in Duran and Grossmann (1986), which allows to create several piecewise linear and convex functions to outer-approximate $G(x)$. This is motivated by the fact that the objective function in our context is separable, i.e., $G(x)$ can be written as a sum of convex functions. This suggests that it is possible to add several cuts to the master problem at each iteration of the outer-approximation algorithm. The idea of introducing multiple cuts for separable objective functions is well-known in the stochastic programming literature, i.e., the multicut L-shape method (see Birge & Louveaux, 1988, for instance).

We assume that the objective function can be written as a sum of T convex functions

$$G(x) = \sum_{t=1}^{T} g_t(x).$$

Then, the corresponding master problem can be defined as

$$\begin{array}{lll} \underset{x}{\text{minimize}} & \sum_{t=1}^{T} \theta_t & \\ \text{subject to} & Ax \leq b & \\ & \Pi_t x - \mathbf{1}\theta_t \leq \pi_{0t} & t = 1 \ldots, T \end{array}$$

$$\theta_t \geq L_t \qquad t = 1\dots,T$$
$$x \in \{0,1\}^m,$$

where $\Pi_t x - \mathbf{1}\theta_t \leq \pi_{0t}$ is the set of subgradient cuts corresponding to $g_t(x)$ and L_t is a lower bound of $g_t(x)$. The multicut outer-approximation works as follows. At each iteration, for a given candidate $x^* \in X$, (up to) T subgradient cuts are added to the master problem

$$\theta_t \geq \nabla g_t(x^*)(x - x^*) + g_t(x^*),\ t = 1,\dots,T, \tag{P2}$$

where $\nabla g_t(x^*)$ is the gradient of $g_t(x^*)$ at x^*. The procedure stops when it finds a solution $(x^*, \theta_1^*, \dots, \theta_T^*)$ such that $\sum_{t=1}^T \theta_t^* \geq G(x^*)$. Similarly to the classical approach, one can show that the multicut algorithm always terminates after a finite number of steps, and the returned solution is optimal to (P1).

We now get back to the maximum capture facility location problem, in which the objective function is a sum of linear fractional functions

$$G(x) = -\sum_{i\in I} q_i\left(\frac{\sum_{j=1}^m x_j V_{ij}}{\sum_{j=1}^m x_j V_{ij} + U_Y^i}\right).$$

If we divide the set of clients I by T disjoint subsets $I_1,\dots,I_T$, and define

$$g_t(x) = -\sum_{i\in I_t} q_i\left(\frac{\sum_{j=1}^m x_j V_{ij}}{\sum_{j=1}^m x_j V_{ij} + U_Y^i}\right),$$

then, we can write $G(x) = \sum_{t=1}^T g_t(x)$, and the multicut outer-approximation scheme can be applied.

A typical business constraint for the facility location problem is $l \leq \sum_{j=1}^m x_j \leq u$, where l and u are the minimum and maximum number of facilities that the firm would like to open. Proposition 1 below indicates that, to maximize the expected number of customers, one needs to open as many facilities as possible. In other words, we can reduce a bound constraint into a p-median constraint, i.e., $\sum_{j=1}^m x_j = r,\ r \in \mathbb{N}_+$.

Proposition 1. *Assume that x^* is an optimal solution to (P1), in which the constraint $Ax \leq b$ is replaced by $l \leq \sum_{j=1}^m x_j \leq u$, then $\sum_{j=1}^m x_j^* = u$.*

Proof. The following remark is easy to verify. Given $i \in I$, and $x \in \{0,1\}^m$, for any $\alpha \geq 0$ we have

$$\frac{\sum_{j=1}^m x_j V_{ij}}{\sum_{j=1}^m x_j V_{ij} + U_Y^i} \leq \frac{\sum_{j=1}^m x_j V_{ij} + \alpha}{\sum_{j=1}^m x_j V_{ij} + \alpha + U_Y^i}. \tag{4}$$

Inequality (4) implies that, we can get better objective values by opening more faculties. The proposition is just a direct result of the above remark. □

The master problem (P2) is initialized with some lower bounds L_t of $g_t(x)$. Since $g_t(x)$ is convex, these lower bounds can be computed by solving the nonlinear convex optimization problem

$$L_t = \min_{\substack{x\in[0,1]^m\\ Ax\leq b}} g_t(x), \tag{5}$$

with a note that solving (5) could be expensive, and we need to solve that nonlinear problem for each $t = 1,\dots,T$. Using (4), we can have a faster way to obtain lower bounds for $g_t(x)$ as

$$g_t(x) = -\sum_{i\in I_t}\left(\frac{\sum_{j=1}^m x_j V_{ij}}{\sum_{j=1}^m x_j V_{ij} + U_Y^i}\right)$$
$$\geq -\sum_{i\in I_t}\left(\frac{\sum_{j=1}^m V_{ij}}{\sum_{j=1}^m V_{ij} + U_Y^i}\right), \forall x \in \{0,1\}^m. \tag{6}$$

