Interdiction Games and Monotonicity

Matteo Fischetti [*1], Ivana Ljubić [†2], Michele Monaci [‡3], and Markus Sinnl[§4]

[1]*DEI, University of Padua, Italy.*
[2]*ESSEC Business School of Paris, France.*
[3]*DEI, University of Bologna, Italy.*
[4]*ISOR, University of Vienna, Austria.*

Abstract

Two-person interdiction games represent an important modeling concept for applications in marketing, defending critical infrastructure, stopping nuclear weapons projects or preventing drug smuggling.

We present an exact branch-and-cut algorithm for interdiction games, under the assumption that feasible solutions of the follower problem satisfy a certain monotonicity property. Prominent examples from the literature that fall into this category are knapsack interdiction, matching interdiction, and packing interdiction problems. We also show how practically-relevant interdiction variants of facility location and prize collecting problems can be modeled in our setting. Our branch-and-cut algorithm uses a solution scheme akin to Benders decomposition, based on a family of so-called interdiction cuts. We present modified and lifted versions of these cuts along with exact and heuristic procedures for the separation of interdiction cuts, and heuristic separation procedures for the other versions. In addition, we derive further valid inequalities and present a new heuristic procedure.

We computationally evaluate the proposed algorithm on a benchmark of 360 knapsack interdiction instances from literature, including 27 instances for which the optimal solution was not known. Our approach is able to solve each of them to optimality within about one minute of computing time on a standard PC (in most cases, within just seconds), and is up to 4 orders of magnitude faster than any previous approach from the literature. To further assess the effectiveness of our branch-and-cut algorithm, an additional computational study is performed on 144 randomly generated instances based on 0/1 multidimensional knapsack problems.

1 Introduction and Problem Definition

In many real-world optimization scenarios, a decision maker is not deciding alone, but has to make her decisions taking decisions of other parties into account. In its simplest form, such a decision process can be modeled as a *two-player Stackelberg game* (Von Stackelberg 1952). In such a game, there are two non-cooperating players, denoted as leader and follower, taking their decisions in a sequential way, i.e., in the first round the leader takes an action, and in the second round the follower reacts to it. Thereby, follower decisions are influenced by the leader who possesses a complete knowledge of the follower optimization setting.

Problems of this nature can be tackled via bilevel optimization, a problem class that received an increased attention in recent years, and was used to model important problems arising in real-world applications. Due to their relevance, increasingly effective general-purpose solvers have been designed very recently; see, e.g., the work of Moore and Bard (1990), DeNegre (2011), Xu (2012), Xu and Wang (2014), Zeng and An (2014), Kleniati and Adjiman (2015), Fischetti et al. (2016b,c) for mixed-integer linear bilevel problems.

[*]matteo.fischetti@unipd.it
[†]ivana.ljubic@essec.edu
[‡]michele.monaci@unibo.it
[§]markus.sinnl@univie.ac.at

In this article we consider a family of mixed-integer linear bilevel problems known as *interdiction games*. This family of problems covers important and diverse applications, such as critical infrastructure defense (Brown et al. 2005, 2006), stopping nuclear weapons projects (Brown et al. 2009, Morton et al. 2007), drug smuggling (Washburn and Wood 1995), military applications (Cormican et al. 1998), and marketing (DeNegre 2011); see also the surveys on *network interdiction* given by Smith and Lim (2008), Wood (2010), and Tang et al. (2015).

These problems can be seen as two-player zero-sum Stackelberg games where the leader and follower typically share a set of *items*, and the leader can select some items and *interdict* their usage by the follower. The adversarial nature of the game is expressed through the common objective function that is optimized in the opposite direction by the two players. Typically, connection between the leader and the follower optimization problems is established through binary decision variables ("interdiction variables") that are controlled by the leader. The only constraints in the follower subproblem involving leader decision variables impose that, if an interdiction variable is selected by the leader, then certain actions of the follower are inhibited. Very often these actions correspond to setting values of certain follower variables to zero, in which case a 1-1 correspondence between an interdiction leader variable and an interdicted follower variable exists.

More precisely, we focus on *Interdiction Games* (IGs) stated in the following form:

$$\min_{x\in X} \max_{y\in\mathbb{R}^{n_2}} \; d^T y \tag{1}$$

$$Q\,y \le Q_0 \tag{2}$$

$$0 \le y_j \le u_j(1-x_j), \qquad \forall j \in N \tag{3}$$

$$y_j \text{ integer}, \qquad \forall j \in J_y \tag{4}$$

where

$$X = \{x \in \mathbb{R}^{n_1} : Ax \le b,\ x_j \text{ integer } \forall j \in J_x,\ x_j \text{ binary } \forall j \in N\}$$

denotes the set of feasible leader solutions, and n_1 and n_2 are the number of leader variables x and follower variables y, respectively. We assume that d, Q, Q_0, u, A, b are given rational matrices/vectors of appropriate size. In particular, vector u provides finite upper bounds on the follower variables y_j involved in constraints (3).

The set N appearing in (3) will be called the *item set*, and it corresponds to the $n = |N|$ items subject to possible interdiction. Therefore, the interpretation of constraints (3) is that the leader can completely "forbid" an item $j \in N$ by setting $x_j = 1$, but if she does not do so, then an arbitrary number of these items (up to u_j) can be taken by the follower. Set J_x identifies instead the non-empty subset of indices of the integer-constrained variables in x, among which those in $N \subseteq J_x$ identify the indices of interdiction variables, that are assumed to be binary.

As to the follower, her variable set $\{1, \ldots, n_2\}$ is partitioned into (N, R), where R denotes the indices of the y variables that are not directly linked to x variables via constraints (3). Observe that the inner maximization problem over y (namely, the follower problem for a fixed x) can be either a Linear Program (LP) or a Mixed Integer Linear Program (MILP), depending on whether the set J_y of follower integer-constrained variables is empty or not. Also note that we do not require $N \subseteq J_y$, i.e., interdicted follower variables y_j's with $j \in N$ are not necessarily required to be integer, while the corresponding x_j's must be binary.

Our model generalizes previous proposals from the literature, in that it allows for an "extended formulation" of the follower subproblem that makes use of the "additional variables" y_j with $j \notin N$. In the following, we will denote by $y_N = (y_j)_{j\in N}$ the vector containing only the variables that can be interdicted, and by $y_R = (y_j)_{j\in R}$ the vector of remaining decision variables at the follower level. When useful, we will also use notation $Q = (Q_N, Q_R)$ and $d^T = (d_N^T, d_R^T)$.

Whenever $x \in X$, we will say that x is a *feasible interdiction policy.* Given a feasible interdiction policy $\hat{x}$, we will say that $y \in \mathbb{R}^{n_2}$ is a *feasible follower solution for* x if y satisfies (2), (3), and (4). In addition, we will denote by $y^*(\hat{x})$ an optimal follower solution obtained by solving (1)-(4) for $x = \hat{x}$. We assume that variable bounds on x and y other than those in (3), if any, are explicitly included in constraints $Ax \le b$ and $Qy \le Q_0$, respectively. Notation A_j or Q_j will be used for the j-th column of matrix A or Q, respectively.

As customary, in what follow we will assume that the follower problem is feasible and bounded for any feasible interdiction policy x.

As observed above, IGs are a special case of more general bilevel optimization problems in which the leader and the follower take their decisions in a hierarchical fashion, but their own objective functions and the interplay between their decisions can be of a more general form. In interdiction games (as opposed to the more general bilevel optimization) there is no need to distinguish between the optimistic and pessimistic setting, since both players optimize the same objective function—but in the opposite direction.

The following *(downward) monotonicity* is an important assumption made throughout this article, that will be exploited for deriving a valid Branch-and-Cut (B&C) approach based on interdiction constraints.

Assumption 1 (Downward Monotonicity)**.** *If $\hat{y} = (\hat{y}_N, \hat{y}_R)$ is a feasible follower solution for a given x and $y' = (y'_N, \hat{y}_R)$ satisfies constraints* (4) *and $0 \le y'_N \le \hat{y}_N$, then y' is also a feasible follower solution for x.*

If all follower variables are binary and $R = \emptyset$, Assumption 1 implies that the family of sets $\mathcal{S} = \{S \subseteq N : Q\,\chi_S \le Q_0\} \subseteq 2^N$ defines an *independent system*, where χ_S denotes the 0/1 incidence vector of S. However, as shown in Section 2, there are many other classes of IGs that satisfy it.

Observe that the monotonicity assumption does not reduce the computational complexity of the problem, in that it is satisfied (among others) by the the knapsack interdiction problem, which has been shown to be Σ_2-hard by Caprara et al. (2013). Moreover, it has been shown by Dinitz and Gupta (2013) and Zenklusen (2010) that monotone IGs remain NP-hard, even when $|N| = n_1 = n_2$ and the follower problem is a pure LP, i.e., $J_y = \emptyset$.

Due to monotonicity, we will assume without loss of generality that $d_N > 0$; otherwise all variables y_j with $j \in N$ and $d_j \le 0$ could be fixed to zero and removed from the model. In addition, we will assume $Q_N \ge 0$ due to the following result.

