Maximizing submodular utility functions combined with a set-union operator over a discrete set

Stefano Coniglio · Fabio Furini · Ivana Ljubić

Received: date / Accepted: date

Abstract We study a discrete optimization problem calling for the maximization of a submodular, concave, and differentiable function $f$ combined with a set-union operator. The latter models a covering relationship between two ground sets, a set of items $N$ and a set of metaitems $\hat{N}$. The goal of the problem is to find an optimal subset of metaitems $\hat{S} \subseteq \hat{N}$ such that the total utility of the items they cover, as determined by $f$, is maximized. This problem, which is a generalization of the one introduced by Ahmed and Atamtürk [2011], can be modeled as a mixed integer nonlinear program involving binary decision variables associated with the items and metaitems in $N$ and $\hat{N}$.

In the paper, we propose a double-epigraph decomposition method which allows for projecting out the variables associated with the items in $N$ by separately exploiting the structural properties of the function $f$ and of the set-union operator. With this technique, the function $f$ is linearized using an outer-approximation technique, whereas the set-union operator is linearized in two ways: (i) via a reformulation based on submodular cuts, and (ii) via a Benders decomposition. We compare the strength of the two resulting mixed integer linear programming reformulations and embed them into two corresponding branch-and-cut algorithms. We compare them experimentally to a standard reformulation based on submodular cuts as well as to a state-of-the-art global-optimization solver. The results reveal that, on our testbed, the method based on combining an outer approximation with Benders cuts significantly outperforms the other ones.

Keywords Submodular maximization · Branch-and-Cut · Benders decomposition

1. Introduction

The problem of maximizing the expected value of a strictly concave, increasing, and differentiable utility function $f : \mathbb{R} \to \mathbb{R}$ over a discrete set plays an important role in many applications. Examples can be found in, among others, investment problems with discrete choices such as infrastructure projects, venture capital, and private equity deals [Ahmed and Atamtürk 2011], project selection [Klastorin 1990, Mehrez and Simany-Stern 1983, Wein- gartner 1966], competitive facility location [Berman and Krass 1998, Abdoilian et al. 2007],
Ljubić and Moreno [2018], and combinatorial auctions Feige et al. [2011], Lehmann et al. [2006]. In such utility-maximization problems, the function $f$ typically models the decision maker's risk-averse attitude.

Together with his coauthors, Shabbir Ahmed made substantial contributions in advancing the state-of-the-art of mathematical programming methods for solving this challenging set of problems Ahmed and Atamtürk [2011], Yu and Ahmed [2016, 2017]. In this article, we study combinatorial-optimization problems in which the function $f$ is maximized over a discrete set when combined with a set union operator, thus generalizing the problem addressed in the latter three articles.

1.1 The problem of Ahmed and Atamtürk [2011]

Let $N$ be a ground set of $n$ items and $M$ a finite set of $m$ scenarios, each occurring with probability $\pi_i$, for $i \in M$, and let $a_{ij} \in \mathbb{Z}^n_+$, $i \in M$, $j \in N$, and $d_i \in \mathbb{Z}_+$, $i \in M$, be two nonnegative integers. Letting the variable $x_j \in \{0, 1\}^n$ be equal to 1 if and only if item $j \in N$ is chosen, the problem tackled in Ahmed and Atamtürk [2011] can be cast as the following Mixed Integer Nonlinear Program (MINLP):

$$\max_{x \in \{0, 1\}^n} \left\{ \sum_{i \in M} \pi_i f\left( \sum_{j \in N} a_{ij} x_j + d_i \right) : x \in X \right\},$$

(1)

where $X$ is a polyhedron encapsulating the constraints of the problem and $X \cap \{0, 1\}^n$ is the discrete set of feasible solutions. From a stochastic-optimization perspective, Problem (1) calls for maximizing

$$\mathbb{E}_\xi\left[ f\left( \sum_{j \in N} a_j(\xi) x_j + d(\xi) \right) \right]$$

subject to $x \in X$, where $\xi$ is a random variable and the discrete realizations of $a_j(\xi)$, $j \in N$, and $d(\xi)$ are, respectively, $\{a_{1j}, \ldots, a_{mj}\}$ and $\{d_1, \ldots, d_m\}$.

The objective function of Problem (1) can be equivalently interpreted as a linear combination of $m$ set functions $h_i : 2^N \to \mathbb{R}^+$ with weights $\pi_i$, for $i \in M$. For each $i \in M$ and $S \subseteq N$, $h_i$ is defined as the composition of $f$ to an affine set function $q_i : 2^N \to \mathbb{R}$ which maps subsets of $N$ into the reals. Formally:

$$h_i(S) := f\left( \sum_{j \in S} a_{ij} + d_i \right), \quad i \in M, S \subseteq N.$$

For every $i \in M$, it is not difficult to see that, since $f$ is concave and increasing, the function $h_i$ is submodular, cf. Ahmed and Atamtürk [2011], Yu and Ahmed [2017], and so is the whole objective function of the problem. Thanks to this, Problem (1) can be reformulated as a Mixed Integer Linear Program (MILP) by exploiting a reformulation, originally due to Nemhauser et al. [1978], which features an exponential number of linear inequalities, one per subset $S$ of the ground set $N$. Ahmed and Atamtürk [2011] show how to tighten such inequalities by a sequence-independent sequential lifting procedure. In a follow-up article, Yu and Ahmed [2017] further improve the strength of such inequalities for the case where the set of discrete choices $X$ is subject to a single knapsack constraint. Further results on the nature of the exact lifting function adopted in Ahmed and Atamtürk [2011] are reported in Shi et al. [2020].
1.2 The generalized problem

In this work, we study a generalization of Problem (1) in which, besides the set of items $N$, we are given an additional ground set $\hat{N}$ of $\hat{n}$ metaitems. We assume that the two ground sets $N$ and $\hat{N}$ are linked by a covering relationship modeled by a bipartite graph $G = (\hat{N} \cup N, E)$ where, for each $j \in N$ and $\ell \in \hat{N}$, $\{j, \ell\} \in E$ if and only if item $j$ is covered by metaitem $\ell$. Throughout the paper, we denote by $e := |E|$ the size of the edge set of $G$, which corresponds to the number of covering relationships between items and metaitems. For each item $j \in N$, we define by $\hat{N}(j) \subseteq \hat{N}$ the set of metaitems of $\hat{N}$ by which $j$ is covered. Similarly, we define by $N(\ell) \subseteq N$ the set of items that are covered by the metaitem $\ell$, for each $\ell \in \hat{N}$. As a further generalization of Problem (1), we only assume $f$ to be concave, differentiable and nondecreasing (as opposed to concave, differentiable, and increasing, as done in Ahmed and Atamtürk [2011], Yu and Ahmed [2016]). An illustration of an instance of the problem is reported in Figure 1.

The key feature of this generalized problem is that, in order to take an item $j$ of $N$ and benefit from its profit, the decision maker has to select at least one among the metaitems $\ell \in \hat{N}$ which cover it.

Upon introducing a binary variable $y_\ell \in \{0, 1\}$ for each $\ell \in \hat{N}$, equal to 1 if and only if metaitem $\ell$ is chosen, the problem we study can be cast as the following MINLP:

$$
\max_{x \in \{0, 1\}^n} \sum_{i \in M} \pi_i f \left( \sum_{j \in N} a_{ij} x_j + d_i \right)
$$

subject to

$$
x_j \leq \sum_{\ell \in \hat{N}(j)} y_\ell \quad j \in N
$$

$$
y_\ell \leq x_j \quad \ell \in \hat{N}, j \in N(\ell)
$$

where $Y$ is a polyhedron encapsulating the constraints of the problem and $Y \cap \{0, 1\}^\hat{n}$ is the discrete set of feasible choices of metaitems. The parameters $\pi$, $a$, and $d$ have the
same meaning as in Problem (1). For each \( j \in \mathbb{N} \), the constraints linking the \( x \) and \( y \) variables imply that an item \( j \) belonging to the ground set \( \mathbb{N} \) is taken if and only if at least a metaitem \( \ell \) in \( \hat{\mathbb{N}} \) has been chosen from the set of metaitems \( \mathbb{N}(j) \subseteq \mathbb{N} \) that cover it. Note that, as the objective function of the problem is nondecreasing in \( x \), the constraints \( y_{xj} \leq x_j, \ell \in \hat{\mathbb{N}}, j \in \mathbb{N}(\ell) \), can be dropped due to being automatically satisfied in any optimal solution. Besides being binary, we do not assume further constraints on \( y \).

Problem (2) is a proper generalization of Problem (1). Indeed, the latter is obtained from the former by setting \( \hat{\mathbb{N}} := \mathbb{N} \) and defining the edge set \( \mathbb{E} \) as a matching, i.e., by introducing an edge for each pair of item \( j \in \mathbb{N} \) and metaitem \( \ell \in \hat{\mathbb{N}} \) having the same index.

Similarly to Problem (1), we can interpret the objective function of Problem (2) as a linear combination of \( m \) set functions \( \hat{h}_i : 2^{\hat{\mathbb{N}}} \to \mathbb{R}^+ \) with weights \( \pi, i \in \mathbb{M} \), each defined as the composition of \( f \) to an affine function \( \hat{q}_i : 2^{\hat{\mathbb{N}}} \to \mathbb{R} \):

\[
\hat{h}_i(\hat{S}) := f\left( \sum_{j \in N(\hat{S})} a_{ij} + d_i \right). 
\]

Differently from the function \( q_i \) occurring in Problem (1), which maps the chosen set of items \( S \) into its value, \( \hat{q}_i \) in Problem (2) maps the choice of metaitems \( \hat{S} \) into the total value of the items that \( \hat{S} \) covers. Indeed, \( \hat{q}_i \) can be equivalently defined as:

\[
\hat{q}_i(\hat{S}) := \sum_{j \in N(\hat{S})} a_{ij} + d_i \quad i \in \mathbb{M}, \hat{S} \subseteq \hat{\mathbb{N}}, 
\]

where

\[
N(\hat{S}) := \bigcup_{\ell \in \hat{S}} N(\ell), \quad \hat{S} \subseteq \hat{\mathbb{N}}
\]

is a set-union operator, expressing the covering relationship between \( \hat{\mathbb{N}} \) and \( \mathbb{N} \). The function \( \hat{q}_i \) occurs frequently in combinatorial optimization problems which feature a set covering component, and it has been heavily studied in the combinatorial optimization literature—see, e.g., Schrijver [2003], pag. 768.

Crucially, due to \( d_i \) and \( a_{ij} \) being nonnegative for all \( i \in \mathbb{M}, j \in \mathbb{N} \), \( \hat{q}_i \) is submodular in its input \( \hat{S} \subseteq \hat{\mathbb{N}} \) (rather than just affine, as it is the case for \( q_i \)). As \( f \) is strictly concave and increasing and \( \hat{q}_i \) is submodular, it is not difficult to see that their composition \( \hat{h}_i \) is submodular as well.