Moreover, if we consider the problem with a business constraint $\sum_{j=1}^m x_j = r$, where r is the number of facilities to be opened, then we can obtain tighter bounds by sorting the utilities $\{V_{ij},\ j = 1,\dots,m\}$ and using the following inequality, which is easy to validate:

$$g_t(x) \geq -\sum_{i\in I_t}\left(\frac{\sum_{k=1}^r V_{ij_k^i}}{\sum_{k=1}^r V_{ij_k^i} + U_Y^i}\right),\ \forall x \in \{0,1\}^m, \sum_{j=1}^m x_j = r, \tag{7}$$

where $(j_1^i,\dots,j_m^i)$ is a permutation of $\{1,\dots,m\}$ such that $V_{ij_1^i} \geq V_{ij_2^i} \geq \dots \geq V_{ij_m^i}$.

Similarly to the classical outer-approximation algorithm (Duran & Grossmann, 1986), if the multicut algorithm finds a candidate solution $x^* \in \{0,1\}^m$ that has been already found previously, then x^* is an optimal solution to P1. This suggests a way to avoid recomputing the objective function $G(x)$, which could be costly with large instances. More precisely, each time a solution x^* is found, we can add x^* to a set Z and also save the objective value $G(x^*)$. At each iteration, after solving the master problem to obtain $(x^*, \theta^*,\dots,\theta_T^*)$, we can first check if x^* is in Z, then we can return x^* as an optimal solution. Otherwise, we compute the gradient of $g_t(.)$ at x^* and add the corresponding subgradient cuts to the master problem.

We describe the multicut outer-approximation scheme in Algorithm 1. The difference between Algorithm 1 and the standard outer-approximation algorithm presented in Bonami et al. (2008) is that: (i) at each iteration, the multicut algorithm creates several subgradient cuts and then adds them to the master problem, (ii) we do not solve the continuous relaxation of the problem to initialize the master problem, instead, we compute the lower bound by using either (6) or (7), and (iii) we save the set of binary solutions found at each iteration to avoid recomputing the objective function and its gradient. The latter (simple) modification helps to reduce the computing time in cases that the objective function is expensive to evaluate, and/or the outer-approximation algorithm only needs a few iterations to converge.

Algorithm 1: Multicut outer-approximation algorithm.

begin
 # Initialization
Step 1. Chose a lower bound L_t, $t = 1,\dots,T$ and a convergencetolerance $\epsilon > 0$, and $Z = \emptyset$.
Step 2. Initialize the master problem (P2) with empty Π.
Step 3. Compute $(x^*, \theta_1^*,\dots,\theta_T^*)$ as the first solution by solving (P2).
 # Iteratively adding cuts until getting an optimal solution
Step 4. If $x^* \in Z$ then go to Step 6, otherwise set $Z = Z \cup \{x^*\}$ and compute $G(x^*)$
Step 5. If $\sum_{t=1}^T \theta_T^* \geq G(x^*) - \epsilon$, then go to Step 6, otherwise

5.1 Compute $\nabla g_t(x^*)$, $t = 1,\dots,T$, and add subgradient cuts to the master problem (P2)

$$\theta_t \geq \nabla g_t(x^*)(x - x^*) + g_t(x^*),\ t = 1,\dots,T$$

5.2 Solve (P2) to obtain new solution $(x^*, \theta_1^*,\dots,\theta_T^*)$, and go back to Step 4

 # Finalization
Step 6. Return x^* as an optimal solution and $\sum_{t=1}^T \theta_T^*$ as the optimal value.

Basically, the advantage of the multicut algorithm is that it allows to add cuts based on each concave component of the objective function. Therefore, we can expect that the approach can explore better the structure of the nonlinear function, and requires less iterations to converge as compared to the single-cut one. However, the number of cuts added to the master problem is T times larger than the single-cut version, leading to the fact that each iteration of the multicut algorithm is more expensive than an iteration of the single-cut one. If the number of clients, i.e. $|I|$, is small, then we can choose $T = |I|$. In cases that $|I|$ is too large, we can select $T \ll |I|$ to avoid having too many cuts added to the master problem. This trade-off is discussed in details at the end of the computation evaluation of the next section.