Theorem 1. *Assumption 1 holds if and only if there exits a formulation* (2)-(4) *of the follower problem with $Q_N \ge 0$.*

Proof. The fact that the condition is sufficient is obvious. To show that it is also necessary, consider any formulation of the follower problem (2)-(4). We will prove the claim by showing that every negative entry in Q_N, if any, can be increased to zero to produce an alternative system that is not worse (as $y_N \ge 0$) than the original one. To this end, let $q_i^T y \le q_{i0}$ denote any inequality in $Q\,y \le Q_0$ with $q_{ih} < 0$ for a certain $h \in N$ (if any). We have to show that the improved inequality $\overline{q}_i^T y \le q_{i0}$ is valid for the follower problem, where $\overline{q}_i$ is obtained from q_i by setting $\overline{q}_{ih} = 0$ and leaving the other entries unchanged. Indeed, take any feasible follower solution $\hat{y}$, and let y' be obtained from $\hat{y}$ by setting $y'_h = 0$. Due to Assumption 1, y' is a feasible follower solution, hence $q_i^T y' \le q_{i0}$ holds. By construction,

$$\overline{q}_i^T \hat{y} = \sum_{j \ne h} \overline{q}_{ij} \hat{y}_j = \sum_{j \ne h} q_{ij} y'_j = q_i^T y' \le q_{i0}$$

hence $\overline{q}_i^T y \le q_{i0}$ is a valid inequality due to the arbitrariness of $\hat{y}$. □

Our Contribution In this paper, we study IGs under Assumption 1. Differently from other approaches from the literature, we allow for an extended formulation of the follower problem (i.e., for $R \ne \emptyset$), thus gaining extra flexibility in modeling practically relevant situations. Various examples of applications are discussed, including IGs for important network problems such as the facility location problem and the prize-collecting traveling salesman problem.

We propose a Benders-like algorithm in which the problem is reformulated as a single-level problem (with an exponential number of constraints called *interdiction cuts*) and all follower variables are projected out. We introduce a new family of interdiction cuts that generalize those given (without proof) by Caprara et al. (2016) and Ralphs (2015) for the special case $R = \emptyset$, giving a formal proof of their validity for general monotone IGs and showing that they are instead not valid for the non-monotone case. We then propose a procedure for lifting these cuts, along with a family of related cuts whose validity is based on certain integer disjunctions. We also introduce a family of new cuts exploiting dominances among items. For interdiction cuts, exact and heuristic separation procedures are designed, while for the other families of cuts we propose fast heuristic separation algorithms. Moreover, we present a fast primal heuristic procedure for quite general (not necessarily monotone) interdiction games. This heuristic turns out to be extremely effective on some classes of instances, as its execution within a pre-processing procedure dramatically reduces the computing time needed to prove optimality. In our computational study, we consider benchmark sets for the knapsack interdiction problem proposed by Caprara et al. (2016), DeNegre (2011) and Tang et al. (2015), and show that our algorithm significantly outperforms the specialized codes proposed in Caprara et al. (2016) and DeNegre (2011), as well as the state-of-the art approaches for interdiction games (Tang et al. 2015) and for general bilevel mixed integer programming (Fischetti et al. 2016b). We test 360 knapsack interdiction instances from the literature, and prove the optimality for all of them—including the 27 problems that were previously unsolved. Our algorithm needs at most 84 seconds for solving any of these instances on a standard PC (for only 4 of these 360 instances, it requires more than 10 seconds), thus outperforming previous approaches from literature by orders of magnitudes. In addition to the above knapsack interdiction instances from literature, we also generated 144 random instances based on 0/1 multidimensional knapsack problems, with the aim of analyzing the dependency of our approach on the number of leader and follower constraints. To the best of our knowledge, this is by far the largest computational study on interdiction games reported in the literature.

Outline In Section 2 we illustrate a number of important practical problems that can be modeled as interdiction games satisfying the monotonicity property. The basic idea of a Branch-and-Cut framework using interdiction cuts is provided in Section 3, where we also provide theoretical foundations for deriving modified/lifted interdiction cuts, as well as valid inequalities based on dominance criteria. In Section 4 we provide implementation details of our framework, including separation algorithms and a primal heuristic procedure. Finally, Section 5 reports our computational study, while Section 6 gives a short conclusion.

2 Applications

We briefly describe some relevant IGs that satisfy the monotonicity assumption, and therefore can be tackled by our proposed methods.

An important observation is that Theorem 1 requires the non-negativity condition to be satisfied only by the columns of the constraint matrix Q associated with follower variables that can be interdicted—the remaining columns being immaterial for what concerns the downward monotonicity property. This fact greatly extends the applicability of our results, as many practically-relevant interdiction variants of classical problems like the facility location and the prize-collecting traveling salesman problem (and other similar prize collecting problems) can be handled by our approach.

Knapsack Problems. A prominent example of an interdiction game that satisfies the monotonicity property is the *Knapsack Interdiction Problem* (KIP) studied by Caprara et al. (2016), DeNegre (2011) and Tang et al. (2015). The problem models a Stackelberg game in which both leader and follower own their private knapsacks with capacities a_0 and q_0 (say), and fill them by choosing items from a common item set N. Each item $j \in N$ has a positive profit d_j, and weights a_j and q_j in the leader and in the follower problem, respectively. In the first step, the leader chooses some of the items while respecting her own knapsack capacity (called *interdiction budget*). In the second step, the follower solves a 0/1 knapsack problem and selects some of the items that are not taken by the leader to maximize the profit while respecting her capacity constraint. The goal of the leader is to obtain the worst possible outcome for the follower. Using binary variables x_j and y_j to denote the items selected by the leader and the follower, respectively, KIP can be modelled as an

IG as follows:

$$\min_x \max_y \sum_{j \in N} d_j y_j \tag{5}$$

$$\sum_{j \in N} a_j x_j \leq a_0 \tag{6}$$

$$\sum_{j \in N} q_j y_j \leq q_0 \tag{7}$$

$$y_j \leq 1 - x_j, \qquad \forall j \in N \tag{8}$$

$$x_j, y_j \in \{0,1\} \qquad \forall j \in N \tag{9}$$

As mentioned in (DeNegre 2011), a typical application of this problem arises in marketing, when a company A dominates the market, and company B wishes to design a marketing campaign, while choosing the specific geographic regions to target, subject to the available budget. Whenever companies A and B target the same region, the marketing campaign of company B fails. Consequently, the goal of the hostile company A is to minimize the established benefit of company B. In (DeNegre 2011), the author solves KIP through a cutting plane procedure in which the problem is reformulated as a single level problem with an exponential number of constraints, to be separated on-the-fly by using disjunctive cut-generating LPs. In (Caprara et al. 2016), a problem-tailored approach is introduced; in this iterative MILP-based procedure, the lower and upper bounds are sequentially improved, until an optimal solution (or a given time limit) is reached. Finally, due to the simplicity of its definition, the knapsack interdiction problem is a commonly used benchmark for testing solvers for bilevel optimization as well. In (Tang et al. 2015), the authors propose three ideas for deriving a generic solver for interdiction games. A new generic solver for bilevel mixed-integer programs has been recently proposed in (Fischetti et al. 2016b). In both papers, KIP instances constitute an important part of the considered benchmark set.

Of course, the most natural generalizations of the knapsack problem, namely the multi-dimensional knapsack problem and the multiple knapsack problem, satisfy the monotonicity property as well.

Facility Location Problems. Many IGs considered in the previous literature assume that every variable at the follower level can be interdicted by the leader, i.e., that $R = \emptyset$. In the following, we illustrate an important application from marketing/facility location which requires existence of additional decision variables at the follower level that cannot be explicitly interdicted by the leader, but anyway contribute to the follower objective function. Assume there are two companies, say A (the leader) and B (the follower) that compete for the same set of customers. Let I be the set of available facilities and J the set of customers served by them. Assume that company A dominates the market (i.e., it has already established service facilities and all customers are currently served by A), and that company B wants to enter the market. For B, facility opening costs $f_i \geq 0$ need to be paid for each $i \in I$, and profit $p_{ij} \geq 0$ can be collected if customer $j \in J$ is served by the open facility $i \in I$. The leader can provide an incentive $a_j \geq 0$ to a customer $j \in J$, so as to convince her not to switch the service, but there is a limited (interdiction) budget $b > 0$ to do so. The follower aims to maximize its revenue, assuming that all customers that are not "interdicted" by the leader will switch to the follower (if it is able to provide the service). The revenue for company B is defined as the sum of collected profits minus the costs for opening the facilities. The leader defines the interdiction policy using binary variables (v, x). Each variable v_j takes value 1 iff customer $j \in J$ receives an incentive from company A. In addition, there are auxiliary variables x_{ij} that are set to one (for all $i \in I$) whenever customer j receives an incentive from A. The x variables are used to possibly interdict some y variables in the follower, where $y_{ij} = 1$ iff customer $j \in J$ is served by facility $i \in I$ of company B. Finally, the follower also defines variables z_i to denote the set of facilities that company B has to open. We obtain the following

IG formulation:

$$\min_{(x,v)} \max_{(y,z)} \sum_{i\in I} \sum_{j\in J} p_{ij} y_{ij} - \sum_{i\in I} f_i z_i \tag{10}$$

$$x_{ij} - v_j = 0, \qquad \forall i \in I, j \in J \tag{11}$$

$$\sum_{j\in J} a_j v_j \leq b \tag{12}$$

$$y_{ij} \leq 1 - x_{ij}, \qquad \forall i \in I, j \in J \tag{13}$$

$$y_{ij} - z_i \leq 0, \qquad \forall i \in I, j \in J \tag{14}$$

$$\sum_{i\in I} y_{ij} \leq 1, \qquad \forall j \in J \tag{15}$$

$$x_{ij},\, y_{ij} \in \{0,1\}, \qquad \forall i \in I, j \in J \tag{16}$$

$$v_j \in \{0,1\}, \qquad \forall j \in J \tag{17}$$

$$z_i \in \{0,1\}. \qquad \forall i \in I \tag{18}$$

Thus, in this example, allocation variables y_{ij} (that are the only ones that can be interdicted by the leader) do satisfy the monotonicity property, whereas the remaining variables at the follower level (z_i) contribute to the objective function, but are not subject to interdiction, and as such, do not need to satisfy the monotonicity property.