Problem (2) is \( \mathcal{NP} \)-hard even in the, arguably, simplest setting where \( Y \) only contains a single cardinality constraint, and even if one of the two submodular aspects of the problem is dropped. Trivializing \( f \) by defining it as the identity function \( f : z \mapsto z \) for all \( z \in \mathbb{R}^+ \), the decision version of the set covering problem can be reduced to Problem (2). This is because, for \( m = 1, a_{1j} = 1 \), for all \( j \in \mathbb{N} \), \( d_1 = 0 \), and \( Y = \{ \hat{S} \subseteq \hat{\mathbb{N}} : |\hat{S}| \leq b \} \), Problem (2) admits a solution of value \( n \) if and only if there is a feasible solution to the problem of covering \( \mathbb{N} \) with at most \( b \) subsets from \( \{ N(\ell) \}_{\ell \in \hat{\mathbb{N}}} \). Letting now \( G \) be the matching graph with \( \hat{\mathbb{N}} = \mathbb{N} \) and an edge between each pair \( \ell \in \hat{\mathbb{N}}, j \in \mathbb{N} \) having the same index, thereby trivializing the set-union operator, Problem (2) becomes identical to problem of maximizing a submodular function subject to a cardinality constraint, which is shown to be \( \mathcal{NP} \)-hard in [Nemhauser et al. 1978].

1.3 Applications

Problem (2) arises in different applications. An important one is found in the analysis of social networks, when identifying “key players” or “influencers” who can help to quickly distribute a piece of information through the network (Wu and Küçükyavuz [2018], Kempe et al. [1978]).
In this case, the set of metaitems \( \hat{N} \) corresponds to the set of influencers, the set of items \( N \) corresponds to the set of users of the social network, and, for each \( \ell \in \hat{N} \), \( N(\ell) \) is the set of users that can be reached by influencer \( \ell \) through a propagation mechanism. If the independent cascade model is chosen to model the propagation, the set \( M \) corresponds to possible propagation scenarios obtained by sampling (see, e.g., Güney et al. [2018]). The goal is then to choose a subset of influencers so as to maximize the expected number of users that can be reached. A binary \( a_{ij} \), for each \( i \in M, j \in N \), is used to denote whether the user of index \( j \) can be reached by influencer \( j \) under scenario \( i \).

A second application arises in marketing problems, where the set \( M \) models various products that a marketer is interested in advertising to a set \( N \) of potential customers, and \( \hat{N} \) is a set of marketing campaigns, where each \( \ell \in \hat{N} \) allows for reaching a subset \( N(\ell) \) of customers. In such case, the value of \( a_{ij} \), for each \( i \in M, j \in N \), corresponds to the demands of product \( i \) that is due to customer \( j \). Letting \( \pi_i \) be a weight measuring the relevance of a product \( i \in M \), the utility function \( f \) is used to express the decreasing marginal utility of reaching an additional unit of customer demand.

A third application arises in a class of stochastic competitive facility location problems with uncertain customer demands. In the deterministic competitive facility location problem (see, e.g., Ljubić and Moreno [2018] and further references therein), a subset of facilities has to be open subject to a budget constraint so that the captured customer demand is maximized (each customer can also choose to be served by the competitor instead). The set \( \hat{N} \) corresponds to a set of potential facility locations, whereas the customers are represented by the set \( N \). Using the multinomial logit model, for example, the market share function satisfies the properties of our function \( f \). In the stochastic setting, the expected market share is then calculated over the set \( M \) of possible scenario realizations (each with probability \( \pi_i \), \( i \in M \)), and \( a_{ij} \) corresponds to the demand of customer \( j \in N \) under scenario \( i \in M \).

### 1.4 Contribution and outline of the paper

Throughout the paper, we focus on the case with \( \hat{n} \ll n \), where \( n \) is a few orders of magnitude larger than \( \hat{n} \). This is the case of many applications, including those we introduced before. The main contribution of the paper is an exact method for solving Problem (2), based on a double-epigraph decomposition which allows for projecting out the variables associated with the items in \( N \). The method exploits the structural properties of the function \( f \) and of the set-union operator. In it, the function \( f \) is linearized using an outer-approximation method, whereas the set-union operator is linearized in two ways: (i) via a reformulation based on submodular cuts, and (ii) via a Benders decomposition. In particular, we show that the inequalities arising from the outer approximation combined with Benders decomposition can be separated in linear time even for fractional points. After comparing the strength of the two resulting mixed integer linear programming reformulations from a theoretical perspective, we embed them into two branch-and-cut algorithms. According to our computational experiments, the most efficient of them allows for solving to optimality instances of the problem with up to \( n = 20,000, \hat{n} = 65 \), and \( m = 100 \) in a short amount of computing time.

The paper is organized as follows. After recalling some background notions on submodularity in Section 2, we introduce our decomposition approach and our two problem reformulations in Section 3. The strength of the two reformulations we introduce is compared in Section 4. Section 5 presents an extension of our method allowing for a concave and differentiable function \( f \) which is not necessarily increasing. In Section 6, we demonstrate via computational experiments the advantages offered by our method based on linearization and variable projection. Concluding remarks are reported in Section 7.
2. Preliminaries

Before introducing the decomposition approach that we propose for solving Problem $\mathcal{P}$, we recall, in this section, the basic terminology and notions used in optimization problems involving submodular functions. We also summarize a key result due to Nemhauser et al. [1978], which leads to a direct MINLP reformulation of Problem $\mathcal{P}$.

Let $h : 2^N \rightarrow \mathbb{R}$ be a generic set function defined for some ground set $N$. The associated set function $\rho^i_h(S) := h(S \cup \{j\}) - h(S)$, $j \in N, S \subseteq N$, is called marginal contribution of $j$ with respect to $S$. The function $h$ is said nondecreasing if $\rho^j_h(S) \geq 0$ for all $j \in N, S \subseteq N$, and submodular if $\rho^j_h(S) \geq \rho^j_h(T)$ for all $S, T \subseteq N$ with $S \subseteq T$ and $j \in N$.

Every submodular set function $h$ enjoys the following properties:

**Proposition 1** (Nemhauser et al. [1978]) If $h$ is submodular, then:

\[
\begin{align*}
\rho^j_h(S) & \geq 0, \quad \text{for all } S \subseteq N, j \in N, \\
\rho^j_h(S) & \geq \rho^j_h(T), \quad \text{for all } S, T \subseteq N, S \subseteq T, j \in N.
\end{align*}
\]

By applying Proposition 1 to the function $\hat{h}_i$, $i \in M$, of Problem $\mathcal{P}$, we straightforwardly obtain the following MILP reformulation, which we refer to as Submodular Cuts Reformulation (SC-R):

\[
\begin{align*}
\max_{w \in \mathbb{R}^n, y \in Y \cap (0, 1)^n} & \quad \sum_{i \in M} \pi_i w_i \\
\text{s.t.} & \quad w_i \leq \hat{h}_i(\hat{S}) + \sum_{\ell \in N \setminus \hat{S}} \rho^h_i(\hat{S}) y_\ell - \sum_{\ell \in \hat{S}} \rho^h_i(\hat{N} \setminus \{\ell\})(1 - y_\ell) \quad \hat{S} \subseteq \hat{N}, i \in M \quad (\text{SC}_1) \\
& \quad w_i \leq \hat{h}_i(\hat{S}) + \sum_{\ell \in N \setminus \hat{S}} \rho^h_i(\emptyset) y_\ell - \sum_{\ell \in \hat{S}} \rho^h_i(\hat{S} \setminus \{\ell\})(1 - y_\ell) \quad \hat{S} \subseteq \hat{N}, i \in M \quad (\text{SC}_2)
\end{align*}
\]

The formulation is clearly correct as, for $\hat{S} = \{\ell \in \hat{N} : y_\ell = 1\}$, Constraints (SC1) and (SC2) boil down to $w_i \leq \hat{h}_i(\hat{S})$ for each $i \in M$, thereby guaranteeing $\sum_{i \in M} \pi_i w_i = \sum_{i \in M} \pi_i \hat{h}_i(S)$, which is equal to the objective function value of Problem $\mathcal{P}$.

We remark that the reformulation would still correct if we were to replace Constraints (SC1) by the following constraints:

\[
\begin{align*}
\rho^h_i(\hat{S}) y_\ell & \geq 0, \quad \hat{S} \subseteq \hat{N}, i \in M \quad (\text{SC}_3)
\end{align*}
\]

Such constraints are as strong as (SC1) unless there is an item $j \in N$ which is covered by a single metaitem $\ell \in \hat{N}$, i.e., such that $|\hat{N}(j)| = 1$. In such case, Constraints (SC1) dominate Constraints (SC3). This is because, since the function $\hat{h}_i$ is nonnegative and increasing, $\rho^h_i(\hat{N} \setminus \{\ell\})$ is always equal to 0 unless there is an item $j \in N$ such that $\hat{N}(j) = \{\ell\}$.

We extensively used in many works involving submodular functions, formulations similar to this one are known to be rather weak [Ahmed and Atamtürk 2011, Ljubić and Moreno 2018, Nemhauser et al. 1988]. Computational experiments obtained for it are reported in Section 4.
3. Decomposition approaches and MILP reformulations for Problem (2)

In spite of its weakness, an advantage offered by the SC-R is that it works in the $y$-space only. Since, as we mentioned, in this work we focus on cases where $n \gg n$, in this section we propose two techniques which, similarly to the SC-R, allow for solving Problem (2) in the $y$ space by projecting out the $x$ variables. As we will show in Section (6) with computational experiments, the reformulations arising from such techniques substantially outperform the SC-R on the testbed we consider.

The technique is based on a decomposition approach which allows for exploiting the structural properties of the function $f$ as well as those of the set-union operator underlying the function $\hat{q}_i, i \in M$. In particular, it allows for combining linear approximations introduced for each of them into a single MILP formulation to be solved by a branch-and-cut method by, at the same time, projecting out all the $x$ variables.

3.1 Double-epigraph decomposition

The key idea of our decomposition approach is reformulating Problem (2) in such a way that the function $h_i, i \in M,$ is decomposed into its two constituent parts, $f$ and $\hat{q}_i$.

Letting $y$ be the characteristic vector of $\hat{S} \subseteq \hat{N}$, the set function $\hat{q}_i(\hat{S}), i \in M,$ can be rewritten as the following vector function $\hat{Q}_i(y): \{0,1\}^n \rightarrow \mathbb{Z}_+$:

$$\hat{Q}_i(y) := \max_{x \in \{0,1\}^n} \left\{ \sum_{j \in N} a_{ij}x_j + d_i: \ x_j \leq \sum_{\ell \in N(j)} y_{\ell}, j \in N \right\} \quad i \in M. \quad (VF)$$

$\hat{Q}_i(y)$ can be interpreted as the value function of the problem of computing $\hat{q}_i(\hat{S})$ for a subset $\hat{S} \subseteq \hat{N}$ with incidence vector $y \in Y \cap \{0,1\}^n$. Such function is studied by Cordeau et al. [2019] in the context of covering facility location problems. Notice that the problem of computing $\hat{Q}_i(y)$ can be solved in linear time $O(c)$ by letting $x_j^* := 1$ for each $j \in N$ such that $\sum_{\ell \in N(j)} y_{\ell} \geq 1$, and letting $x_j^* := 0$ otherwise. Moreover, as the problem only features upper bounds on the variables, equal to $\min\{1, \sum_{\ell \in N(j)} y_{\ell}\}$ for each $j \in N$, its Linear Programming (LP) relaxation is integer. This property will be useful later.