4. Computational experiments

In this section, we evaluate the performance of our multicut outer-approximation (MOA) algorithm on standard data sets from the literature and we provide a comparison between MOA and the state-of-the-art approach proposed by Ljubić and Moreno (2018), i.e., a Branch-and-Cut algorithm based on a multicut implementation of outer-approximation and sub-modular cuts. Let use denote this approach by BC. We use the three data sets used in Ljubić and Moreno (2018) as benchmark instances. We also compare our approach with other existing convex MINLP solvers that are based on the outer-approximation scheme as well. More precisely, we consider the algorithms implemented in the BONMIN package (Bonami et al., 2008). Note that BONMIN contains 4 different algorithms for solving convex MINLP problems and two of them are based on the outer-approximation scheme, i.e., one is an outer-approximation decomposition algorithm (denoted as BM-OA) and the other one is a hybrid outer-approximation-based Branch-and-Cut algorithm (denoted as BM-HYB). In general, both BC and BM-HYB are based on a Branch-and-Cut scheme, i.e., the outer approximation is performed within a unique enumeration tree. The main difference between the two approaches is that BC uses submodular cuts in addition to outer-approximation cuts and it generates cuts for each fractional (concave) component of the objective function. On the other hand, BM-HYB generates cuts for the entire objective function, which means that only one cut is generated at each iteration. It is also important to note that the BM-OA also generates subgradient cuts for the entire objective function, while for the MOA we divide the set of separable components of the objective function into some smaller groups, and we generate cuts for each group (Eq. (3), where T is the number of group).

We refer the reader to Bonami et al. (2008) for a detailed description of these algorithms. We use a MATLAB interface of BONMIN, i.e., the *OPTI Toolbox* (http://www.i2c2.aut.ac.nz/Wiki/OPTI/) for the experiments.

4.1. Experimental setting

We briefly describe the three data sets in the following and refer the reader to Ljubić and Moreno (2018) and Freire et al. (2016) for more details.

- HM14: The data set includes instances generated randomly in a plane, with $|I| \in \{50, 100, 200, 400\}$ and $|\mathcal{M}| \in \{25, 50, 100\}$.
- ORlib: The data set consists of 11 problems, in which there are eight problems with $|I| = 50$, $|\mathcal{M}| \in \{25, 50\}$ and three problems with $|I| = 1000$, $|\mathcal{M}| = 100$.
- P&R-NYC (or simply NYC): the data set comes from a large-scale park-and-ride location problem in New York City, with $|I| = 82,341$ and $|\mathcal{M}| = 59$. These are the largest and most challenging instances, as reported by previous studies.

Only constraints of the form $\sum_{j=1}^{m} x_j = r$ are considered, with $r \in \{2, \dots, 10\}$. For the NYC data set, we also test with $r = \lfloor |\mathcal{M}|/2 \rfloor$, i.e., $r = 29$. Similar to previous studies, we specify the deterministic part of the utility associated with a location $j \in \mathcal{M}$ as $v_{ij} = -\beta c_{ij}$ and $v_{ij'} = -\beta\alpha c_{ij'}$ for each competitor j', where c_{ij} is the distance between zone/client i and location j. The parameter β refers to the sensitivity of customers about the perceived utilities and parameter α represents the competitiveness of the competitors. We choose the same parameters as in Ljubić and Moreno (2018), i.e., $\alpha = \{0.01, 0.1, 1\}$ and $\beta = \{1, 5, 10\}$ for data sets HM14 and ORlib, and $\beta = \{0.5, 1, 2\}$ and $\alpha = \{0.5, 1, 2\}$ for the NYC. For the NYC data set and its chosen parameters, we refer the reader to Holguin-Veras, Reilly, and Aros-Vera (2012) for more details. It is important to note that, in general, other features of the zones/clients and locations can be used to model customers' utilities, and the parameters β and α can be learned in supervised fashion on data about how customers select the locations. In summary, we test on three data sets, in which the number of instances in HM14, ORlib, NYC is 972, 891, 90, respectively. These are also the instances considered in Ljubić and Moreno (2018) and Freire et al. (2016).

The experiments are conducted on a PC with processor Intel(R)Core(TM) CPUs of 2.8 gigahertz, RAM of 12 gigabytes and operating system Window 10. The MOA algorithm is coded in MATLAB and linked to IBM-ILOG CPLEX 12.6 optimization routines under default settings. We also take the code used in Ljubić and Moreno (2018) to generate results for the BC approach.