Prize Collecting Problems. There are many problems in literature, which are of a *prize collecting* type, including the *prize-collecting traveling salesman problem* (PCTSP) (Balas 1989, Bienstock et al. 1993, Balas 2007), the *prize-collecting Steiner tree problem* (Ljubić et al. 2006, Prodon et al. 2010, Fischetti et al. 2016a), and various variants of the *orienteering problem* (Vansteenwegen et al. 2011). For example, in the PCTSP we are given a complete graph $G = (V, E)$ with positive prizes p_i associated with nodes $i \in V$, and positive costs c_e with the edges $e \in E$. We are looking for a subset of nodes, such that the revenue, i.e., the difference between the sum of the prizes in this subset and the cost of a tour on it, is maximized. In other words, a salesperson is looking for the most profitable tour through a subset of nodes, taking into account both the prize she can collect from each visited client, and the travel cost between clients. Similar to the previous example, assume that there are two salespersons A and B, where A dominates the market, while B wants to enter it. Each client $i \in V$ only allows for one salesperson to collect its prize. Again, A can provide some incentive $a_i \geq 0$ to each client i not to switch to the competitor, subject to a limited budget $b \geq 0$. The goal of A is now to find the best subset of clients to offer incentives (i.e., to interdict), such that the revenue of B is minimized, assuming that all clients to which A offered the incentive do not want the service of B (i.e., even if B visits such a client in her tour, B does not collect its prize).

Let the leader binary variable x_i be 1 iff A interdicts $i \in V$, and let the follower binary variable y_i by 1 iff B collects the prize of client $i \in V$ (i.e., iff client i is visited by B and not interdicted by A). We also introduce additional follower binary variables $z^E_e = 1$ iff B travels on edge $e \in E$, and additional follower binary variables $z_i^V = 1$ iff B visits client $i \in V$. The described problem can then be modeled as follows:

$$\min_x \max_{(y,z^E,z^V)} \sum_{i\in V} p_i y_i - \sum_{e\in E} c_e z_e^E \tag{19}$$

$$\sum_{i\in V} a_i x_i \le b \tag{20}$$

$$y_i \le 1 - x_i, \qquad \forall i \in V \tag{21}$$

$$y_i - z_i^V \le 0, \qquad \forall i \in V \tag{22}$$

$$(z^E, z^V) \in F \tag{23}$$

$$x_i,\, y_i \in \{0,1\}, \qquad \forall i \in V \tag{24}$$

$$z_e^E \in \{0,1\}, \qquad \forall e \in E \tag{25}$$

$$z_i^V \in \{0,1\}. \qquad \forall i \in V \tag{26}$$

where F denotes the set of the feasible follower solutions, i.e., F contains the incidence vectors (z^E, z^V) of all simple cycles of G and of the corresponding visited nodes. (In a variant of the problem, all such cycles can be required to satisfy additional conditions; e.g., they must visit a certain "depot" node.)

Constraints (22) ensure that B can only collect the prize of client i if she visits i, while constraint (23) states follower feasibility in a generic way—this could be modeled, e.g., by using subtour elimination constraints; see, e.g., (Bienstock et al. 1993). The interdiction actions of A are modeled by (21). Note that, under the very reasonable assumption that edge costs c_e's satisfy the triangle inequality, the optimal follower cycle will only visit non-interdicted clients.

It is easy to see that variables y_i (which are the only interdictable follower variables) fulfill the monotonicity property, as the system $Q\,y \le Q_0$ is just (22) in this case, hence $Q_N = I \ge 0$.

Observe that above formulation gives a very general recipe to formulate interdiction problems fulfilling the monotonicity property, as the generic constraint (23) can be replaced with any other set of constraints; e.g., if one wants to consider the prize-collecting Steiner tree as the follower problem, constraint (23) can be replaced by a set of constraints specifying that (z^E, z^V) describes a suitable Steiner tree.

Other Problems. Other relevant problems from the literature that fall into the category of interdiction games under monotonicity are the set-packing interdiction problem (Dinitz and Gupta 2013), the maximum weight matching interdiction problem (Zenklusen 2010) and the maximum weight independent set interdiction problem (Bazgan et al. 2011). Observe that the independent set problem is an example of a *hereditary graph problem* (Halldórsson 2000). A graph is said to possess the hereditary property Π, if every subgraph induced by its node subsets also possesses the same property. For a property Π, the corresponding hereditary graph problem is then defined to find the maximum node-weighted subgraph satisfying Π. It is easy to see that feasible solutions to such a problem define an independent system (with respect to the nodes), thus node-interdiction variants of these problems fall within our setting.

3 Interdiction Cuts

In this section we first recall the idea of reformulating interdiction games as single-level problems with an exponential number of constraints, called *interdiction cuts*. This idea has been frequently used in the interdiction literature; see, e.g., the seminal paper by Israeli and Wood (2002) or the survey by Wood (2010). However, in most of the cases, the quality of derived cuts is not satisfactory, since large big-M coefficients (or indicator constraints) must be used. In the remainder of this section we demonstrate that big-M values can be avoided (resulting in much tighter interdiction cuts), under the assumption that the follower satisfies the monotonicity property. We then provide a counter-example that shows that these specific interdiction cuts are not valid if the monotonicity property is violated. We finally conclude this section by providing additional theoretical results for strengthening and lifting the basic form of interdiction cuts.

3.1 Single-Level Reformulation

For a given $x \in X$ we define the *value function* as follows:

$$\begin{aligned}
\Phi(x) = \max_{y \in \mathbb{R}^{n_2}} \ & d^T y && && (27)\\
& Q\,y \le Q_0 && && (28)\\
& 0 \le y_j \le u_j(1 - x_j), && \forall j \in N && (29)\\
& y_j \text{ integer}, && \forall j \in J_y && (30)
\end{aligned}$$

so that problem (1)-(4) can be restated in the $\mathbb{R}^{n_1+1}$ space as

$$\begin{aligned}
\min_{x \in \mathbb{R}^{n_1}, w \in \mathbb{R}} \ & w && && (31)\\
& w \ge \Phi(x) && && (32)\\
& Ax \le b && && (33)\\
& x_j \text{ integer}, && \forall j \in J_x && (34)\\
& x_j \in \{0, 1\}, && \forall j \in N. && (35)
\end{aligned}$$

Constraint (32) can be rewritten in the following different form, see, e.g., (Wood 2010). We consider an alternative formulation of the follower subproblem (27)–(30) in which interdiction constraints (29) are removed and a penalization term $-\sum_{j \in N} M_j x_j y_j$ is added to the objective function. For sufficiently large values of multipliers M_j, this penalty term guarantees that any optimal solution of the follower has $x_j y_j = 0 \ \ \forall j \in N$, no matter the choice of x. For a given x, the follower subproblem can then be rewritten as

$$\Phi(x) = \max\{d^T y - \sum_{j \in N} M_j x_j y_j : y \in Y\}, \qquad (36)$$

where

$$Y = \{y \in \mathbb{R}^{n_2} : Q\,y \le Q_0, \quad 0 \le y_j \le u_j \ \forall j \in N, \quad y_j \text{ integer } \forall j \in J_y\}.$$

Note that, using the reformulation above, the feasible space Y of the follower does not depend on the interdiction policy x anymore. Furthermore, for a given x, the objective function is linear, which means that its optimal solution corresponds to a vertex of conv(Y). Consequently, the follower subproblem can be restated as

$$\Phi(x) = \max\{d^T y - \sum_{j \in N} M_j x_j y_j : y \in \hat{Y}\}, \qquad (37)$$

where $\hat{Y}$ contains all extreme points of conv(Y).

One can therefore derive a reformulation of the interdiction game as a single-level MILP akin to Benders decomposition (with the follower variables y being projected out of the model), namely:

$$\begin{aligned}
\min_{x \in \mathbb{R}^{n_1}, w \in \mathbb{R}} \ & w && && (38)\\
& w \ge d^T \hat{y} - \sum_{j \in N} M_j x_j \hat{y}_j && \forall \hat{y} \in \hat{Y} && (39)\\
& Ax \le b && && (40)\\
& x_j \text{ integer}, && \forall j \in J_x && (41)\\
& x_j \text{ binary}, && \forall j \in N. && (42)
\end{aligned}$$

In the following, we refer to (39) as *interdiction cuts.*

The above reformulation projects y variables out from the model and allows for the application of a B&C procedure in which interdiction cuts are initially removed from the model, and then dynamically added through the following separation procedure: Given an optimal (possibly, fractional) solution (w^*, x^*) at the current B&C node, the follower subproblem is solved for $x = x^*$ to obtain an optimal point $y^* \in \hat{Y}$. If the current solution violates the interdiction cut (39) associated with $\hat{y} = y^*$, then this globally-valid cut is added to the current formulation; otherwise no interdiction cut needs to be generated for (w^*, x^*).