After introducing the auxiliary variables $\eta_i, w_i, i \in M$, we consider the following double epigraph reformulation, which employs two epigraph reformulations applied in sequence:

$$\max_{w, y \in \mathbb{R}^m \cap \{0,1\}^n} \sum_{i \in M} \pi_i w_i$$

$$w_i \leq f(\eta_i) \quad i \in M \quad (EPI1)$$

$$\eta_i \leq \hat{Q}_i(y) \quad i \in M. \quad (EPI2)$$

This reformulation is correct as, due to $\pi_i \geq 0$ for all $i \in M$, $w_i = f(\eta_i)$ holds in any optimal solution. In turn, due to $f$ being nondecreasing this implies $\eta_i = \hat{Q}_i(y)$ in any optimal solution.

Starting from this decomposition, in the following we propose two MILP reformulations of Problem (2) belonging to the $(w, y)$ space and featuring $O(n + m)$ variables. Both reformulations are obtained by projecting out the $n$ decision variables $x$ associated with items $j \in N$, as well as the $m$ auxiliary variables $\eta$.

3.2 Projecting out the $\eta$ variables

The auxiliary variable $\eta_i, i \in M$, can be projected out as follows using Fourier-Motzkin elimination:
3.3 OA+SC MILP reformulation based on submodular cuts

Proposition 2 Constraints (EPI1)-(EPI2) can be replaced by the following constraints:

\[
\frac{w_i}{f'(p)} - \frac{f(p) - f'(p)p}{f'(p)} \leq \hat{Q}_i(y) \quad i \in M, p \in [0, \sum_{j \in N} a_{ij} + d_i].
\] (5)

Proof Since \( f \) is concave, for each \( i \in M \) the set of all pairs \((w_i, \eta_i) \in \mathbb{R}^2\) which satisfy Constraint (EPI1) forms a convex set. By means of an outer-approximation technique, we can restate Constraints (EPI1) as:

\[
w_i \leq f'(p)\eta_i + (f(p) - f'(p)p) \quad i \in M, p \in [0, \sum_{j \in N} a_{ij} + d_i],
\] (6)

where \( f' \) is the first derivative of \( f \) and \([0, \sum_{j \in N} a_{ij} + d_i] \) is a superset of the set of values that \( \eta_i \) (and, therefore, \( \hat{Q}_i \) and \( \hat{q}_i \)) can take. As \( f' > 0 \) due to \( f \) being nondecreasing, using Fourier-Mötzkin elimination we can combine Constraints (6) and (EPI2) to project out the \( \eta_i \) variables, which results in Constraints (5). ■

In the following two subsections, we discuss two ways of obtaining MILP reformulations of Problem (2) starting from Constraints (5). The first one, which we refer to as Outer Approximation plus Submodular Cuts Reformulation (OA+SC-R), exploits the submodularity of \( \hat{q}_i, i \in M \). The second one, which we refer to as Outer Approximation plus Benders Cuts Reformulation (OA+BC-R), relies on the integrality property of the LP relaxation of (VF), and it exploits LP duality in a Benders-cuts fashion. The idea is to combine Constraints (5) with a finite collection of affine functions yielding an over-estimation of \( \hat{Q}_i \).

3.3 OA+SC MILP reformulation based on submodular cuts

As \( \hat{q}_i \) is submodular, the following constraints are valid due to Proposition 1:

\[
\hat{Q}_i(y) \leq \hat{q}_i(\bar{S}) + \sum_{\ell \in \bar{S}} \rho^i_\ell(\bar{S})y_\ell - \sum_{\ell \in \bar{S}} \rho^i_\ell (\hat{N} \setminus \{\ell\})(1 - y_\ell) \quad \bar{S} \subseteq \hat{N} \quad (SC1)\]

\[
\hat{Q}_i(y) \leq \hat{q}_i(\bar{S}) + \sum_{\ell \in \bar{S}} \rho^i_\ell(\emptyset)y_\ell - \sum_{\ell \in \bar{S}} \rho^i_\ell (\hat{S} \setminus \{\ell\})(1 - y_\ell) \quad \bar{S} \subseteq \hat{N}. \quad (SC2)
\]

Combining these constraints with Constraints (5) of Proposition 2 we obtain the following MILP reformulation of Problem (2), which we refer to as Outer Approximation plus Submodular Cuts Reformulation (OA+SC-R):

\[
\max_{y \in Y \cap \{0,1\}^n} \sum_{i \in M} \pi_i w_i
\]

s.t.

\[
\frac{w_i}{f'(p)} - \frac{f(p) - f'(p)p}{f'(p)} \leq \hat{q}_i(\bar{S}) + \sum_{\ell \in \bar{S}} \rho^i_\ell(\bar{S})y_\ell - \sum_{\ell \in \bar{S}} \rho^i_\ell (\hat{N} \setminus \{\ell\})(1 - y_\ell) \quad \bar{S} \subseteq \hat{N}, i \in M, p := \hat{q}_i(\bar{S}) \quad (OA+SC1)
\]

\[
\frac{w_i}{f'(p)} - \frac{f(p) - f'(p)p}{f'(p)} \leq \hat{q}_i(\bar{S}) + \sum_{\ell \in \bar{S}} \rho^i_\ell(\emptyset)y_\ell - \sum_{\ell \in \bar{S}} \rho^i_\ell (\hat{S} \setminus \{\ell\})(1 - y_\ell) \quad \bar{S} \subseteq \hat{N}, i \in M, p := \hat{q}_i(\bar{S}). \quad (OA+SC2)
\]

Proposition 3 OA+SC-R is valid.

Proof Let \( y' \in Y \cap \{0,1\}^n \) be a feasible solution to Problem (2). Letting \( \hat{S}' := \{\ell \in \hat{N} : y'_\ell = 1\} \) for each \( i \in M \), the pair of Constraints (OA+SC1) and (OA+SC2) corresponding to \( \bar{S} = \hat{S}' \) boil down to \( w_i \leq f(\hat{q}_i(\hat{S}')) \), thereby guaranteeing \( \sum_{i\in M} \pi_i w_i = \sum_{i\in M} f(\hat{q}_i(\hat{S}')) \), which coincides with the objective function of Problem (2). ■
When solving the OA+SC-R with a branch-and-cut method, as we will do for the computational results reported in Section 3 to obtain a convergent method it suffices to look for the existence of violated constraints among (OA+SC1) and (OA+SC2) just for integer points $y^*$ which arise during the execution of the algorithm. The following result shows that such binary vectors can be separated very efficiently.

**Proposition 4** Given a vector $y^* \in \{0, 1\}^n$, the separation problem calling for a constraint among (OA+SC1)–(OA+SC2) that is strictly violated by $y^*$ can be solved in linear time $O(e)$.

**Proof** As observed in the proof of Proposition 3, Constraints (OA+SC1)–(OA+SC2) impose $w_i \leq f(\tilde{q}_i(S^*))$ for every set $S^* \subseteq N$. Letting $S^* := \{ \ell \in N : y^*_\ell = 1 \}$, it follows that, if $y^*$ is infeasible, the two constraints among (OA+SC1)–(OA+SC2) with $S = \hat{S}^*$ are violated for some $i \in M$. Given $S^*$, the coefficients in the right-hand side of Constraints (OA+SC1)–(OA+SC2) can be computed in linear time $O(e)$. In particular, the coefficients $\rho_i^N(N \setminus \{\ell\})$, $\rho_i^N(\emptyset)$ do not depend on $S^*$ and, hence, they can be precomputed. Computing the coefficients in the left-hand side is equivalently easy as, besides simple arithmetic operations, it only requires the evaluation of $f$ and its derivative $f'$ at $p = \tilde{q}_i(\hat{S}^*)$. ■

### 3.4 OA+BC MILP reformulation based on a Benders reformulation

Let us focus on the problem underlying the value function $\hat{Q}_i$, $i \in M$, defined in (VF). Assuming $y \in Y \cap \{0, 1\}^n$, every optimal solution to the LP relaxation of the problem of computing $\hat{Q}_i$ (in which $y \in \{0, 1\}^n$ is replaced by $y \in [0, 1]^n$) is binary. As the problem is bounded, we can consider its LP dual without loss of generality. Let $\pi$ and $\sigma$ be the dual variables associated with, respectively, the constraints linking the $x$ and the $y$ variables and the unit upper-bound constraints on $x$. Let

$$P_i := \{ (\pi, \sigma) \in \mathbb{R}^{n+n}_+ : \pi_j + \sigma_j \geq a_{ij}, j \in N \}, \quad i \in M,$$

be the polyhedron of the dual feasible solutions (for simplicity, we drop the index $i$ when referring to $(\pi, \sigma)$). By relying on LP duality, we obtain:

$$\hat{Q}_i(y) = d_i + \min \left\{ \sum_{j \in N} \pi_j \left( \sum_{\ell \in N(j)} y_{\ell} \right) + \sum_{j \in N} \sigma_j : (\pi, \sigma) \in P_i \right\}. \quad (8)$$

Letting $P_i^e$, $i \in M$, denote the set of extreme points of $P_i$, we derive the following *Benders cuts*:

$$\hat{Q}_i(y) \leq d_i + \sum_{j \in N} \tilde{\pi}_j \left( \sum_{\ell \in N(j)} y_{\ell} \right) + \sum_{j \in N} \tilde{\sigma}_j, \quad i \in M, (\tilde{\pi}, \tilde{\sigma}) \in P_i^e. \quad (BC)$$

Thanks to Proposition 2 we obtain the following alternative MILP reformulation of Problem (2), which we refer to as the *Outer Approximation plus Benders Cuts Reformulation* (OA+BC-R):

$$\max_{\mathbf{w} \in \mathbb{R}^n, y \in \text{int} \cap (0, 1)^n} \sum_{i \in M} \pi_i w_i$$

s.t. $\frac{w_i}{f'(p)} - \frac{f(p) - f'(p)p}{f'(p)} \leq d_i + \sum_{j \in N} \tilde{\pi}_j \left( \sum_{\ell \in N(j)} y_{\ell} \right) + \sum_{j \in N} \tilde{\sigma}_j$

$$i \in M, (\tilde{\pi}, \tilde{\sigma}) \in P_i^e, p \in [0, \sum_{j \in N} a_{ij} + d_i], \quad (\text{OA+BC})$$

where, as in the OA+SC-R, $[0, \sum_{j \in N} a_{ij} + d_i]$ is a superset of the image of $\hat{q}_i$, $i \in M$, thus being a superset of the domain of the function $f$. 