4.2. Computational evaluation and comparison

This section provides comparison results using the instances described above. An important setting for our MOA algorithm is the number of cuts T. Indeed, $1 \le T \le |I|$. On the one hand, if we choose small T, the MOA may perform similarly to the single-cut version, thus, may require a large number of iterations to converge. On the other hand, if T is large, the MOA may better explore the structure of the objective function, hence, would be able to reduce the number of iterations but the master problem at each iteration becomes more costly to solve. To achieve good performance, we choose $T = \min\{|I|, 100\}$ for the HM14 instances and $T = \min\{|I|, 1000\}$ for ORlib data set. For the largest instances from NYC data set, because $|I| = 82,341$, we only choose $T = 20$. Those values represent a reasonable compromise in the attempt of not over tuning the algorithm while a detailed discussion on the impact of T on the performance of the MOA algorithm is reported at the end of the section. Given T, we simply cluster the customers into groups of the same size (except the last group, as $|I|$ may be not divisible by T).

Table 1 reports the computational comparison of the four approaches, i.e., MOA, BC, BM-OA and BM-HYB, for the HM14 data set. We report, in the table, the number of solved instances within a time limit of one hour and the average CPU time (in seconds) among those instances solved to optimality. For the MOA and BM-OA, we report the average number of iterations for the instances that can be solved by both approaches. We also indicate in bold the largest number of solved instances, as well as the best CPU time(s), and we only compare the computing time between methods that perform the best in terms of number of solved instances. Note that each row of the table corresponds to 81 solved instances.

Table 1 indicates the strength of both BC and MOA as compared to the BOMMIN ones, as they are capable of solving all the instances to optimality in a few seconds on average. Overall, BC is faster than MOA in terms of computing time although in almost all cases the difference is negligible and, on average, way smaller than one order of magnitude. We also notice that, in terms of number of solved instances, BM-HYB is better than BM-OA, but if we look at the average computing times over solved instances, BM-OA is faster than BM-HYB. It is also interesting to see that, among the instances that BM-OA is capable of solving successfully, the algorithm only needs a few seconds and very few iterations to reach optimality.

Table 2 reports the comparison results of the four approaches for the ORlib instances under a time limit of one hour. Each row of

Table 1
Numerical results for HM14 instances, grouped by the problem name (81 instances per row).

		# (Solved instances)				Computing time (s)*				# (Iterations)*	
$\|I\|$	$\|V\|$	BC	MOA	BM-OA	BM-HYB	BC	MOA	BM-OA	BM-HYB	MOA	BM-OA
50	25	**81**	**81**	67	72	**0.02**	0.05	3.73	49.16	2.00	2.70
50	50	**81**	**81**	60	68	**0.06**	0.10	0.51	58.16	1.78	2.27
50	100	**81**	**81**	48	65	**0.26**	0.32	1.62	76.76	2.63	1.85
100	25	**81**	**81**	68	78	**0.04**	0.11	0.18	44.29	1.90	2.35
100	50	**81**	**81**	45	59	**0.12**	0.28	2.43	24.55	2.24	2.51
100	100	**81**	**81**	43	43	**0.61**	0.66	3.74	102.03	2.72	2.42
200	25	**81**	**81**	55	65	**0.07**	0.29	1.06	1.90	2.12	3.47
200	50	**81**	**81**	45	65	**0.26**	0.76	1.18	71.98	2.24	1.96
200	100	**81**	**81**	42	59	**1.28**	4.54	4.62	76.74	2.88	2.91
400	25	**81**	**81**	40	59	**0.16**	2.49	2.85	1.16	2.40	2.45
400	50	**81**	**81**	36	57	**0.57**	3.66	4.13	4.94	2.56	2.39
400	100	**81**	**81**	43	45	**2.71**	12.90	6.62	63.11	3.26	3.12
Average		**81**	**81**	49.3	62.4	**0.51**	2.18	2.72	47.9	2.39	2.53

* Average among solved instances. BC: Branch&Cut (Ljubić & Moreno, 2018) MOA: Multicut outer-approximation (Algorithm 1) BM-OA: BONMIN's single-cut outer-approximation BM-HYB: BONMIN's Branch&Cut

Table 2
Numerical results for ORlib data set, grouped by the problem name (81 instances per row).