The single-level reformulation above has already been used in the literature within an iterative cutting plane procedure; see, e.g. the procedure called CP in (Caprara et al. 2016) or Israeli and Wood (2002) and Wood (2010). In all these approaches, however, every time a single interdiction cut is added, the current model is solved as a MILP, before the new cut is separated in a cutting-plane fashion.

3.2 Interdiction Cuts for Followers with the Property of Monotonicity

A crucial point for the effectiveness of the proposed reformulation is how to determine appropriate values for M_j's so as to guarantee tight lower bounds—the smaller these coefficients the better the formulation. The choice of M_j's is problem-dependent; see, e.g., (Wood 2010). For the KIP, it has been observed by Caprara et al. (2016) and Ralphs (2015) that the values can be set as $M_j = d_j$ for all $j \in N$, though no formal proof for this result has been stated explicitly. In the following, we prove validity of these tightened constraints, not only for the KIP, but for the broader family of interdiction games satisfying the property of monotonicity—allowing, in particular, for $R \neq \emptyset$.

Theorem 2. *Under Assumption 1, the following* interdiction cuts *are valid for* (31)-(35)*:*

$$w \geq \sum_{j \in R} d_j \hat{y}_j + \sum_{j \in N} d_j \hat{y}_j (1 - x_j), \quad \forall \hat{y} \in \hat{Y}. \tag{43}$$

Proof. Let $\hat{y} \in \hat{Y}$ and take any feasible solution (w, x) to (31)-(35). Define a follower solution $y' = (y'_N, \hat{y}_R)$ where $y'_j = \hat{y}_j(1 - x_j)$ for all $j \in N$. Because of Assumption 1, y' is a feasible follower solution for the given x, hence

$$w \geq \Phi(x) \geq d^T y' = d_R^T y'_R + d_N^T y'_N = \sum_{j \in R} d_j \hat{y}_j + \sum_{j \in N} d_j \hat{y}_j (1 - x_j),$$

as claimed. □

Note that the point $\hat{y} \in \hat{Y}$ in the theorem above does not depend on x, i.e., it does not have to satisfy any complementarity condition of the form (29). Furthermore, we observe that interdiction cuts (43) are valid not only for extreme points $\hat{y} \in \hat{Y}$, but also for any arbitrary point in Y.

It is worth observing that, in case $R \neq \emptyset$, the above proof remains valid even if Assumption 1 is relaxed as follows: "if $\hat{y} = (\hat{y}_N, \hat{y}_R)$ is a feasible follower solution for a given x and y'_N satisfies constraints (4) and $0 \leq y'_N \leq \hat{y}_N$, then there exists y'_R with $d^T y'_R \geq d^T \hat{y}_R$ such that $y' = (y'_N, y'_R)$ is a feasible follower solution for x".

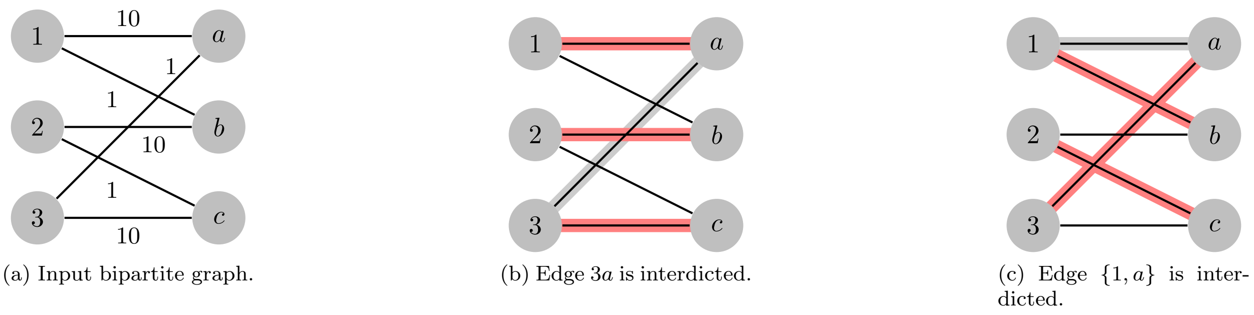
Theorem 3. *Under Assumption 1, interdiction game* (31)-(35) *can be reformulated by replacing constraint* (32) *with the family of (linear) interdiction cuts* (43).

Proof. Observe that there are exponentially many interdiction cuts (43). We have to show that, for any feasible interdiction policy x, these inequalities imply $w \geq \Phi(x)$. Indeed, the interdiction inequality for $\hat{y} = y^*(x)$ reads

$$w \geq \sum_{j \in R} d_j \hat{y}_j + \sum_{j \in N} d_j y_j^*(x)(1 - x_j) = \sum_{j \in R} d_j \hat{y}_j + \sum_{j \in N} d_j y_j^*(x) = \Phi(x)$$

where the first equality follows from the fact that, for all $j \in N$, $y_j^*(x) \cdot x_j = 0$ due to (29). □

Figure 1: (a) Example of the assignment interdiction problem. Weight of the horizontal edges is equal to 10, weight of the remaining edges is one. (b) and (c) show two possible solutions $y^*(x) \in \hat{Y}$ for two feasible interdiction policies $x \in X$: gray edges are interdicted by the leader, and red edges are chosen by the follower.



(a) Input bipartite graph.

(b) Edge $3a$ is interdicted.

(c) Edge $\{1, a\}$ is interdicted.

Definition 1. *A follower solution $\hat{y} = (\hat{y}_N, \hat{y}_R) \in \hat{Y}$ is* maximal *if there is no $(y'_N, \hat{y}_R) \in \hat{Y} \setminus \{\hat{y}\}$ such that $y'_N \geq \hat{y}_N$.*

The following result shows that, among all extreme points $\hat{y} \in \hat{Y}$, it is in fact sufficient to consider only maximal solutions. This fact can be computationally exploited to avoid the generation of useless interdiction cuts; see Section 4 for further details.

Theorem 4. *Let $\hat{y} = (\hat{y}_N, \hat{y}_R) \in \hat{Y}$ be nonmaximal and let $y' = (y'_N, \hat{y}_R) \in \hat{Y} \setminus \{\hat{y}\}$ be such that $y'_N \geq \hat{y}_N$. Then, under Assumption 1, the interdiction inequality (43) for $\hat{y}$ is dominated by that for y'.*

Proof. Obvious as, for all $j \in N$, $x_j \in [0, 1]$ implies $y'_j(1 - x_j) \geq \hat{y}_j(1 - x_j)$. □

In the following example we show that by dropping our assumption that the follower solutions satisfy the monotonicity property, the resulting interdiction cuts (43) are not valid. To this end, consider a problem instance in which the follower solves the maximum-weight assignment problem (i.e., a perfect matching on a bipartite graph), and the leader tries to minimize its outcome by interdicting some of the edges of the input bipartite graph.

Consider the graph depicted in Figure 1, and assume that the interdiction budget allows the leader to interdict at most one edge. If the leader interdicts edge $3a$, we have $\Phi(x) = 30$ and $y^*(x) = \chi_{\{1a,2b,3c\}}$. The resulting interdiction cut for $\hat{y} = y^*(x)$ would be $w \geq 30 - 10x_{1a} - 10x_{2b} - 10x_{3c}$, which is however violated by the feasible leader policy x' in which the leader interdicts edge $1a$ for which $y^*(x') = \chi_{\{1b,2c,3a\}}$ and $w' = \Phi(x') = 3$. Note that the above cut would instead be valid for a non-perfect variant of the problem allowing for isolated nodes—that would in fact satisfy the monotonicity property.

3.3 New Classes of Cuts

In this subsection we address the questions of how to modify the basic form of interdiction cuts (43) to derive further valid inequalities, and how to lift them (in a computationally inexpensive way, if possible), in order to improve the performance of the resulting B&C algorithm. We first propose a new class of modified interdiction cuts, then we introduce a lifting procedure for interdiction cuts, and finally we present a new family of cuts based on dominance relationships among items. For the validity of the new cuts, we impose an additional assumption:

Assumption 2. *All follower variables y_N are binary, i.e., $N \subseteq J_y$ and $u = 1$.*

Theorem 5. *For any $\hat{y} \in \hat{Y}$, let $S_a = \{a_1, \dots, a_K\} \subset N$ and $S_b = \{b_1, \dots, b_K\} \subset N$ be two distinct collections of items such that $\hat{y}_{a_k} = 1$, $\hat{y}_{b_k} = 0$, and $Q_{a_k} \geq Q_{b_k}$ for $k = 1, \dots, K$. Under Assumptions 1 and 2, the following* modified interdiction cut *is valid for* (31)-(35)*:*

$$w \geq \sum_{j \in R} d_j \hat{y}_j + \sum_{j \in N} d_j \hat{y}_j (1 - x_j) + \sum_{k=1}^{K} d_{b_k} (x_{a_k} - x_{b_k}). \tag{44}$$