Notice that, even though the OR-BC-R is stated as a semi-infinite formulation, from a separation perspective we only need to impose $p = \hat{Q}_i(y)$ for it to be correct. For a given (not necessarily integer) $y^* \in [0, 1]^N$, optimal solutions to (8), which correspond to the so-called Benders subproblem, can be computed in closed form according to the following lemma:

**Lemma 1** A pair $(\pi, \sigma) \in \mathbb{R}^{n \times n}$ is an optimal solution to (8) if and only if it satisfies the following:

$$\begin{align*}
\pi_j &= a_{ij} \quad \sigma_j = 0 \quad \text{for } j \in N : \sum_{\ell \in \hat{N}(j)} y_{\ell} < 1 \\
\pi_j &= 0 \quad \sigma_j = a_{ij} \quad \text{for } j \in N : \sum_{\ell \in \hat{N}(j)} y_{\ell} > 1 \\
\pi_j &= \gamma_j a_{ij} \quad \sigma_j = (1 - \gamma_j) a_{ij} \quad \text{for some } \gamma_j \in [0, 1] \quad \text{for } j \in N : \sum_{\ell \in \hat{N}(j)} y_{\ell} = 1.
\end{align*}$$

**Proof** Problem (8) decomposes into $n$ subproblems, one per item $j \in N$. Each subproblem asks for a value $(\pi, \sigma)$ for $\pi_j$ and $\sigma_j$ satisfying $\pi_j + \sigma_j \geq a_{ij}$ at minimum cost ($\sum_{\ell \in \hat{N}(j)} y_{\ell} \pi_j + \sigma_j$). If $\sum_{\ell \in \hat{N}(j)} y_{\ell} < 0$, the coefficient of $\pi_j$ is smaller than the coefficient of $\sigma_j$ and, thus, we have $\pi_j = a_{ij}$ and $\sigma_j = 0$. If $\sum_{\ell \in \hat{N}(j)} y_{\ell} > 1$, the coefficient of $\pi_j$ is strictly larger than the coefficient of $\sigma_j$ and, thus, we have $\pi_j = 0$ and $\sigma_j = a_{ij}$. If $\sum_{\ell \in \hat{N}(j)} y_{\ell} = 1$, the coefficients of $\pi_j$ and $\sigma_j$ are identical and $\pi_j = \gamma_j a_{ij}, \sigma_j = (1 - \gamma_j) a_{ij}$ is optimal for every $\gamma_j \in [0, 1]$. 

We remark that, in the lemma, the latter case corresponds to the case where the Benders subproblem is (dual) degenerate, thus admitting multiple optimal solutions. Thanks to this, Lemma 1 generalizes the results given in Propositions 1, 2, and 4 in Cordeau et al. (2019).

Thanks to the lemma, the following holds:

**Proposition 5** For a given $w^*_i \in \mathbb{R}^+$ and a (not necessarily integer) $y^* \in [0, 1]^n$, the separation problem for Constraints (OA+BC) can be solved in linear time $O(e)$ for each $i \in M$.

**Proof** Constraint (OA+BC) is violated by $w^*_i$ and $y^* \in Y$ if and only if $w^*_i > f(\hat{Q}_i(y^*))$, which implies that $w^*_i$ is strictly larger than the outer approximation of $f$ at $p^* := \hat{Q}_i(y^*)$. Thanks to Lemma 1, the value of $\hat{Q}_i(y^*)$, as well as that of the dual multipliers $(\pi, \sigma)$, can be computed in linear time $O(e)$. 

A major advantage of the Benders-based reformulation OA+BC-R over the submodular-cuts-based reformulation OA+SC-R lies in the fact that the separation problem for Constraints (OA+BC) can be solved in polynomial (linear) time even when $y^*$ is fractional, a property which is not enjoyed by Constraints (OA+SC1) and (OA+SC2). From a branch-and-cut perspective, this allows for the efficient separation of Constraints (OA+BC) at each node of the branch-and-bound tree and after solving each LP relaxation, thus allowing for a tighter bound throughout the execution of the algorithm, including the root node.

4. On the strength of the two reformulations

In this section, we provide a theoretical comparison between the OA+SC-R and the OA+BC-R. For the purpose, we introduce $N_1 := \{j \in N : |\hat{N}(j)| = 1\}$, corresponding to the set of items in $N$ which can be covered by a single metaitem $\ell \in \hat{N}$, and we define $\ell(j)$ as item in $N$ covering it. Similarly, for every $\hat{S} \subseteq \hat{N}$, we let $N_1(\hat{S}) := \{j \in N(\hat{S}) : |\hat{N}(j)| = 1\}$ be the set of items in $N$ which can be covered by a single metaitem $\ell \in \hat{S}$.

Given that the left-hand side of the constraints featured in the OA+SC-R and the OA+BC-R are identical, without loss of generality we can focus on their right-hand sides. To establish the relationship between the submodular cuts (SC1), (SC2) and the Benders cut (BC), we first consider an additional submodular cut:

$$\hat{Q}_i(y) \leq q_i(\hat{S}) + \sum_{\ell \in N_1(\hat{S})} p_{\ell}^i(\hat{S}) y_{\ell} \quad i \in M, \hat{S} \subseteq \hat{N}. \quad \text{(SC3)}$$
Similarly what we observed in Section 2 when comparing \((SC3_3)\) to \((SC1_1), (SC3_4)\) and \((SC1_4)\) are equally strong if \(N_1 = \emptyset\). We can establish the following:

**Proposition 6** For each pair of \(i \in M\) and \(\hat{\mathcal{S}} \subseteq \hat{N}\), the submodular cut \((SC3_4)\) coincides with the Benders cut \((BC)\) obtained with dual multipliers defined as follows:

\[
\hat{\pi}_j = \begin{cases} 
    a_{ij} & \text{if } j \notin N(\hat{\mathcal{S}}) \text{ or } j \in N_1 \\
    0 & \text{otherwise}
\end{cases} \\
\hat{\sigma}_j = \begin{cases} 
    a_{ij} & \text{if } j \in N(\hat{\mathcal{S}}) \backslash N_1 \\
    0 & \text{otherwise}
\end{cases} \\
\text{for } j \in N. 
\] (9)

**Proof** Observe that, due to Lemma 1, \((\hat{\pi}, \hat{\sigma})\) is an extreme point belonging to \(P^e\), obtained by setting \(\gamma_j := 0\) for all \(j \in N\). After rearranging the terms in \((BC)\), we obtain:

\[
\hat{Q}_i(y) \leq d_i + \sum_{\ell \in \hat{N}} \left( \sum_{j \in N(\ell)} \hat{\pi}_j \right) y_\ell + \sum_{j < N} \hat{\sigma}_j.
\]

By the choice of multipliers in \((\hat{\pi}, \hat{\sigma})\), we have \(\sum_{j \in N} \hat{\sigma}_j = \sum_{j \in N(\hat{\mathcal{S}})} a_{ij}\). Thus, we have \(d_i + \sum_{j \in N} \hat{\sigma}_j = \hat{Q}_i(\hat{\mathcal{S}})\). Furthermore, for each \(\ell \in N\), the coefficient of \(y_\ell\) (i.e., \(\sum_{j \in N(\ell)} \hat{\pi}_j\)) is equal to \(\sum_{j \notin N(\hat{\mathcal{S}})} a_{ij}\), which corresponds to \(\hat{p}^{\hat{\mathcal{S}}}(\hat{\mathcal{S}})\), i.e., to the contribution of all items in \(N\) not covered by \(\hat{\mathcal{S}}\) but which would be covered by adding \(\ell\) to \(\hat{\mathcal{S}}\). The claim follows. \(\blacksquare\)

In the general case where \(N_1\) may be nonempty, \((SC3_4)\) are dominated by \((SC1_4)\). The following result shows that Constraints \((SC3_4)\) are contained in the OA+BC-R:

**Proposition 7** For each pair of \(i \in M\) and \(\hat{\mathcal{S}} \subseteq \hat{N}\), the submodular cut \((SC1_4)\) is equal to the Benders cut \((BC)\) with dual multipliers defined as follows:

\[
\hat{\pi}_j = \begin{cases} 
    a_{ij} & \text{if } j \notin N(\hat{\mathcal{S}}) \text{ or } j \in N_1 \\
    0 & \text{otherwise}
\end{cases} \\
\hat{\sigma}_j = \begin{cases} 
    a_{ij} & \text{if } j \in N(\hat{\mathcal{S}}) \backslash N_1 \\
    0 & \text{otherwise}
\end{cases} \\
\text{for } j \in N. 
\] (10)

**Proof** Observe that, as a consequence of Lemma 1, the pair \((\hat{\pi}, \hat{\sigma})\) corresponds to the extreme solution to the dual of the Benders subproblem defined in \((\hat{\mathcal{S}})\) that is obtained by setting \(\gamma_j := 1\) if \(j \in N_1\) and \(\gamma_j := 0\) otherwise. Due to \((10)\), we have:

\[
d_i + \sum_{j \in N} \hat{\sigma}_j = \sum_{j \in N(\hat{\mathcal{S}}) \backslash N_1} a_{ij} = \hat{Q}_i(\hat{\mathcal{S}}) - \sum_{j \in N(\hat{\mathcal{S}}) \cap N_1} a_{ij}
\] (11)

\[
\sum_{j \in N(\ell) \backslash N(\hat{\mathcal{S}})} a_{ij} = \sum_{j \in N(\ell) \cap N_1} a_{ij} \quad \ell \notin \hat{\mathcal{S}}
\] (12)

Equation \((11)\) is clearly correct. To see that \((12)\) is also correct, let us consider its two cases.

1. Assume \(\ell \notin \hat{\mathcal{S}}\). If \(N_1 = \emptyset\), due to \(\ell \notin \hat{\mathcal{S}}\) the weights \(a_{ij}\) of the items \(j \in N(\hat{\mathcal{S}})\) can be covered by their \(\pi_j\)’s only if such items are not covered by the other metaitems in \(\hat{\mathcal{S}}\). The expression is therefore correct as, among all items \(j \in N(\hat{\mathcal{S}})\), it ignores all those which are already covered (i.e., those belonging to \(N(\hat{\mathcal{S}})\)). If \(N_1 \neq \emptyset\), since \(\ell \notin \hat{\mathcal{S}}\) any item \(j \in N_1\) belonging to \(N(\ell)\) can only be covered by \(\ell\). Since \(\ell \notin \hat{\mathcal{S}}\), \(N(\hat{\mathcal{S}})\) does not contain \(j\). Thus, \(j \in N(\ell) \backslash N(\hat{\mathcal{S}})\), implying that its weight \(a_{ij}\) is covered.