	# (Solved instances)				Computing time (s)*				# (Iterations)*	
Name	BC	MOA	BM-OA	BM-HYB	BC	MOA	BM-OA	BM-HYB	MOA	BM-OA
cap101	**81**	**81**	**81**	**81**	**0.03**	0.06	0.14	3.84	2.00	1.99
cap102	**81**	**81**	**81**	**81**	**0.03**	0.06	0.17	3.14	2.19	2.09
cap103	**81**	**81**	**81**	**81**	**0.03**	0.06	0.10	3.15	2.00	1.85
cap104	**81**	**81**	**81**	**81**	**0.03**	0.06	0.15	3.35	1.90	2.10
cap131	**81**	**81**	**81**	**81**	**0.09**	0.12	0.12	4.16	2.41	2.16
cap132	**81**	**81**	**81**	**81**	**0.09**	0.12	**0.09**	3.77	2.37	1.99
cap133	**81**	**81**	**81**	**81**	0.09	0.11	**0.08**	4.00	2.25	2.15
cap134	**81**	**81**	**81**	**81**	**0.09**	0.12	0.10	5.51	2.36	1.98
capa	**73**	**73**	55	60	**150.14**	242.64	131.00	1601.30	8.95	6.53
capb	**75**	71	59	57	**133.46**	404.11	47.18	1437.95	5.10	7.51
capc	64	**66**	59	62	71.25	**284.22**	19.44	748.00	4.59	5.81
Average	**78.2**	78.0	74.6	75.2	32.30	84.70	18.05	347.10	3.28	3.29

* Average among solved instances.

the table corresponds to 81 instances and we also indicate the best performance in bold. We can separate the instances into two sets; the first set consists of instances of $|I| = 100$ (cap101 - cap134) and the second set contains instances of 1000 customer zones (capa, capb, capc). While the former is easy for the four approaches, the latter is more challenging to be solved. More precisely, BC and MOA perform the best in terms of number of solved instances, and BC is slightly better than the MOA, solving overall 2 more instances within the time limit, although, again, the computing times are close, in the same order of magnitude. In addition, BC is also faster than MOA in terms of computing time. Furthermore, even if worse than BC and MOA in terms of number of solved instances, the average computing times of the BM-OA are remarkably smaller as compared to other approaches. Moreover, when comparing the two BONMIN solvers, we see that BM-OA and BM-HYB perform similarly in terms of number of solved instances, but BM-OA is much faster in terms of computing time. Those observations are in line with the well-known observation that, in case the number of MILPs to be solved is very small, classical outer approximation algorithms are faster than hybrid (branch-and-cut based) ones.

We now discuss the numerical results for the largest data set. Table 3 reports what we obtained when testing the four approaches on the instances from the NYC data set, in which each approach is given a time budget of one hour and each row corresponds to 9 instances. Clearly, our MOA improves over other approaches either solving more instances within the time limit or doing it faster. More precisely, only MOA is able to solve all the instances to optimality. In general, BC is better than the two approaches from BONMIN in terms of number of solved instances, but if we look at the average CPU times, the BONMIN ones and our MOA are remarkably faster, this time over an order of magnitude faster. The reason is that the NYC instances contains a large number of clients/demand points (i.e., 82,341). Because BC generates cuts for each demand point, the number of cuts becomes very large and the problem becomes expensive to solve. On the other hand, for the other approaches (BONMIN and MOA), cuts are only generated for the aggregated objective function (for the BONMIN) or for groups of demand points (MOA), so way less cuts are generated per iteration.

Moreover, we notice that the number of iterations required by our MOA is 6 times larger than that required by the BM-OA. This is in line with the computing times observed. It is also worth noting that, even being similar in terms of number of solved instances, BM-OA is about 3 times faster than BM-HYB. As already partially observed, if BM-OA can solve the problem, it does it in few iterations and it is faster than other approaches, otherwise it does not solve the problem.

In summary, Tables 1–3 report comparison results based on three sets of instances HM14, ORlib and NYC. The results indicate the robustness of BC and MOA as compared to the two BONMIN solvers. Namely, BC is slightly better than the MOA approach for the simulated instances, i.e., ORlib and HM14, but significantly worse than MOA for the real and large ones (i.e. NYC). The results are also consistent with those reported in the previous studies (Ljubić & Moreno, 2018), with a note that BC seems to perform a bit worse in our experiments, which may be due to the differences between the machines used for the tests. Based on the comparative results reported in Ljubić and Moreno (2018) with respect

Table 3
Numerical results for NYC data set, grouped by r (9 instances per row).