Proof. For any given $\hat{y} \in \hat{Y}$, we have to show that (44) is satisfied by any given feasible interdiction policy x. In case $x_{a_k} - x_{b_k} \leq 0$ for each $k = 1, \dots, K$, this is obvious as x satisfies the interdiction inequality (43). Otherwise, let $\overline{\mathcal{K}} = \Big\{ k \in \{1, \dots, K\} : x_{a_k} - x_{b_k} = 1 \Big\}$, i.e., $x_{a_k} = 1$ and $x_{b_k} = 0$ for each $k \in \overline{\mathcal{K}}$. Consider the alternative follower solution y' obtained from $\hat{y}$ by flipping, for each $k \in \overline{\mathcal{K}}$, $\hat{y}_{a_k}$ and $\hat{y}_{b_k}$, i.e., by setting $y'_{a_k} = 0$ and $y'_{b_k} = 1$, and leaving the remaining entries unchanged. Under the assumption $Q_{a_k} \geq Q_{b_k}$, one has $Q\, y' \leq Q_0$, i.e., $y' \in \hat{Y}$ hence x satisfies the interdiction inequality associated with y', namely:

$$\begin{aligned} w &\geq \sum_{j \in R} d_j y'_j + \sum_{j \in N} d_j y'_j (1 - x_j) = \\ &= \sum_{j \in R} d_j \underbrace{y'_j}_{=\hat{y}_j} + \sum_{j \in N \setminus \{a_k, b_k : k \in \overline{\mathcal{K}}\}} d_j \underbrace{y'_j}_{=\hat{y}_j} (1 - x_j) + \sum_{k \in \overline{\mathcal{K}}} \Big(d_{a_k} \underbrace{y'_{a_k}}_{=0} (1 - x_{a_k}) + d_{b_k} \underbrace{y'_{b_k}}_{=1} (1 - x_{b_k}) \Big) \end{aligned} \tag{45}$$

Rewrite also (44) in a similar way to obtain

$$\begin{aligned} w \geq & \sum_{j \in R} d_j \hat{y}_j + \sum_{j \in N \setminus \{a_k, b_k : k \in \overline{\mathcal{K}}\}} d_j \hat{y}_j (1 - x_j) + \\ & \sum_{k \in \overline{\mathcal{K}}} \Big(d_{a_k} \underbrace{\hat{y}_{a_k}}_{=1} (1 - x_{a_k}) + d_{b_k} \underbrace{\hat{y}_{b_k}}_{=0} (1 - x_{b_k}) + d_{b_k} (x_{a_k} - x_{b_k}) \Big) + \\ & \sum_{k \in \mathcal{K} \setminus \overline{\mathcal{K}}} d_{b_k} (x_{a_k} - x_{b_k}). \end{aligned} \tag{46}$$

As (45) is a valid inequality and the left-hand side of both (45) and (46) are the same, it remains to be shown that the right-hand side of (46) is smaller or equal to the right-hand side of (45). To this end, it is enough to subtract the right-hand side of (46) from the right-hand side of (45) to obtain

$$\sum_{k \in \overline{\mathcal{K}}} \underbrace{\Big(d_{b_k} (1 - \underbrace{x_{b_k}}_{=0}) - d_{a_k} (1 - \underbrace{x_{a_k}}_{=1}) - d_{b_k} (\underbrace{x_{a_k} - x_{b_k}}_{=1}) \Big)}_{=0} - \sum_{k \in \mathcal{K} \setminus \overline{\mathcal{K}}} \underbrace{d_{b_k}}_{>0} \underbrace{(x_{a_k} - x_{b_k})}_{\leq 0} \;\geq\; 0. \tag{47}$$

□

As the above proof shows, the modified interdiction cuts (44) can be seen as disjunctive cuts based on the disjunctions $x_{a_k} - x_{b_k} \leq 0$ or ≥ 1, whose validity exploits the integrality of x. Note that, even if $d_{b_k} > 0$ by assumption, the additional terms $d_{b_k}(x_{a_k} - x_{b_k})$ in the right-hand side can be negative for some feasible x's, meaning that these cuts do not dominate (nor are dominated by) interdiction cuts.

Interdiction cuts can also be lifted by exploiting some further properties of Q, thus producing a new family of cuts that are strictly better (i.e., that dominate) the standard ones (43).

Theorem 6. *For a given $\hat{y} \in \hat{Y}$, let $S_a = \{a_1, \dots, a_K\} \subset N$ and $S_b = \{b_1, \dots, b_K\} \subset N$ be two distinct collections of items such that $\hat{y}_{a_k} = 1$, $\hat{y}_{b_k} = 0$, $d_{a_k} < d_{b_k}$, and $Q_{a_k} \geq Q_{b_k}$ for each $k \in \{1, \dots, K\}$. Under Assumptions 1 and 2, the following* lifted interdiction cut *is valid for* (31)-(35)*:*

$$w \geq \sum_{j \in R} d_j \hat{y}_j + \sum_{j \in N} d_j \hat{y}_j (1 - x_j) + \sum_{k=1}^{K} (d_{b_k} - d_{a_k})(1 - x_{b_k}). \tag{48}$$

Proof. We have to show that (48) is satisfied by any given feasible interdiction policy x. In case $x_{b_k} = 1$ for each $k = 1, \dots, K$, this is obvious as x satisfies the interdiction inequality (43). Otherwise denote by $\overline{\mathcal{K}} = \{k \in \{1, \dots, K\} : x_{b_k} = 0\}$ the subset of indices associated with items in S_b that are not selected in policy x.

Consider the alternative follower solution y' obtained from $\hat{y}$ by flipping, for each $k \in \overline{\mathcal{K}}$, $\hat{y}_{a_k}$ and $\hat{y}_{b_k}$, i.e., by setting $y'_{a_k} = 0$ and $y'_{b_k} = 1$, and leaving the remaining entries unchanged. Under the assumption $Q_{a_k} \geq Q_{b_k}$, one has $Q\,y' \leq Q_0$, i.e., $y' \in \hat{Y}$ hence x satisfies the interdiction inequality associated with y', namely:

$$
\begin{aligned}
w &\geq \sum_{j \in R} d_j y'_j + \sum_{j \in N} d_j y'_j (1 - x_j) = \\
&= \sum_{j \in R} d_j \underbrace{y'_j}_{=\hat{y}_j} + \sum_{j \in N \setminus \{a_k, b_k : k \in \overline{\mathcal{K}}\}} d_j \underbrace{y'_j}_{=\hat{y}_j} (1 - x_j) + \sum_{k \in \overline{\mathcal{K}}} \Big(d_{a_k} \underbrace{y'_{a_k}}_{=0} (1 - x_{a_k}) + d_{b_k} \underbrace{y'_{b_k}}_{=1} (1 - x_{b_k}) \Big)
\end{aligned} \tag{49}
$$

Rewrite also (48) in a similar way to obtain

$$
\begin{aligned}
w \geq & \sum_{j \in R} d_j \hat{y}_j + \sum_{j \in N \setminus \{a_k, b_k : k \in \overline{\mathcal{K}}\}} d_j \hat{y}_j (1 - x_j) + \\
& \sum_{k \in \overline{\mathcal{K}}} \Big(d_{a_k} \underbrace{\hat{y}_{a_k}}_{=1} (1 - x_{a_k}) + d_{b_k} \underbrace{\hat{y}_{b_k}}_{=0} (1 - x_{b_k}) + (d_{b_k} - d_{a_k})(1 - x_{b_k}) \Big) + \\
& \sum_{k \in \mathcal{K} \setminus \overline{\mathcal{K}}} (d_{b_k} - d_{a_k}) \underbrace{(1 - x_{b_k})}_{=0}.
\end{aligned} \tag{50}
$$

As (49) is a valid inequality and the left-hand side of both (49) and (50) are the same, it remains to be shown that the right-hand side of (50) is smaller or equal to the right-hand side of (49). To this end, subtract the right-hand side of (50) from the right-hand side of (49) to obtain

$$
\sum_{k \in \overline{\mathcal{K}}} \Big(d_{b_k} (1 - \underbrace{x_{b_k}}_{=0}) - d_{a_k} (1 - x_{a_k}) - (d_{b_k} - d_{a_k})(1 - \underbrace{x_{b_k}}_{=0}) \Big). \tag{51}
$$

For each $k \in \overline{\mathcal{K}}$, the corresponding term in (51) is zero if $x_{a_k} = 0$, while it is equal to $d_{a_k} > 0$ if $x_{a_k} = 1$. Thus, the sum is nonnegative, which concludes the proof. □

Notice that items in S_a and S_b may often be paired in different ways, still satisfying the requirements of the theorem above, thus producing different lifted inequalities; our specific recipe for their separation will be provided in the next section.

Finally, the following theorem introduces a new family of valid inequalities that exploits dominance relationships between pairs of items.

Theorem 7. *Let $i, s \in N$ be two distinct items such that $A_i \leq A_s$, $Q_i \leq Q_s$, and $d_i \geq d_s$. Under Assumptions 1 and 2, the following* dominance inequality

$$
x_s \leq x_i \tag{52}
$$

is satisfied by at least one optimal solution to problem (31)-(35).