2. Assume \(\ell \in \hat{\mathcal{S}}\). As all its items are covered, their \(\pi_j\)’s should all be zero and, thus, their \(a_{ij}\)’s do not have to be covered with the only exception of the items in \(N_1\), whose weight is correctly accounted for in \((12)\).
Thanks to (11) and (12), (BC) can be rewritten as:
\[
\hat{Q}_i(y) \leq \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \setminus N(\hat{S})} a_{ij} \right) y_\ell + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1} a_{ij} \right) y_\ell + \hat{q}_i(\hat{S}) - \sum_{j \in N(\hat{S}) \cap N_1} a_{ij}. \tag{13}
\]
Similarly to the proof of Lemma 6, the coefficient of \( y_\ell \) in the first term is the marginal contribution of \( \ell \in \hat{N} \) with respect to \( \hat{S} \). As the summation in the last term only considers items \( j \in N(\hat{S}) \) which can be covered by a single metaitem in \( \hat{N} \), we have:
\[
\sum_{j \in N(\hat{S}) \cap N_1} a_{ij} = \sum_{\ell \in \hat{S}} \sum_{j \in N(\ell) \cap N_1} a_{ij}.
\]
By reordering the terms of the last two summations in (13), we obtain:
\[
\hat{Q}_i(y) \leq \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \setminus N(\hat{S})} a_{ij} \right) y_\ell + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1} a_{ij} \right) (y_\ell - 1) + \hat{q}_i(\hat{S}).
\]
By noting that \( \rho_i^\hat{q}(\hat{N} \setminus \{\ell\}) = \sum_{j \in N(\ell) \cap N_1} a_{ij} \), the proof is concluded.

Proposition 7 shows that, for each \( i \in M \) and \( \hat{S} \subseteq \hat{N} \), Constraint (OA+SC1) coincide with the following (OA+BC) constraint:
\[
\frac{w_i}{f'(p)} - \frac{f(p) - f'(p)p}{f'(p)} \leq \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \setminus N(\hat{S})} a_{ij} \right) y_\ell + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1} a_{ij} \right) y_\ell + \hat{q}_i(\hat{S}) - \sum_{j \in N(\hat{S}) \cap N_1} a_{ij} \quad \hat{S} \subseteq \hat{N}, i \in M, p := \hat{q}_i(\hat{S}). \tag{OA+BC_1}
\]
The following proposition shows that the submodular constraint in (SC2a) corresponding to each pair that \( i \in M \) and \( \hat{S} \subseteq \hat{N} \) is strictly dominated by the Benders cuts in (BC) obtained according to Lemma 1 for \( y \) equal to the characteristic vector of \( \hat{S} \) and \( \gamma_j := 1 \) for all \( j \in N \).

Proposition 8 The submodular cut (SC2a) corresponding to each pair of \( i \in M \) and \( \hat{S} \subseteq \hat{N} \) can be tightened into the following strictly stronger (BC):
\[
\hat{Q}_i(y) \leq d_i + \sum_{j \in N(\hat{S}) \setminus N_1(\hat{S})} a_{ij} + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1(\hat{S})} a_{ij} \right) y_\ell + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1(\hat{S})} a_{ij} \right) y_\ell,
\]
which is obtained for the following choice of dual multipliers:
\[
\tilde{\pi}_j = \begin{cases} a_{ij} & \text{if } j \not\in N(\hat{S}) \text{ or } j \in N_1(\hat{S}) \\ 0 & \text{otherwise} \end{cases} \quad \tilde{\sigma}_j = \begin{cases} a_{ij} & \text{if } j \in N(\hat{S}) \setminus N_1(\hat{S}) \\ 0 & \text{otherwise} \end{cases} \quad j \in N.
\]

Proof Observe that \( \rho_i^\hat{q}(\emptyset) = \sum_{j \in N(\ell)} a_{ij} \) and \( \rho_i^\hat{q}(\hat{S} \setminus \{\ell\}) = \sum_{j \in N(\ell) \cap N_1(\hat{S})} a_{ij} \). Thus, (SC2a) can be rewritten as:
\[
d_i + \sum_{j \in N(\hat{S})} a_{ij} = \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1(\hat{S})} a_{ij} \right)(1 - y_\ell) + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell)} a_{ij} \right) y_\ell =
\]
\[
d_i + \sum_{j \in N(\hat{S}) \setminus N_1(\hat{S})} a_{ij} + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1(\hat{S})} a_{ij} \right) y_\ell + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell)} a_{ij} \right) y_\ell =
\]
\[
\geq d_i + \sum_{j \in N(\hat{S}) \setminus N_1(\hat{S})} a_{ij} + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1(\hat{S})} a_{ij} \right) y_\ell + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_1(\hat{S})} a_{ij} \right) y_\ell \geq \hat{Q}_i(y).
\]
The latter inequality corresponds to a Benders cut derived at one of the extreme points described in Lemma 1, obtained for the same \( i \in M \) and \( y \) equal to the characteristic vector of \( \hat{S} \), for \( \gamma_j = 1 \) for all \( j \in N \). This concludes the proof.

Proposition 8 shows that, for each \( i \in M \) and \( \hat{S} \subseteq \hat{N} \), Constraint (OA+SC2) can be strengthened into the following (OA+BC) constraint:

\[
\frac{w_i}{f'(p)} \cdot \frac{f(p) - f'(p)p}{f'(p)} \leq d_i + \sum_{j \in N(\hat{S}) \setminus N_i(\hat{S})} a_{ij} + \sum_{\ell \in \hat{S}} \left( \sum_{j \in N(\ell) \cap N_i(\hat{S})} a_{ij} \right) y_\ell 
+ \sum_{\ell \not\in \hat{S}} \left( \sum_{j \in N(\ell) \setminus N(\hat{S})} a_{ij} \right) y_\ell \quad \hat{S} \subseteq \hat{N}, i \in M, p := \hat{q}_i(\hat{S}) \quad \text{(OA+BC2)}
\]

In summary, letting \( P(OA+SC-R) \) and \( P(OA+BC-R) \) be the polytopes associated with the LP-relaxations of OA+SC-R and OA+BC-R, Propositions 7 and 8 imply the following:

**Proposition 9** \( P(OA+BC-R) \subseteq P(OA+SC-R) \), and there exist instances for which containment is strict.

5. Extension to the case where \( f \) is not nondecreasing

In this section, we discuss an extension of our methods to the case where \( f \) is concave and differentiable, but not necessarily nondecreasing. Such case arises in applications where, e.g., \( f \) is composed of two terms: a revenue term which increases with the amount of customers, with a positive decreasing derivative due to economies of scale, and a negative component which accounts for the total amount of money spent for running the service, assuming a unit cost per customer, decreasing with the number of customers. In such cases, once the system becomes overloaded and the optimal running capacity is overreached, \( f \) changes from being nondecreasing to being nonincreasing. As an example, consider the function

\[
f(z) := 1 - e^{-\lambda z} - \alpha z \quad \text{for} \quad \alpha > 0,
\]

where \( 1 - e^{-\lambda z} \) is the revenue term and \(-\alpha z\) is the cost term.

If \( f \) is not nondecreasing, it is not, in general, submodular and, therefore, the SCR and the submodular constraints in Proposition 1 are not valid. To account for this case, we can modify the double-epigraph decomposition we proposed in Section 3.1 as follows:

\[
\max_{w, \eta \in \mathbb{R}^m, y \in \mathcal{Y} \cap \{0, 1\}^n} \sum_{i \in M} \pi_i w_i \\
w_i \leq f(\eta_i) \quad i \in M \quad \text{(EPI1')}
\eta_i = \hat{Q}_i(y) \quad i \in M, \quad \hat{S} \subseteq \hat{N}.
\]

where the inequality sign “\( \leq \)” in Constraints (EPI2) has been replaced by the equality sign in Constraints (EPI2') due to the fact that \( f \) is not nondecreasing.

We can reformulate Constraints (EPI2') linearily thanks to the following result:

**Proposition 10** Constraints (EPI2') can be replaced by (SC1\( q \)) and (SC2\( q \)) together with the following constraints:

\[
\eta_i \geq \hat{q}_i(\hat{S}) + \sum_{\ell \in \hat{S}} \rho^\ell (\hat{N} \setminus \{\ell\}) y_\ell - \sum_{\ell \in \hat{S}} \rho^\ell (\emptyset)(1 - y_\ell) \quad i \in M, \hat{S} \subseteq \hat{N}.
\]

**Proof** To deal with the equations in Constraints (EPI2'), we need under- and over-estimators on the (submodular) function \( \hat{Q}_i(y) \). Over-estimators are given by Constraints (SC1\( q \)) and (SC2\( q \)).
To build an under-estimator, we rely on Proposition 2.1 of Nemhauser et al. [1978] which states that, for a submodular function \( \hat{q}_i \), the following inequality holds:

\[
\hat{q}_i(\hat{S}) \leq \hat{q}_i(\hat{T}) + \sum_{\ell \in \hat{S} - \hat{T}} \rho^\ell_i(\hat{S} \cap \hat{T}) - \sum_{\ell \in \hat{T} - \hat{S}} \rho^\ell_i(\hat{T} - \ell) \quad \hat{S}, \hat{T} \subseteq \hat{N}.
\]

Constraint (15) is obtained by substituting the marginals with the following upper and lower bounds

\[
\rho^\ell_i(\hat{S} \cap \hat{T}) \leq \rho^\ell_i(\emptyset); \quad \rho^\ell_i(\hat{T} - \ell) \geq \rho^\ell_i(\hat{N} - \ell),
\]

and encoding the set \( \hat{T} \) using the binary vector \( y \). Such constraints are tight at the point \( y \) corresponding to the characteristic vector of \( \hat{S} \).

We notice that the marginal contribution of \( \ell \) in the first summation of (15) is non-zero if and only if there exists an item \( j \in N(\ell) \) which is uniquely covered by \( \ell \), i.e., if and only if \( N(\ell) \cap N_j \neq \emptyset \). Hence, these inequalities act as big-M/no-good cut constraints that are binding at the given set \( \hat{S} \).