	# (Solved instances)				Computing time (s)*				# (Iterations)*	
r	BC	MOA	BM-OA	BM-HYB	BC	MOA	BM-OA	BM-HYB	MOA	BM-OA
2	7	**9**	6	6	232.47	**4.46**	1.18	13.57	7.67	1.83
3	8	**9**	6	6	311.05	**9.14**	0.99	5.55	8.67	2.00
4	8	**9**	6	6	236.06	**11.06**	0.67	5.88	8.5	2.00
5	8	**9**	7	7	314.42	**12.56**	0.89	7.95	17.71	1.71
6	**9**	**9**	6	6	414.74	**14.85**	1.60	5.33	7.33	2.50
7	**9**	**9**	8	8	337.44	**12.29**	1.06	11.45	10.12	2.25
8	8	**9**	8	8	249.66	**14.75**	0.76	16.95	9.5	2.13
9	8	**9**	8	8	248.35	**17.10**	0.82	15.50	8.00	2.25
10	**9**	**9**	8	8	360.08	**19.83**	11.67	6.37	6.25	2.38
29	**9**	**9**	8	8	395.61	**0.47**	15.32	8.23	4.25	2.58
Average	8.3	**9**	7.1	7.1	309.99	**8.80**	3.50	9.68	12.13	2.16

* Average among solved instances.