Proof. We provide a constructive proof of the existence of an optimal solution of (31)-(35) that is not cut off by (52). Let x^* be an optimal solution that violates (52) (if any), i.e., such that $x^*_s = 1$ while $x^*_i = 0$. Define an alternative leader solution x' obtained from x^* by flipping its components indexed by $\{s, i\}$, namely

$$
x'_j = \begin{cases} x^*_j, & j \notin \{s, i\} \\ 0, & j = s \\ 1, & j = i \end{cases} \qquad j \in \{1, \dots, n_1\}.
$$

Solution x' clearly satisfies (52) and is feasible because of assumption $A_i \leq A_s$. It remains to be shown that x' is also optimal for (31)-(35), i.e., that $\Phi(x') \leq \Phi(x^*)$. To this end, let $y' = y^*(x')$ denote an optimal follower solution for x' (where $y'_i = 0$ as $x'_i = 1$) and define an alternative follower solution $\hat{y}$ obtained from y' by flipping its entries indexed by $\{s, i\}$ in case $y'_s = 1$, while $\hat{y} = y'$ otherwise. By definition, one has $\hat{y}_s = 0$ in both cases. In addition, because of assumption $Q_i \leq Q_s$, $\hat{y}$ is a feasible follower solution for x^*, hence

$$\Phi(x') = d^T y' \leq d^T \hat{y} \leq \Phi(x^*)$$

where the first inequality follows from assumption $d_s \leq d_i$. □

It is worth noting that there are only $O(|N|^2)$ dominance inequalities, so they can be statically added to the original model formulation without the need to design a run-time separation procedure. To avoid dominance loops, in case items i and s are identical (i.e., $A_i = A_s$, $Q_i = Q_s$ and $d_i = d_s$), we skip one of two inequalities—namely, that for $i < s$.

4 A Branch-and-Cut Approach for Monotone Interdiction Games

We have designed a B&C approach that works in the (w, x) space and dynamically adds the cuts described in the previous section. We next give implementation details about our approach.

4.1 Separation of Interdiction Cuts

Let (w^*, x^*) be the solution of the LP relaxation at a B&C node. The separation problem for (43) consists of solving the following problem:

$$\max\{\sum_{j \in R} d_j y_j + \sum_{j \in N} d^*_j y_j : y \in Y\}, \tag{53}$$

where $d^*_j = d_j(1 - x^*_j)$ for all $j \in N$. Let z^* be the optimal solution value of such a problem and let y^* be the solution found. If $w^* < z^*$, then y^* gives a maximally-violated interdiction cut (43), otherwise no violated cut exists.

Note that entries $x^*_j = 1$ produce zero-coefficients d^*_j in the objective function of the separation problem (53), possibly yielding an optimal solution y^* that is nonmaximal. In this case, there could be some other $y'_N \neq y^*_N$ with $y'_N \geq y^*_N$ and $y'_R = y^*_R$ which is an alternative optimal solution of the separation problem. According to Theorem 4, the interdiction cut associated with y' dominates the one associated with y^*. Thus, to favor maximal solutions, in our implementation we actually solve separation problem (53) with a perturbed objective function $\sum_{j \in N} d^*_j y_j$ where each $d^*_j = 0$ with $j \in N$ is replaced by $\epsilon\, d_j$ for a very small $\epsilon > 0$ ($\epsilon = 0.001$ was used).

In case the follower is a single (integer) knapsack problem, the separation problem can be solved using the well-known dynamic programming algorithm for knapsack problems (see, e.g., Martello and Toth 1990), running in pseudo-polynomial time. Otherwise, the separation problem is solved using a general purpose MILP solver. In both cases, separation is an NP-hard problem, which can make exact separation time consuming. However, the correctness of our approach requires to apply exact separation of interdiction cuts (43) only in case x^* is integer. For fractional x^*'s, in order to speed-up execution we heuristically solve the separation problem as follows. If the follower subproblem is a single knapsack problem, a simple greedy heuristic is applied (Martello and Toth 1990): items are ordered according to non-increasing values of d^*_i / q_{i0}, and a solution is constructed by collecting items until no more fit into the knapsack. In case the follower subproblem involves multiple constraints, instead, a general-purpose MILP solver is used and the run is interrupted after the root node is finished (if no feasible solution is found, no cut is added).

4.2 Separation of Modified Interdiction Cuts

We have implemented a heuristic to separate modified interdiction cuts (44). The heuristic takes on input the (possibly non-violated) interdiction cut produced by the separation routine described in Section 4.1, and tries to modify it to obtain a violated cut (44) in a greedy way. At each iteration, the next item that is candidate to enter set S_a is the item $a \in N$ with $\hat{y}_a = 1$ and maximum d_a. Given a, its "twin" item b is selected among those with $\hat{y}_b = 0$ and $Q_a \geq Q_b$ as the one with largest value $d_b(x_a^* - x_b^*)$: if $d_b(x_a^* - x_b^*) > 0$, items a and b are inserted into sets S_a and S_b, respectively, and then removed from any further consideration.

4.3 Separation of Lifted Interdiction Cuts

In our algorithm, lifted interdiction cuts are separated in a heuristic way as well. The separation procedure is very similar to the one described in the previous subsection to obtain a modified interdiction cut. Given an interdiction cut (43), we heuristically try to lift it to an inequality (48) in a greedy way. We scan the items $a \in N$ such that $\hat{y}_a = 1$ (that are the only candidate to be included in S_a), according to non-increasing d_a values. For each such item a, every item b with $\hat{y}_b = 0$ is checked for creating a possible lifting pair (a, b). More precisely, we scan all such items b with $\hat{y}_b = 0$, $d_b > d_a$ and $Q_a \geq Q_b$ (if any) and pick the one with minimum $d_j^* = d_j(1 - x_j^*)$ value. If such an item pair (a, b) is found, items a and b are inserted into sets S_a and S_b, respectively, and then removed from any further consideration. In preliminary computational tests, we experimented with alternative procedures for selecting the item pairs to lift, but the simple heuristic above turned out to be the most effective.

4.4 A Heuristic for General Interdiction Games

We next introduce a quite general heuristic for (possibly non-monotone) interdiction games, which is based on the idea of adding invalid leader constraints on the x variables that allow the optimal follower solution be expressed analytically as an a-priori linear function of x.

To be more specific, let $N^+ = \{j \in N : d_j > 0\}$ (recall that d_j can be nonpositive in the non-monotone case), and assume $R = \emptyset$, i.e., all follower variables y_j appear in a constraint (3). We introduce the invalid leader constraints

$$\sum_{j \in N^+} Q_j u_j (1 - x_j) \leq Q_0 \tag{54}$$

stipulating that *all* the non-interdicted items (those with $x_j = 0$) with positive profit d_j can be selected by the follower (at their highest-possible level) in a feasible solution. As a consequence, an optimal follower solution $y^*(x)$ always exists with

$$y_j^*(x) = \begin{cases} u_j(1 - x_j), & \text{if } j \in N^+, \\ 0, & \text{otherwise.} \end{cases}$$

The restricted interdiction game (i.e., problem (31)-(35) with the addition of constraints (54)) can therefore be reformulated *exactly* as the following (compact) single-level MILP

$$(HEU\_REF) \qquad \min \sum_{j \in N^+} d_j u_j (1 - x_j) \tag{55}$$

$$Ax \leq b \tag{56}$$

$$\sum_{j \in N^+} Q_j u_j (1 - x_j) \leq Q_0 \tag{57}$$

$$x_j \text{ integer}, \qquad \forall j \in J_x \tag{58}$$

$$x_j \in \{0, 1\}, \qquad \forall j \in N \tag{59}$$

where the y variables have been projected out as their optimal value is known. The above MILP can of course be infeasible. If this is not the case, its optimal solution provides a valid upper bound UB (say) for the original interdiction game, due to the obvious fact that the feasible solution set of (55)-(59) is a subset of that of the original problem (31)-(35) due to the addition of constraints (54).

If a finite UB is obtained, one can modify the original model (31)-(35) by adding the objective cutoff constraint

$$w \leq UB - \epsilon \tag{60}$$

for a sufficiently small $\epsilon > 0$ ($\epsilon = 1$ in case of integer d). In addition, one can impose a disjunction stating that at least one of the constraints in (54) must be violated. In our implementation, this is done through the following (possibly weak) single linear constraint

$$\sum_{j \in N^+} \max_i \{q_{ij}\} \, u_j (1 - x_j) \geq \min_i \{q_{i0} + \epsilon\} \tag{61}$$

where, in case $Q_0 > 0$, the single inequalities have been normalized to get $q_{i0} = 1$ for all i.

According to our computational experience, the addition of constraints (60) and (61) to the original model (31)-(35) often reduces solution time in a very significant way. This is true, in particular, when the resulting problem turns out to be infeasible, meaning that one is able to quickly prove that UB gives an optimal solution of the original interdiction game as well.

5 Computational Results

To assess the efficiency of our approach, we implemented it in Python, using the commercial solver IBM ILOG CPLEX 12.6 as underlying B&C framework. All CPLEX parameters were left at their default values in our runs, and a time limit of 3600 seconds for each run was set. The runs were made in sequential (single thread) mode on an Intel Xeon E3-1220V2 @3.1 GHz computer with 3GB of RAM.

5.1 Benchmark

We tested our approach on instances from the literature for the Knapsack Interdiction Problem (KIP instances) as well as on new instances with multiple leader and/or follower constraints based on multi-dimensional knapsack instances (MKIP instances). We decided instead not to address the other applications mentioned in Section 2, for which we expect that a customized code (still based on our interdiction cuts but using, e.g., a specialized preprocessing and a run-time separation procedure for subtour elimination constraints) is required to get the best-possible performance.

KIP instances from the literature Our first dataset includes the following 360 KIP instances from literature.