We can linearize Constraints (EP1) following an outer approximation method similar to the one of Section 3.2. Following the derivations in the proof of Proposition 2, we notice that Constraints (6) are valid only at points \( p \) where \( f'(p) > 0 \). For points \( p \) where \( f' < 0 \), these cuts should change their sign, leading to the following counterpart to Constraints (5):

\[
\frac{w_i}{f'(p)} - \frac{f(p) - f'(p)p}{f'(p)} \geq \eta_i, \quad i \in M, p \in [0, \sum_{j \in N} a_{ij} + d_i], \ s.t. \ f'(p) < 0. \tag{16}
\]

For any point \( p \) such that \( f'(p) = 0 \), we need to impose the following upper-bound constraints on \( w_i \):

\[
w_i \leq f(p), \quad i \in M, p \in [0, \sum_{j \in N} a_{ij} + d_i], \ s.t. \ f'(p) = 0. \tag{17}
\]

After applying Fourier-Mötzkin elimination to project out the \( \eta_i \) variables, we obtain a reformulation akin to the OA+SC-R of Section 3.3 featuring Constraints \( (SC1_1), (SC1_2) \), but only imposed for \( p \in [d_i + \sum_{j \in N} a_{ij}] \) such that \( f'(p) > 0 \), Constraint (17), and the following constraints:

\[
\frac{w_i}{f'(p)} - \frac{f(p) - f'(p)p}{f'(p)} \geq \hat{q}_i(\hat{S}) + \sum_{\ell \in \hat{S}} \rho^\ell_i(\hat{N} \setminus \{\ell\})y_\ell - \sum_{\ell \in \hat{S}} \rho^\ell_i(\emptyset)(1 - y_\ell) \quad \hat{S} \subseteq \hat{N}, i \in M, p \in [0, \sum_{j \in N} a_{ij} + d_i] \ s.t. \ f'(p) < 0. \tag{18}
\]

Finally, we point out that we can also combine Benders decomposition with the outer approximation obtaining a reformulation akin to the OA+BC-R of Section 3.4 even if \( f \) is not nondecreasing. Clearly, in this case constraints \( x_j \geq y_\ell, \ j \in N(\ell), \ell \in \hat{N} \) are not redundant and have to be appropriately integrated in the definition of the value function \( f \). For the sake of brevity, the derivation of the corresponding OA+BC cuts is left to the reader.

6. Computational results

In this section, we assess the effectiveness of the MILP reformulations proposed in the paper from a computational perspective. We experiment with the following branch-and-cut algorithms:

1. SC: a branch-and-cut algorithm based on the standard SC-R reformulation in Section 2 in which we generate the submodular inequalities \( (SC1_1) \) and \( (SC1_2) \) for the separation of integer points.
2. **OA+SC**: a branch-and-cut algorithm based on the OA+SC-R reformulation we presented in Section 3.3, in which we generate the inequalities (OA+SC1) and (OA+SC2) for the separation of integer points.

3. **OA+BC**: a branch-and-cut algorithm based on the OA+BC-R reformulation we presented in Section 3.4, in which we generate the inequalities (OA+BC1) and (OA+BC2) for the separation of integer points.

Since the inequalities (OA+BC) can be efficiently separated for fractional solution as well, we experiment with a fourth branch-and-cut algorithm:

4. **OA+BC+f**: a branch-and-cut algorithm similar to **OA+BC** in which inequalities (OA+BC) are also generated for the separation of fractional points. For each fractional point, we generate two Benders cuts, each corresponding to one of the optimal solutions to the dual of the Benders subproblem, as defined in Lemma 1. The first one is obtained by setting \( \gamma_j := 1 \) for all \( j \in \mathbb{N}_1 \) and \( \gamma_j := 0 \) for all \( j \in \mathbb{N} \setminus \mathbb{N}_1 \). The second one is obtained by setting \( \gamma_j := 1 \) for all \( j \in \mathbb{N} \). The separation is performed with a violation tolerance of \( 10^{-1} \).

Our experiments are carried out on a computer equipped with a 3.40 GHz 8-core Intel i7-3770 processor with 16 GB of RAM, running a 64-bit Linux operating system. Our code is written in C and compiled with gcc 9.2.1, using the -O3 compiler optimization flag. Our algorithms are implemented in CPLEX 12.9.0.0 (called just CPLEX for brevity in what follows), using the CALLABLE LIBRARY framework. CPLEX is run in single-threaded mode with all parameters set to their default values, except for Preprocessing_Linear and MIP_Strategy_CallbackReducedLP, which are set to 0 as recommended in the CPLEX user manual for the CPXsetusercutcallbackfunc and CPXsetlazyconstraintcallbackfunc callback functions (these functions are used for separating the constraints featured in our reformulations which are violated by fractional and integer points, respectively). We set a time limit of 600 seconds for each run.

In addition, we experiment with solving the directly using the formulation reported in (2) with BARON Sahinidis [2017], Tawarmalani and Sahinidis [2005], one of the state-of-the-art solvers for global optimization.

The main goals of this computational experience are: (i) comparing the performance of the four branch-and-cut algorithms and of BARON (see Sections 6.2 and 6.3), and (ii) testing the limits of these methods by identifying the size of the largest instances they can solve to proven optimality (see Section 6.4). In Section 6.5, we also measure the performance of our algorithms when applied for solving the problem of Ahmed and Atamtürk [2011], which, due to not containing a set-union operator, boils down to a pure outer-approximation method.

Before presenting our results in detail, we describe the instances used in our experiments.

### 6.1 Instances and settings of the experiments

As done in Ahmed and Atamtürk [2011], Yu and Ahmed [2016], we adopt, in our experiments, the negative exponential function

\[
f(z) = 1 - \exp\left(-\frac{z}{\lambda}\right),
\]

which is frequently used to model the behavior of risk-averse decision makers. Here, \( \lambda > 0 \) represents the risk-tolerance parameter, as larger values of \( \lambda \) account for a reduced risk-aversion attitude—for larger values of \( \lambda \), the function becomes closer to being linear.

To model the covering relationships between the sets of items \( \mathbb{N} \) and metaitems \( \hat{\mathbb{N}} \), we follow and extend the procedure proposed in Cordeau et al. [2019], ReVelle et al. [2008], where the sets \( \mathbb{N} \) and \( \hat{\mathbb{N}} \) are embedded in \( \mathbb{R}^2 \) and the covering relationship is spatially induced by associating a covering radius \( \hat{R} \) with each metaitem. The coordinates of
each item and metaitem in $N \cup \hat{N}$ are chosen uniformly at random from $[0, 30]$. We create a testbed of instances by setting $m := 10$ and choosing the values of $n$ and $\hat{n}$ from \{500, 1000, 5000, 10000\} and \{25, 30, 35\}, respectively. For each item $j \in N$, the set $\hat{N}(j)$ contains all the metaitems $\ell \in \hat{N}$ whose Euclidean distance from $j$ is not larger than the covering radius $R \in \{5.5, 5.75, 6, 6.25\}$. For each scenario $i \in M$ and item $j \in N$, the values of $a_{ij}$ are set to 1,000 with probability $\frac{1}{2}$, and to 0 otherwise. We consider a single cardinality constraint imposing that no more than 10 metaitems be selected. For each choice of $n$, $\hat{n}$, and $R$, we generate 5 instances adopting a different seed. For the utility function $f$, we consider two different values of $\lambda$, equal to 1 and 10. Overall, our testbed contains 480 instances.

6.2 Comparison of the different branch-and-cut algorithms and BARON

In Table 1, we compare the performance of the four branch-and-cut algorithms (SC, OA+SC, OA+BC, and OA+BC+f) and BARON. The table is vertically divided in five parts. The first part reports the instance features, i.e., the number of items $n$, the number of metaitems $\hat{n}$, and the value of $\lambda$ (selected in \{1, 10\}). Each row reports results aggregated over 20 instances with the same values of $n$, $\hat{n}$, $\lambda$ and obtained with 4 different values of the covering radius and 5 different random-generator seeds. For each of the five exact methods, we report the total number of instances (out of 20) solved to optimality within the time limit ($\neq$ opt) and the minimum (min), average (avg), and maximum (max) computing time in seconds. The aggregated values are computed by only considering instances which are solved by the corresponding method to optimality within the time limit. The label “t.l.” is reported when the time limit of 600 seconds is exceeded.

Let us first consider SC and OA+SC. SC fails to solve 5 instances (one with $n = 5,000$ and four with $n = 1,000$), whereas OA+SC fails to solve only a single instance (with $n = 1,000$). Besides, as expected, observing increased computing times for larger values of $n$ and $\hat{n}$, we also note that instances with $\lambda = 1$ tends to be more challenging when compared to the ones with $\lambda = 10$. The computational performance of OA+SC is only slightly better when compared to that of SC, showing that, on this testbed, the two methods are comparable. The two methods are able to solve all the instances with $n = 500$ with an average computing time no larger than 10 seconds. The average time more than doubles when increasing the number of items to $n = 1,000$. For the largest instances with $n = 10,000$, the average time increases by up to $\approx 150$ seconds.

As far as OA+BC and OA+BC+f are concerned, both algorithms are able to solve all the 480 instances. Computationally, the results show that OA+BC and OA+BC+f are superior to OA+SC, as the average and maximum computing time achieved by OA+BC and OA+BC+f are substantially smaller than those for OA+SC for all the instances. The computing time for both OA+BC and OA+BC+f is negatively influenced by an increased number of items and metaitems, but less so than for SC and OA+SC.

By looking at the computing time of OA+BC+f, we can see that a drastic improvement can be achieved by separating the OA+BC cuts also on fractional solutions found on the branching tree. In particular, OA+BC+f is able to solve all the 480 instances with an average computing time which is never larger than 1 second. The maximum computing time, of 3.6 seconds, is achieved on the instances with $n = 10,000$, $\hat{n} = 35$, and $\lambda = 1$.

As far as the performance of BARON is concerned, the table clearly shows that this method is computationally outperformed by all the other branch-and-cut algorithms. Indeed, BARON fails to solve 131 instances within the limit of 600 seconds. It is only able to solve all the instances with $n = 500$ and $n = 1,000$, while it fails on many instance with $n = 5,000$ and it is not able to solve any instance with $n = 10,000$. The computing times for the solved instances are also substantially larger when compared to the other methods, especially when compared to those for OA+BC+f.
Table 1: Computational performance of the four exact methods proposed in this paper and the global optimization solver BARON.