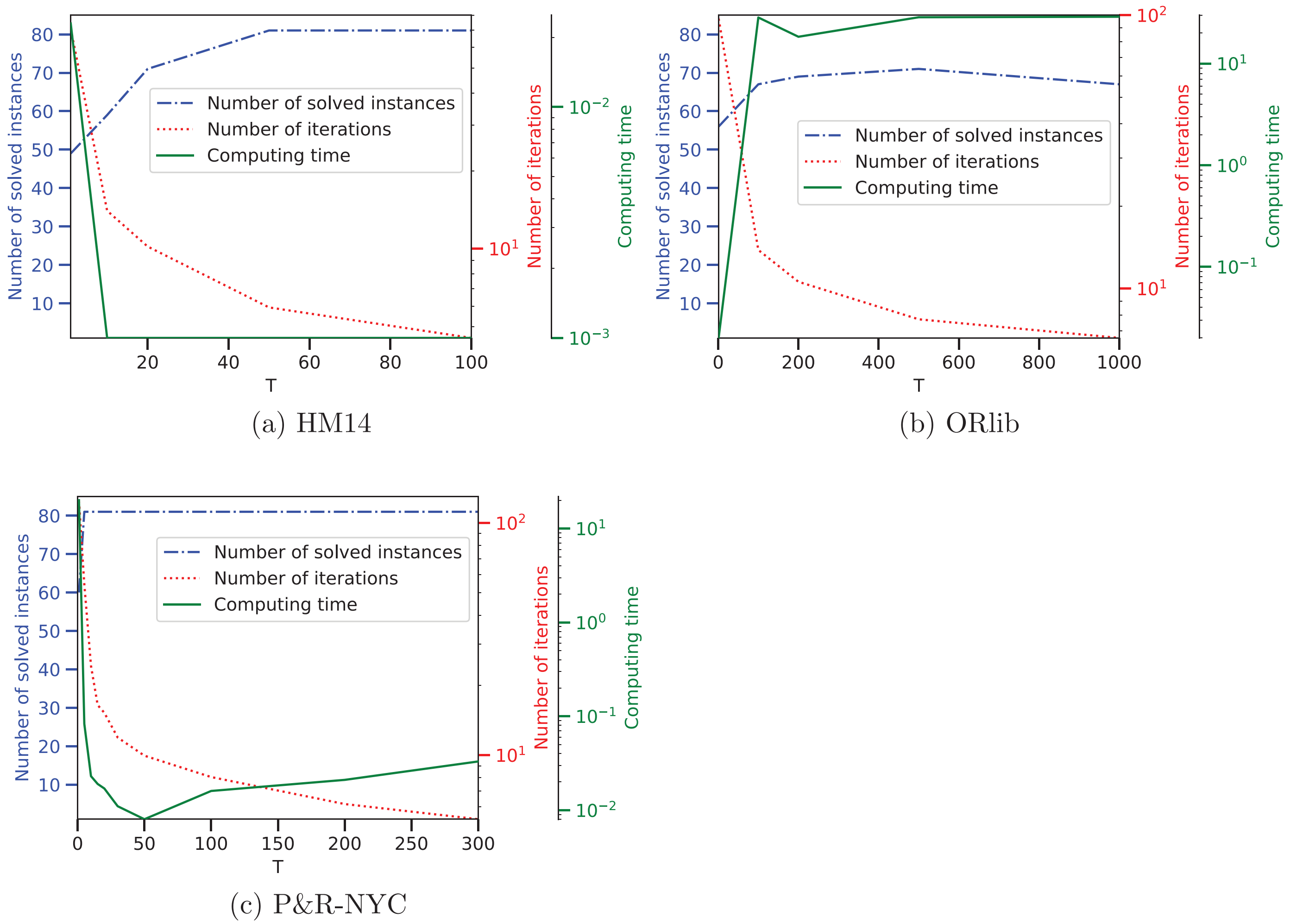

Fig. 1. Performance of MOA algorithm with respect to the numbers of cuts per iteration.

to other approaches in the literature, i.e., the convex programming approach proposed by Benati and Hansen (2002), the linearization technique proposed by Hasse (2009) and the B&B procedure by Freire et al. (2016), we can also conclude that MOA is competitive, generally improving over those other exact methods.

To have a closer look at the performance of the MOA algorithm when the number of cuts T varies, we take, from each data set, a representative problem. Then, we run MOA on the corresponding instances with different number of cuts per iteration (i.e., T), and we report the number of solved instances, computing times and number of iterations.

To be more precise, we take a problem with $|I| = |\mathcal{M}| = 100$ from HM14 instances, a problem with $|I| = 1000$, $|\mathcal{M}| = 100$ from ORlib and all instances of the NYC data set. Each problem from HM14, ORlib results in 81 instances, and the one from NYC results in 90 instances. We let T varies from 1 to 100 for the HM14 instances, $T \in \{1, 1000\}$ for the ORlib instances, and $T \in \{1, 300\}$ for the NYC ones. We also give a time limit of one hour to our MOA algorithm. Fig. 1 shows the number of solved instances, the average computing times and average number of iterations when T varies. The figure shows that when T increases, the average number of iterations decreases quickly, and the average number of solved instances increases for the HM14 and NYC instances. For the ORlib instances, the number of solved instances slightly decreases when $T > 500$. This is in line with the remark that when T is large, the master problem becomes expensive to be solved and MOA cannot converge to optimality within the time budget. For the HM14 instances, the computing times reduce quickly when T increases from 1 to 20. For the NYC instances, the algorithm achieves the best performance, in terms of computing time, with $T = 50$. We also notice that MOA performs differently for the ORlib instances, as the average computing times grow dramatically fast when $T > 1$. This implies that, for these instances, the single-cut version converges faster as compared to the MOA with $T > 1$.

In this paper, we focus on MNL instances but noting that a maximum capture problem under the MMNL model can be viewed as a problem under the MNL with extended customer zones (or clients), so the methods discussed above can be applied. Nevertheless, due to large numbers of demand points, MMNL instances would become (even more) challenging for the B&C and classical outer-approximation methods. Our clustering approach is a promising direction in the sense that it requires less cuts than the B&C method while being able to partially capture the separable structure of the objective function. However, the question of how to cluster the demand points remains not straightforward and we keep this for future work.

5. Conclusions

In the paper, we have proposed an enhanced implementation of the outer approximation scheme to solve the facility location maximum capture problem under random utilities. Our algorithm is based on the outer-approximation scheme but it allows to manipulate the number of subgradient cuts per iteration, instead of one cut per iteration as in a standard outer-approximation algorithm, or one cut per one demand point per iteration as in the state-of-the-art approach. Detailed computational experiments compare, for the first time, several variants of the outer approximation scheme allowing to shed light on the strength of the various ingredients, namely, single vs multi cut and Putting Plane vs Branch and Cut. The results show that our MOA algorithm is very competitive, often better than the state-of-the-art approach, especially on large instances. The MOA algorithm favorably compares with other outer-approximation based algorithms implemented in the BONMIN package, and our results indicate the robustness of our clustered implementation of the multicut version.

The MOA algorithm proposed here is not restricted to the maximum capture facility location problem with random utilities but can be used for any MINLP in which the objective function can be separated into several convex functions. Our results show that, by introducing more cuts per iteration, we can help an outer-approximation algorithm converge faster to optimality. This suggests that this multicut version could possibly be a good alternative to the single-cut one for other applications. Along with the above remarks, the idea of creating several cuts for separable convex functions would be also useful to handle separable convex nonlinear constraints as already observed by Hijazi, Bonami, and Ouorou (2013).