- Instance set `CCLW` has been introduced in Caprara et al. (2016). The follower data has been created using the knapsack-instance generator of Martello et al. (1999); profits d_i and weights q_i are uncorrelated integers in range $[0, 100]$, and the follower budget is set to $q_0 = \lceil \frac{INS}{10} \sum_{i \in N} w_i \rceil$, where INS is the number of the instance, with $1 \leq INS \leq 10$. The leader coefficients a_i are integers chosen uniformly random in $[0, 100]$, while the leader budget a_0 is taken from $[q_0 - 10, q_0 + 10]$. Ten instances are created for $|N| \in \{35, 40, 45, 50, 55\}$, for a total of 50 instances.
- Instance set `TRS` has been proposed by Tang et al. (2015). The interdiction budget is a cardinality constraint allowing at most k items to be interdicted. Item weights and profits are random integers from $[1, 100]$. Ten instances for pairs $(|N|, k)$ with $|N| \in \{20, 22, 25, 28, 30\}$ and three different values of k have been constructed, for a total of 150 instances.

- Instance set `D` has been introduced in DeNegre (2011). This class is based on bicriteria knapsack instances from the *multiple criteria decision making library*: the first objective of the bicriteria problem is used to define the follower objective function, while the second objective defines the interdiction budget constraint of the leader. The interdiction budget of an instance is $\lceil \sum_{i=1}^{n_1} a_i/2 \rceil$, where a_i is the cost of interdicting item i. The instances have $|N| \in \{10, 20, \dots 50\}$, with two additional sets with 11 and 12 items. For every number of items there are 20 instances, except for the 10-item case for which there are 40 instances. Thus, there are 160 instances in this class.

MKIP instances from SAC-94 library (Khuri et al. 1994) The SAC-94 library (Khuri et al. 1994) is a benchmark library containing 0/1 Multidimensional Knapsack Instances from Freville and Plateau (1990) (instances `hp*` and `pb*`), Petersen (1967) (instances `pet*`), Senju and Toyoda (1968) (instances `sento*`), Shih (1979) (instances `weish*`), and Weingartner and Ness (1967) (instances `weing*`). Starting with these 54 instances, we generated 144 new instances of the Multidimensional Knapsack Interdiction Problem (MKIP) as follows.

The instances have 2 to 30 constraints and 10 to 90 items. For each instance of this dataset, we constructed three different interdiction instances by considering

- the first constraint as leader constraint and the remaining constraints as follower constraints (these instances are denoted by `-0` in the name);
- the first 50% of constraints (rounded up) as leader constraints, and the remaining ones as follower constraints (denoted by `-50`);
- all but the last constraint as leader constraints (denoted by `-100`).

Thus, in the `-0` and `-50` instances, the follower problem is a multidimensional knapsack problem, while instances of type `-100` have a single knapsack as follower problem. Moreover, in instances of type `-50` and `-100`, there are multiple leader constraints. Of course, when the underlying multidimensional knapsack instances have just two constraints, all three transformations give the same instance with one leader and one follower constraint, i.e., a single knapsack as follower problem. These instances are `weing*` and instance `pb4`. Thus, we obtained 54 instances of type `-100` and 45 instances of type `-0` and `-50` for a total of 144 instances. Details on the number of variables and of leader/follower constraints and on the obtained optimal solution are presented in Tables 4-6. All instances are available online at `http://homepage.univie.ac.at/markus.sinnl/program-codes/bilevel/`.

5.2 Analyzing the Influence of the Individual Ingredients

In order to asses the influence of the various ingredients proposed in our framework, we tested six different settings of our B&C code:

`-` : this is our *basic setting* in which only basic interdiction cuts (43) are separated using the exact algorithm;

`M` : as before, with the addition of the heuristic separation for Modified interdiction cuts described in Subsection 4.2;

`MH` : as before, but using the Heuristic separation procedure for interdiction cuts (43) described in Subsection 4.1 (instead of the exact separation algorithm);

`MHD` : as before, but all Dominance inequalities (52) are statically added to the initial model;

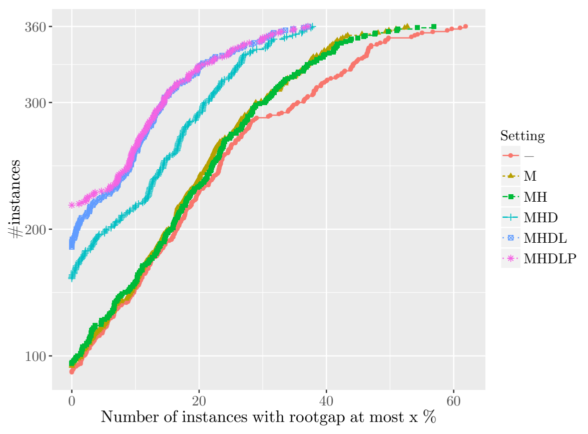
`MHDL` : as before, but instead of adding the basic interdiction cut associated with a heuristic follower solution $\hat{y}$, we perform the Lifting procedure described in Subsection 4.3 to $\hat{y}$, and only generate the associated lifted interdiction cut;

`MHDLP` : as before, but a Preprocessing step is applied that invokes the heuristic of Subsection 4.4 and possibly adds the associated invalid cuts (60) and (61) to the model formulation.

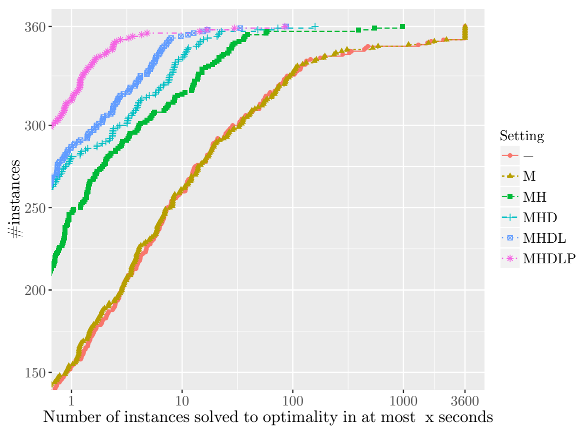
In all settings, only maximal follower solutions are considered for separation. Furthermore, both fractional and integer solutions are separated. Observe that, by construction, each execution of the separation algorithm returns (at most) a single violated (lifted) interdiction cut (possibly plus a modified interdiction cut), hence we did not impose any limit on the number of generated cuts at each separation call.

Figure 2 plots the root node gap and the runtime to optimality for the KIP instances from literature, while Figure 3 gives the same information for MKIP instances. The root gap is calculated as $100 \cdot (BestObj - RootBound)/(10^{-10} + |BestObj|)$, where $BestObj$ is the best objective value found by all settings, and $RootBound$ is the root-node lower bound produced by the setting. Observe that setting `MHDLP` may prove optimality of the heuristic solution UB by proving infeasibility of the problem after preprocessing; in case such infeasibility is already proven at the root node, we report a gap of zero.

Figure 2: Root gap and runtime to optimality for the KIP instances from literature and for different settings.

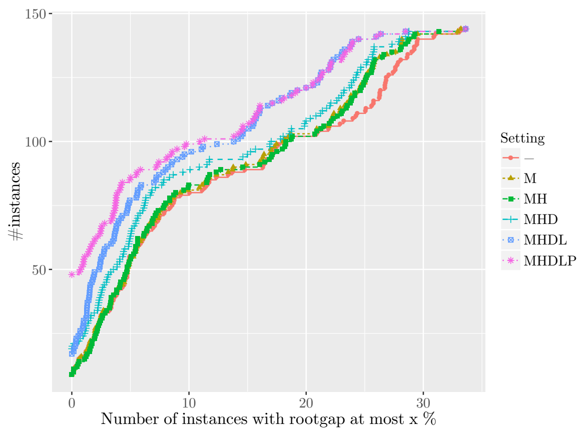


(a) Root gap.

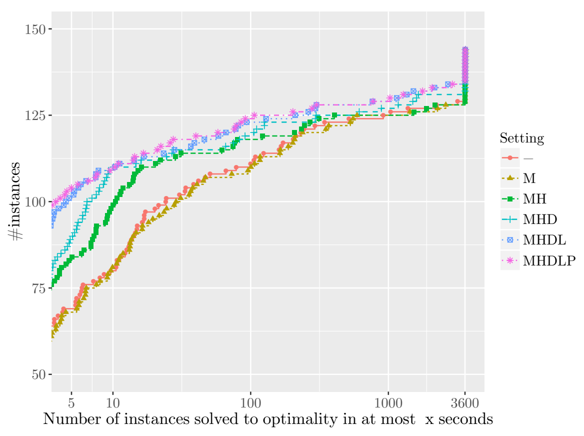


(b) Runtime to optimality.

Figure 3: Root gap and runtime to optimality for the MKIP instances and for different settings.



(a) Root gap.



(b) Runtime to optimality.

Both Figures 3a and 4a show that using the modified interdiction cuts slightly improves the root gap, while heuristic separation provides a root gap that is similar to that obtained using an exact procedure for separation. (Actually, for some MKIP instances, the gap with heuristic separation is even slightly better—this can be explained by the fact the CPLEX additionally generates internal MILP cuts that may affect the final bound in an unpredictable way.)

For the KIP instances from literature, a clear effect on the bounds can be observed when using the dominance inequalities (52) (setting `MHD`), resulting in about 170 instances solved to optimality at the root node, to be compared with about 100 for settings `-` and `H`. Additionally using lifting (setting `MHDL`) and the preprocessing heuristic (setting `MHDLP`) improves the gap even further so that about 220 instances can be solved to optimality at the root node. A similar trend, however less pronounced, can be observed for the MKIP instances. For both classes of instances, the gap at the root node for more than 2/3 of the instances is below 10% when using settings `MHDL*`.