<table>
<thead>
<tr>
<th>n</th>
<th>( \tilde{n} )</th>
<th>( \lambda )</th>
<th>SC</th>
<th>OA+SC</th>
<th>OA+BC</th>
<th>OA+BC+f</th>
<th>BARON</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>time [s]</td>
<td>time [s]</td>
<td>time [s]</td>
<td>time [s]</td>
<td>time [s]</td>
</tr>
<tr>
<td>500</td>
<td>25</td>
<td>1</td>
<td>20</td>
<td>0.0</td>
<td>0.2</td>
<td>0.5</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>20</td>
<td>0.0</td>
<td>0.2</td>
<td>0.8</td>
<td>20</td>
<td>0.0</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>20</td>
<td>0.1</td>
<td>1.6</td>
<td>5.9</td>
<td>20</td>
<td>0.1</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>20</td>
<td>0.1</td>
<td>1.4</td>
<td>5.1</td>
<td>20</td>
<td>0.1</td>
</tr>
<tr>
<td>35</td>
<td>1</td>
<td>20</td>
<td>0.2</td>
<td>9.7</td>
<td>61.4</td>
<td>20</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>20</td>
<td>0.2</td>
<td>8.8</td>
<td>56.6</td>
<td>20</td>
<td>0.2</td>
</tr>
<tr>
<td>1,000</td>
<td>25</td>
<td>1</td>
<td>20</td>
<td>0.1</td>
<td>0.3</td>
<td>1.7</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>20</td>
<td>0.1</td>
<td>0.2</td>
<td>1.1</td>
<td>20</td>
<td>0.0</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>20</td>
<td>0.1</td>
<td>2.1</td>
<td>10.1</td>
<td>20</td>
<td>0.1</td>
</tr>
<tr>
<td>35</td>
<td>1</td>
<td>20</td>
<td>0.1</td>
<td>1.5</td>
<td>7.5</td>
<td>20</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>24</td>
<td>0.3</td>
<td>24.0</td>
<td>132.1</td>
<td>20</td>
<td>0.3</td>
</tr>
<tr>
<td>5,000</td>
<td>25</td>
<td>1</td>
<td>20</td>
<td>0.1</td>
<td>0.6</td>
<td>2.6</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>20</td>
<td>0.1</td>
<td>0.3</td>
<td>0.9</td>
<td>20</td>
<td>0.0</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>20</td>
<td>0.7</td>
<td>10.0</td>
<td>35.8</td>
<td>20</td>
<td>0.6</td>
</tr>
<tr>
<td>35</td>
<td>1</td>
<td>19</td>
<td>1.8</td>
<td>95.9</td>
<td>t.l.</td>
<td>20</td>
<td>2.7</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1.3</td>
<td>40.7</td>
<td>253.9</td>
<td>20</td>
<td>1.5</td>
<td>35.0</td>
</tr>
<tr>
<td>10,000</td>
<td>25</td>
<td>1</td>
<td>20</td>
<td>0.3</td>
<td>1.9</td>
<td>7.0</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>20</td>
<td>0.1</td>
<td>0.5</td>
<td>1.0</td>
<td>20</td>
<td>0.2</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>20</td>
<td>1.6</td>
<td>48.1</td>
<td>255.9</td>
<td>20</td>
<td>1.1</td>
</tr>
<tr>
<td>35</td>
<td>1</td>
<td>16</td>
<td>4.2</td>
<td>138.7</td>
<td>t.l.</td>
<td>19</td>
<td>7.6</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>21</td>
<td>54.2</td>
<td>424.7</td>
<td>20</td>
<td>2.3</td>
<td>46.3</td>
</tr>
</tbody>
</table>
The main drivers of the computing time are the number of items \( n \) and the number of metaitems \( \hat{n} \). To graphically see the impact of these two features on the performance of the four branch-and-cut algorithms, we report two box plots. In Figure 2, we report the computational times aggregating the instances by the number of items. The computing times, aggregated by number of metaitems, are instead reported in Figure 3. In these figures, we graphically represent the computing times (in logarithmic scale) through their quantiles. The lines extending vertically from the boxes indicate the variability outside the upper and lower quantiles. Finally, the outliers are plotted as individual points. As far as the number of items is concerned, Figure 2 shows that the times of all the branch-and-cut algorithms are negatively impacted by it. The best algorithm is \( \text{OA+BC+f} \), which is less affected then the other three. As far as the number of metaitems is concerned, Figure 3 shows that this feature has an even greater impact. \( \text{OA+BC+f} \) is the algorithm which is less affected by an increase in the number of metaitems. This fact proves that \( \text{OA+BC+f} \) scales much better than the other ones. It is worth noticing that, for \( \hat{n} = 35 \), the upper extreme of the quantile of \( \text{OA+BC+f} \) is much smaller than the lower extreme of the quantiles of the other three algorithms. This figure also shows that the difference in the computational performance between \( \text{OA+BC+f} \) and any of the other methods becomes larger when increasing the number of metaitems.

Finally we observe that, for larger values of \( \lambda \), the instances tend to become easier to solve (as evidenced by the smaller computing time), which is likely to be a consequence of the fact that, the larger \( \lambda \), the more the function \( f \) is close to be linear for a given range of the input. By analyzing the impact of the different covering radii on the computational performance, we can conclude that this feature only has a limited impact. For this reason, we do not report results for different values of \( \hat{R} \). In summary, Table 1 and the two box plots in Figures 2 and 3 computationally show that \( \text{OA+BC+f} \) achieves by far the best performance on this set of instances. The tests also show that \( \text{OA+BC+f} \) is computationally affordable even for larger values of \( n \) and \( \hat{n} \).

We conclude this section by showing, in Figure 4, two optimal solutions corresponding to instances with \( n \in \{500, 1000\} \), \( \hat{R} := 5.5 \), and \( m := 10 \). In these figures, the white circles represent the selected metaitems and the black circles the covered items. The colored circles surrounding a black one represent the coefficients \( a_{ij} \neq 0 (i \in M \text{ and } j \in N) \), i.e., the presence of an item \( j \in N \) in a specific scenario \( i \in M \). Both optimal solutions use 10 metaitems due to the cardinality constraints fixing the maximum number of selected metaitems to 10. The figure gives a graphical representation of the covering relationship between items and metaitems in the instances in the testbed.

6.3 Performance profile

A graphical representation of the relative computational performance of the four branch-and-cut algorithms (\( \text{SC} \), \( \text{OA+SC} \), \( \text{OA+BC} \), and \( \text{OA+BC+f} \)), as well as of \( \text{BARON} \), is given by the performance profile (see Dolan and Moré [2002]) reported in Figure 5. The figure clearly shows that \( \text{OA+BC+f} \) is the best algorithm for this set of instances, as it is able to solve all the 480 instances and it is the fastest algorithm in \( \approx 100\% \) of the cases. The second best algorithm is \( \text{OA+BC} \), which solves all the instances when allowing 100 times the computing time required by \( \text{OA+BC+f} \). This shows that separating the OA+BC cuts for fractional solutions is crucial for the efficiency of the method. The two branch-and-cut algorithms based on submodular cuts, i.e., \( \text{OA+SC} \) and \( \text{SC} \), have similar performance and they are both outperformed by \( \text{OA+BC+f} \).
Fig. 2 Impact of the number of items $n$ on the computing time.

Fig. 3 Impact of the number of metaitems $\hat{n}$ on the computing time.
Fig. 4 Two optimal solutions with \( n \in \{500, 1000\} \), \( \hat{n} = 25 \), \( m = 10 \). The covering radius of the dashed circles is \( R = 5.5 \).

by \( OA+BC+f \). As far as the number of instances solved is concerned, \( OA+SC \) solves all but 1 instance, while \( SC \) solves 474 instances, thereby solving 98% of the instances within the time limit of 600 seconds (but with a much larger computing time than \( OA+BC+f \)). Finally, \( BARON \) achieves the worse computational performance, as it is able to solve only 349 instances, i.e., 72% of the instances when it is allowed to run for 1000 times the computing time required by \( OA+BC+f \).

Fig. 5 Performance profile comparing the four branch-and-cut algorithms and \( BARON \).
6.4 Testing the limits of the branch-and-cut algorithms

In this section, we test the computational limits of the branch-and-cut algorithms using an additional set of larger instances, which are generated exactly as the ones presented in Section 6.1 but adopting larger values for the parameters. We consider \( n \in \{10000, 20000\} \), \( \hat{n} \in \{35, 50, 65\} \), and \( m \in \{50, 100\} \), using a single value for the risk-tolerance parameter \( \lambda \), letting \( \lambda := 10 \). As far as the covering radius is concerned, we used the same 4 values as before, generating 5 random instances with a different seed. This way, we obtain a second testbed of 240 larger instances.

In Table 2, we report the results of the two branch-and-cut algorithms \( \text{OA+BC+f} \) and \( \text{OA+SC} \). Similarly to Table 2, the first part of Table 2 reports the instance features, i.e., the number of items in column \( n \), the number of metaitems in column \( \hat{n} \), and number of scenarios in column \( m \). Each row of the table reports results aggregated over 20 instances, considering the 4 different values for the covering radius and the 5 different random seeds. The table is vertically divided into two parts, one containing the results for \( \text{OA+BC+f} \), and the other those for \( \text{OA+SC} \). For both of these two algorithms, we first report the number of instances solved in column \( \# \text{ opt} \). Then, we report the average exit gap computed as the percentage difference between the upper bound and the lower bound computed with respect to the upper bound achieved by the method when the time limit of 600 is reached (column gap%). Next, the table reports the average computing times in seconds considering only the instances which can be solved within the time limit (column time [s]). The last three aggregated values that are reported are: (i) the average number of nodes explored during by the branching tree (columns nodes); (ii) the average number of cuts generated when separating integer solutions (columns cuts); (iii) the average number of cuts generated when separating fractional solutions (column cuts (f), only for \( \text{OA+BC+f} \)). The averages are only computed considering instances which can be solved within the time limit, whereas, in each column, a "-" is reported if no instances are solved within the time limit.

In Table 2, we report the results of the two branch-and-cut algorithms \( \text{OA+BC+f} \) and \( \text{OA+SC} \). Similarly to Table 2, the first part of Table 2 reports the instance features, i.e., the number of items in column \( n \), the number of metaitems in column \( \hat{n} \), and number of scenarios in column \( m \). Each row of the table reports results aggregated over 20 instances, considering the 4 different values for the covering radius and the 5 different random seeds. The table is vertically divided into two parts, one containing the results for \( \text{OA+BC+f} \), and the other those for \( \text{OA+SC} \). For both of these two algorithms, we first report the number of instances solved in column \( \# \text{ opt} \). Then, we report the average exit gap computed as the percentage difference between the upper bound and the lower bound computed with respect to the upper bound achieved by the method when the time limit of 600 is reached (column gap%). Next, the table reports the average computing times in seconds considering only the instances which can be solved within the time limit (column time [s]). The last three aggregated values that are reported are: (i) the average number of nodes explored during by the branching tree (columns nodes); (ii) the average number of cuts generated when separating integer solutions (columns cuts); (iii) the average number of cuts generated when separating fractional solutions (column cuts (f), only for \( \text{OA+BC+f} \)). The averages are only computed considering instances which can be solved within the time limit, whereas, in each column, a "-" is reported if no instances are solved within the time limit.

Table 2 clearly shows that \( \text{OA+BC+f} \) outperforms \( \text{OA+SC} \) also on this set of instances, being able to solve 230 of the 240 instances. \( \text{OA+SC} \) is able to solve only 52 instances, and it already struggles to solve the smaller instances of this testbed. As far as the exit gap of \( \text{OA+SC} \) is concerned, we can observe that this value ranges from \( \approx 1.5\% \) to \( \approx 5\% \). \( \text{OA+SC} \) solves all the instances with \( n = 10,000 \) except for 3 of them (having \( \hat{n} = 65 \) and \( m = 100 \)) and it
solves all the instances with \( n = 20,000 \) except for 7 (having \( n = 65 \) and \( m = 50,100 \)). The table shows the limits of OA+BC+f in tackling instances with more than 65 metaitems. As expected, the number of scenarios affects the computational performance of both methods. As far as the exit gap of OA+BC+f is concerned, we can observe that the exits gap are very small and never larger that \( \approx 0.6\% \). The number of branching nodes explored in the tree is also much smaller for OA+BC+f than for OA+SC. This fact demonstrates that the OA+BC approach based on Benders cuts is more effective for this class of instances than the OA+SC method based on submodular cuts. In particular, the possibility of separating the OA+BC cuts exactly in linear time even for fractional solutions has a strong impact on the number of branch-and-cut nodes that are generated also for this testbed of larger instances.