Acknowledgment

This work has been supported by the Canada Excellence Research Chair in Data Science for Real-Time Decision-Making. We thank Eduardo Moreno Araya and Ivana Ljubić for sharing their code and benchmark instances. The paper has been significantly improved thanks to the comments of three anonymous reviewers to which the authors are indebted.

References

Aros-Vera, F., Marianov, V., & Mitchell, J. E. (2013). p-hub approach for the optimal park-and-ride facility location problem. *European Journal of Operational Research, 226*(2)), 277–285.
Ben-Akiva, M., & Lerman, S. R. (1985). *Discrete choice analysis: Theory and application to travel demand*. Cambridge, Massachusetts: MIT Press.
Benati, S., & Hansen, P. (2002). The maximum capture problem with random utilities: Problem formulation and algorithms. *European Journal of Operational Research, 143*(3)), 518–530.
Berman, O., Drezner, T., Drezner, Z., & Krass, D. (2009). Modeling competitive facility location problems: New approaches and results. *Decision Technologies and Applications* (pp. 156–181). INFORMS's tutorial.
Birge, J. R., & Louveaux, F. V. (1988). A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research, 34*(3)), 384–392.
Bonami, P., Biegler, L. T., Conn, A. R., Cornuéjols, G., Grossmann, I. E., Laird, C. D., … Sawaya, N., et al. (2008). An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization, 5*(2)), 186–204.
Duran, M. A., & Grossmann, I. E. (1986). An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming, 36*(3)), 307–339.
Freire, A. S., Moreno, E., & Yushimito, W. F. (2016). A branch-and-bound algorithm for the maximum capture problem with random utilities. *European Journal of Operational Research, 252*(1)), 204–212.
Haase, K., & Müller, S. (2013a). Management of school locations allowing for free school choice. *Omega, 41*(5)), 847–855.
Haase, K., & Müller, S. (2013b). Management of school locations allowing for free school choice. *Omega, 41*(5)), 847–855.
Haase, K., & Müller, S. (2014). A comparison of linear reformulations for multinomial logit choice probabilities in facility location models. *European Journal of Operational Research, 232*(3)), 689–691.
Haase, K., & Müller, S. (2015). Insights into clients choice in preventive health care facility location planning. *OR Spectrum, 37*(1)), 273–291.
Haase, K., Müller, S., Krohn, R., & Hensher, D. (2016). The maximum capture problem with flexible substitution patterns. Hamburg University Working paper https://www.bwl.uni-hamburg.de/vw/service/stellen/ressource-stelle/workingpaper-simloc.pdf.
Hasse, K. (2009). Discrete location planning. Working paper, Institute of Transport and Logistics Studies.
Hijazi, H., Bonami, P., & Ouorou, A. (2013). An outer-inner approximation for separable mixed-integer nonlinear programs. *INFORMS Journal on Computing, 26*, 31–44.
Holguin-Veras, J., Reilly, J., & Aros-Vera, F., et al. (2012). New york city park and ride study. Technical report, University Transportation Research Center.
Huff, D. L. (1964). Defining and estimating a trading area. *The Journal of Marketing, 28*(3), 34–38.
Ljubić, I., & Moreno, E. (2018). Outer approximation and submodular cuts for maximum capture facility location problems with random utilities. *European Journal of Operational Research, 266*(1)), 46–56.
McFadden, D. (1973). Conditional logit analysis of qualitative choice behaviour. In P. Zarembka (Ed.), *Frontiers in econometrics* (pp. 105–142). New York: Academic Press.
McFadden, D. (1981). Econometric models of probabilistic choice. In C. Manski, & D. McFadden (Eds.), *Structural analysis of discrete data with econometric applications, chapter 5* (pp. 198–272). MIT Press.
McFadden, D., & Train, K. (2000). Mixed MNL models for discrete response. *Journal of applied Econometrics, 15*(5), 447–470.
Müller, S., Haase, K., & Kless, S. (2009). A multiperiod school location planning approach with free school choice. *Environment and Planning A, 41*(12), 2929–2945.
ReVelle, C. (1986). The maximum capture or sphere of influence location problem: Hotelling revisited on a network. *Journal of Regional Science, 26*(2), 343–358.
Train, K. (2003). *Discrete choice methods with simulation*. Cambridge University Press.
Wu, T. H. (1997). A note on a global approach for general 0–1 fractional programming. *European Journal of Operational Research, 101*(1)), 220–223.
Zhang, Y., Berman, O., & Verter, V. (2012). The impact of client choice on preventive healthcare facility network design. *OR Spectrum, 34*(2)), 349–370.