6.5 Experiments when solving the problem of Ahmed and Atamtürk [2011]

In this section, we present the results of a pure outer approximation approach applied for the solution of Problem (1), studied in Ahmed and Atamtürk [2011], Yu and Ahmed [2017], which Problem (2) generalizes. To solve Problem (1) as a special instance of Problem (2), we set \( \hat{\pi} \) which Problem (2) generalizes. To solve Problem (1) as a special instance of Problem (2), we set \( \hat{\pi} \) which Problem (2) generalizes. To solve Problem (1) as a special instance of Problem (2), we set \( \hat{\pi} \). In the remainder of this section, we will denote the two exact methods we proposed when applied for the solution of Problem (1), studied in Ahmed and Atamtürk [2011], Yu and Ahmed [2017], the authors tackle a version of the expected utility with discrete choices problem

\[
\max_{x \in \mathbb{R}^m} \sum_{i \in M} \pi_i w_i \\
\text{s.t.} \quad \frac{w_i}{f'(p)} - \frac{f(p) - f'(p)p}{f'(p)} \leq \sum_{j \in N} a_{ij} x_j + d_i, \quad i \in M, p \in \{0, \ldots, \sum_{j \in N} a_{ij} + d_i\}.
\]

Notice that, given a fractional solution \( x^* \), the separation problem corresponding to these constraints is simply solved by setting \( p := \sum_{j \in N} a_{ij} x_j^* + d_i \).

In Ahmed and Atamtürk [2011], Yu and Ahmed [2017], the authors tackle a version of Problem (1) subject to the knapsack constraint \( \sum_{j \in N} b_j x_j \leq 1 \), where \( b_j, j \in N \), is the item weight and 1 is the knapsack capacity. Subadditive, sequence-independent lifted inequalities valid for the problem are introduced in Ahmed and Atamtürk [2011] by exploiting the submodularity of the function \( f \) and, in Yu and Ahmed [2017], the results are generalized to the case where the problem is subject to a knapsack constraint. In particular, Yu and Ahmed [2017] propose a branch-and-cut algorithm based on the separation of the uplifted inequality (9) reported on page 153 of Yu and Ahmed [2017] as well as on the separation of the downlifted inequality (9) reported on page 160 of Ahmed and Atamtürk [2011]. Extensive computational tests prove that the latter approach of Yu and Ahmed [2017] outperforms the previous one proposed of Ahmed and Atamtürk [2011]. It is important to mention that the separation problem for the new inequalities can be solved in polynomial time only for integer solutions, whereas, for fractional solutions, only an heuristic separation approach is presented. In the remainder of this section, we will refer to this approach as BnC-LI (where LI stands for lifted inequalities).

To test the performance of OA and OA+SC against the state-of-the-art BnC-LI of Yu and Ahmed [2017], we generate the same set of instances tested in Yu and Ahmed [2017], belonging to the following expected utility with discrete choices problem:

\[
\max_{x \in \{0,1\}^n} \left\{ \sum_{i=1}^m \pi_i \left( 1 - \exp \left( - \frac{\sum_{j \in N} a_{ij} x_j}{\lambda} \right) \right) : \sum_{j \in N} b_j x_j \leq 1 \right\}, \quad (20)
\]
which is a special case of (1) in which \( X \) features a single (scaled) knapsack constraint. We refer the reader to Ahmed and Atamtürk [2011] for a detailed description on how the input parameter of the problem are generated.

Along the lines of Yu and Ahmed [2017], we generate a set of instances with \( n \in \{100, 150, 200\} \), \( m \in \{50, 100\} \) and \( \lambda \in \{0.8, 1, 2\} \). For each value of these parameters, we create, as in Yu and Ahmed [2017], 20 random instances, thus obtaining a testbed of 240 instances.

In Table 3, we report the results of this final set of experiments comparing the performance of OA, BnC-LI, and BARON. We do not report results for OA+f as, computationally, the method achieves a performance that is comparable to that of OA. Similarly to the previous tables, the first part of Table 3 reports the instance features, with each row reporting aggregated results over the 20 random instances with identical characteristics. The table is vertically divided in three parts, one per solution method. The parts corresponding to the two branch-and-cut methods report the minimum, average, and maximum number of cuts, nodes, and computing time. The part corresponding to BARON only reports the computing times. While all three methods are able to solve to proven optimality all the considered instances, they exhibit a substantial difference in terms of performance.

As, unfortunately, the source code of BnC-LI is not available, the results of BnC-LI reported in Table 3 are directly taken from Table 1 reported on page 161 in Yu and Ahmed [2017], for which the authors used a Python code based on Gurobi 5.6.3, running on a 2.3 GHz x86 Linux workstation with 7 GB memory. Moreover, the results in Yu and Ahmed [2017] are obtained by halting Gurobi at a 0.01% optimality gap, whereas, in our experiments, CPLEX is run with the default tolerance of \( 10^{-6} \).

While the comparison on the computing times of the algorithms cannot be extremely precise, the table clearly shows the main trends of the results, especially when comparing the order of magnitude of the reported numbers. As far as the computing time is concerned, the table clearly shows that the best approach is OA, which is able to solve all the instances in an average time always smaller than 0.2 seconds, never exceeding 0.32 seconds. Notably, the average computing time of BnC-LI can be as high as several hundreds of seconds, with a maximum of about 800 seconds. We also notice that OA scales much better than BnC-LI for an increasing number of items \( n \), and that BnC-LI tends to struggle more for small values of \( \lambda \). The table also indicates that the number of cutting planes and branch-and-cut nodes for OA is smaller than that for BnC-LI. The difference is stronger for the harder instances with a larger number of items \( n \) and a smaller value of \( \lambda \). The maximum number of cuts for OA never exceeds 1,600, whereas the same value can be as high as 12,864 for BnC-LI. Moreover, the maximum number of nodes for OA never exceeds 1,400, whereas the same value can be as high as 7,031 for BnC-LI.

As to BARON, the computing times achieved with it are substantially larger than those for OA, up to two orders of magnitude larger, and as high as 15 seconds (whereas the computing times for OA never exceed 0.32 seconds).

These results show that, on the instances we considered, outer-approximation-based approaches can be very effective for solving Problem (20) when compared to methods based on the generation of lifted inequalities, as well as to general-purpose MINLP solvers such as BARON.

7. Conclusions

In this article, we have studied a generalization of a discrete maximization problem with a concave utility function originally studied by Ahmed and Atamtürk [2011], obtained by combining the concave and differentiable function with a set-union operator. We have introduced a double-epigraph reformulation for the problem that allowed us to decompose it into its two main components, one related to the concave utility function, and the other one related to the set-union operator. The two components are then linearized: the concave function is
Table 3: Computational results comparing OA, BnC-LI, and BARON when solving Problem (20).

<table>
<thead>
<tr>
<th>N</th>
<th>m</th>
<th>λ</th>
<th>cuts</th>
<th>time [s]</th>
<th>nodes</th>
<th>min</th>
<th>ave</th>
<th>max</th>
<th>min</th>
<th>ave</th>
<th>max</th>
<th>min</th>
<th>ave</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>50</td>
<td>2.25</td>
<td>3.34</td>
<td>0.02</td>
<td>0.05</td>
<td>0.10</td>
<td>0.55</td>
<td>1.77</td>
<td>3.46</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>50</td>
<td>2</td>
<td>2.73</td>
<td>0.02</td>
<td>0.05</td>
<td>0.10</td>
<td>0.55</td>
<td>1.77</td>
<td>3.46</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>1.5</td>
<td>4.06</td>
<td>0.03</td>
<td>0.09</td>
<td>0.15</td>
<td>0.96</td>
<td>3.80</td>
<td>11.73</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>150</td>
<td>50</td>
<td>1.8</td>
<td>5.14</td>
<td>0.03</td>
<td>0.10</td>
<td>0.18</td>
<td>1.28</td>
<td>3.62</td>
<td>7.46</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>50</td>
<td>1</td>
<td>6.21</td>
<td>0.03</td>
<td>0.12</td>
<td>0.27</td>
<td>2.11</td>
<td>3.95</td>
<td>7.23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>250</td>
<td>50</td>
<td>0.8</td>
<td>7.32</td>
<td>0.03</td>
<td>0.13</td>
<td>0.27</td>
<td>2.11</td>
<td>3.95</td>
<td>7.23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>50</td>
<td>0.6</td>
<td>8.43</td>
<td>0.03</td>
<td>0.14</td>
<td>0.27</td>
<td>2.11</td>
<td>3.95</td>
<td>7.23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>350</td>
<td>50</td>
<td>0.4</td>
<td>9.54</td>
<td>0.03</td>
<td>0.15</td>
<td>0.27</td>
<td>2.11</td>
<td>3.95</td>
<td>7.23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>50</td>
<td>0.2</td>
<td>10.65</td>
<td>0.03</td>
<td>0.16</td>
<td>0.27</td>
<td>2.11</td>
<td>3.95</td>
<td>7.23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

BARON

BnC-LI of Yu and Ahmed [2017]
replaced by its first order over-estimation, and the set-union operator is linearized either via submodular cuts or via a Benders decomposition, thus leading to two MILP reformulations featuring one or two exponential family of constraints. We have studied the relationship between the reformulations we proposed, providing theoretical results which compare the strength of the associated constraints. In our computational study, we have showed that a branch-and-cut approach based on Benders cuts combined with outer approximation drastically outperforms the alternative MILP reformulations as well as a state-of-the-art solver for mixed-integer nonlinear optimization. This is particularly true when tackling problems in which the number of items is many orders of magnitude larger than the number of metaitems. Finally, we have also provided necessary modifications to tackle the problem setting in which function $f$ is not nondecreasing.

Many open questions still remain to be addressed for this rich optimization problem. One way to linearize the problem is to work in the $(x, y)$-space and proceed along the lines of Ahmed and Atamtürk [2011] to “project $f$ out onto the $x$ variables”, which is a way of deriving outer-approximation cuts for an epigraph reformulation of the problem. Once the linearization is performed in the $(x, y)$-space, the $x$ variables could be projected out in a Benders-like fashion. Finally, one could obtain an alternative MILP formulation in the $y$-space by generating Generalized Benders Cuts [Geoffrion 1972, Fischetti et al. 2016] derived from the associated convex continuous subproblem. The connection between these different types of decomposition and linearizations and the strength of the respective MILP reformulations (both in the $(x, y)$- and in the $y$-space) remain an open topic for future research.

References