Turning our attention to the runtime to optimality, i.e., to Figures 3b and 4b, we see that the noticeable difference in root gap between the settings does not directly translate into a similar difference in runtime for the KIP instances. In particular, while setting `M` slightly improves the root gap, the runtime to optimality is nearly identical to the basic setting `-`. On the other hand, additionally using the heuristic separation (`H`) gives a big improvement in runtime. The explanation of this behavior is that a much high node-throughput can be achieved in branch-and-bound when using heuristic separation, while slightly improved bounds (`M`) may not be crucial to quickly solve an instance to optimality. For MKIP, this effect is less pronounced, which is due to the fact that the greedy heuristic used in the KIP case is more efficient than the MILP-based heuristic for the MKIP-case.

Dominance inequalities and lifted interdiction cuts are both very important ingredients for harder instances. For example, for KIP instances, when using `MH` the most difficult instance takes about 1000 seconds, while setting `MHD` drastically reduces runtime to about 160 seconds, and `MHDL*` to about 80 seconds. Finally, using the primal heuristic in a preprocessing step as described in Section 4.4, further improves performance, especially for easy instances. This improvement may be crucial, if such problems have to be solved in a real-time setting.

We conclude by observing that the setting where all ingredients of our framework are used, namely `MHDLP`, gives the best overall performance. In particular, it solves all KIP instances from literature in at most 84 seconds (the most challenging problem being instance 55-3 of set `CCLW`, see next section), and only 4 out of the 360 instances take more than 10 seconds.

In view of the above, `MHDLP` is chosen as our default setting, and will be simply denoted by `B&C` in what follows.

5.3 Results for Instance Set `CCLW`

Table 1 gives a comparison of the results achieved by `B&C` (i.e., by our approach using its most-advanced setting `MHDLP`) against the integer cutting plane approach using interdiction cuts (column `CP`) and the specialized `CCLW` algorithm, both presented in Caprara et al. (2016). The results for `CP` and `CCLW` in Caprara et al. (2016) have been obtained on a four-core Intel Xeon @2.6 GHz. Column z^* gives the optimal solution value, while the remaining columns provide the runtime to optimality (in seconds) for the respective approaches. Entries TL in this column indicate runs for which the time limit of 3600 seconds has been reached. It may be observed that many instances of this dataset are very easy for both `CCLW` and `B&C` and are solved in around one second of computing time, while they are much harder for `CP`.

Recalling that the instances of set `CCLW` are constructed in such a way that a larger instance number means larger budget (for leader and follower, since the two budgets are set in a correlated way), one can observe that there is a peak of difficulty for `CCLW` for instances numbered three and four for all sizes. For `B&C`, this can only be observed for the largest set with 55 items.

Turning our attention to the hardest instances of the set, we see that `B&C` outperforms `CCLW` by up to 3 orders of magnitudes. Notably, `B&C` finds the optimal solution for the two unsolved instances 55-3 and 55-4 in just 84 and 16 seconds, respectively. Moreover, `B&C` solves instance 50-2 in just 2 seconds, while `CCLW` takes as long as $1,520$ seconds.

Table 1: Runtime to optimality, in seconds, for our approach (`B&C`) vs. the cutting plane (`CP`) and `CCLW` approaches from Caprara et al. (2016)

size	instance	z^*	`CP`	`CCLW`	`B&C`
35	1	279	0.34	0.79	0.12
	2	469	1.59	2.57	0.21
	3	448	55.61	40.39	0.66
	4	370	495.50	1.48	0.87
	5	467	TL	0.72	0.93
	6	268	71.43	0.06	0.11
	7	207	144.46	0.06	0.07
	8	41	0.50	0.04	0.07
	9	80	0.97	0.03	0.07
	10	31	0.12	0.03	0.08
40	1	314	0.66	1.06	0.16
	2	472	6.67	7.50	0.36
	3	637	324.61	162.80	1.02
	4	388	1900.03	0.34	0.82
	5	461	TL	0.22	0.58
	6	399	2111.85	0.09	0.13
	7	150	83.59	0.05	0.08
	8	71	1.73	0.04	0.09
	9	179	137.16	0.08	0.09
	10	0	0.03	0.03	0.04

size	instance	z^*	`CP`	`CCLW`	`B&C`
45	1	427	1.81	2.37	0.23
	2	633	13.03	11.64	0.37
	3	548	TL	344.01	1.81
	4	611	TL	38.90	3.30
	5	629	TL	3.42	2.78
	6	398	3300.76	0.07	0.17
	7	225	60.43	0.04	0.09
	8	157	60.88	0.05	0.10
	9	53	0.83	0.05	0.10
	10	110	0.40	0.05	0.11
50	1	502	2.86	4.55	0.21
	2	788	1529.16	1520.56	2.38
	3	631	TL	105.59	2.40
	4	612	TL	3.64	1.27
	5	764	TL	0.60	4.82
	6	303	1046.85	0.05	0.14
	7	310	2037.01	0.09	0.11
	8	63	2.79	0.05	0.12
	9	234	564.97	0.10	0.12
	10	15	0.09	0.04	0.13

size	instance	z^*	`CP`	`CCLW`	`B&C`
55	1	480	TL	18.57	0.46
	2	702	TL	443.53	1.50
	3	778	TL	TL	84.83
	4	889	TL	TL	16.75
	5	726	TL	0.24	1.36
	6	462	TL	0.09	0.16
	7	370	TL	0.08	0.12
	8	387	TL	0.10	0.13
	9	104	TL	0.06	0.13
	10	178	TL	0.06	0.14

5.4 Results for Instance Set TRS

Table 2 gives the results for instance set TRS. We compare the results of our B&C with the results obtained by the best-performing approach presented in Tang et al. (2015) (columns TRS), where this dataset has been proposed. We also benchmark our results against the best-performing setting of a state-of-the-art general purpose bilevel mixed-integer programming solver, namely the exact approach presented in (Fischetti et al. 2016b); see column MIX++. The results of Tang et al. (2015) have been obtained on "a PC with 3.30 GHz using CPLEX 12.5", while the results of Fischetti et al. (2016b) have been obtained with four-thread runs on the same machine we used for the runs in this paper. Results are given as averages over the ten instances per each $(|N|, k)$ pair. For each approach, column $t[s]$ reports runtime (in seconds); for TRS, we also provide the number of instances that were not solved to optimality within the time limit of one hour (column "N^*").

We observe that, for all $(|N|, k)$ pairs, our approach needs an average runtime of at most 0.3 seconds for computing a provably optimal solution. These computing times are smaller than those for MIX++ by up to 3 orders of magnitude. Furthermore, most of the instances with 22 or more items were unsolved by the approach of Tang et al. (2015) within one hour of computing time, while all of them are just trivial for our algorithm. Finally, note that all approaches except B&C are very sensitive to the value of k (i.e., to the number of items that can be interdicted): the most challenging instances for MIX++ are those with small k values, whereas medium values of k produce the hardest instances for the approach by Tang et al. (2015). No dependency with respect to k can instead be observed in B&C.

Table 2: Results for instance set TRS compared to results obtained by the best algorithm presented in Tang et al. (2015) (TRS) and by the state-of-the-art general purpose bilevel solver presented in (Fischetti et al. 2016b) (MIX++). Every row reports average results over ten instances. N^* gives the number of instances not solved to proven optimality by TRS.

		TRS		MIX++	B&C
$\|N\|$	k	t[s]	N^*	t[s]	t[s]
20	5	721.4	0	5.4	0.1
20	10	2992.6	3	1.7	0.1
20	15	129.5	0	0.2	0.1
22	6	1281.2	6	10.3	0.1
22	11	3601.8	10	2.3	0.1
22	17	248.2	0	0.2	0.1
25	7	3601.4	10	33.6	0.2
25	13	3602.3	10	8.0	0.2
25	19	1174.6	0	0.4	0.1
28	7	3601.0	10	97.9	0.3
28	14	3602.5	10	22.6	0.3
28	21	3496.9	8	0.5	0.1
30	8	3601.0	10	303.0	0.3
30	15	3602.3	10	31.8	0.3
30	23	3604.5	10	0.6	0.1

5.5 Results for Instance Set D

Table 3 gives results for instance set D. These instances have been introduced in (DeNegre 2011), where computational results have been only presented for the smallest problems with at most 30 items. As a much better general-purpose bilevel solver has been recently proposed by Fischetti et al. (2016b), in Table 3 we compare only the best setting of Fischetti et al. (2016b) (namely, MIX++) with our own B&C solver. Table 3 reports the value of the best solution found (column "BestSol") and, for each approach, the best lower bound (LB), the associated optimality gap (%gap) and the runtime in seconds (t[s]). For B&C, in case the heuristic solution obtained during preprocessing is optimal, we report CUTOFF in column LB. We report only the results for the larger instances with 30 to 50 items, as the smaller instances with up to 20 items were solved to optimality by both approaches in less than 10 seconds (in most cases, in less than one second).

The table shows that our `B&C` gives a speedup of 2-3 orders of magnitudes compared to `MIX++` for most of the instances (note however that the latter solver, though better than any previous method on these instances, is not specialized for interdiction). The speedup becomes more pronounced as the number of items grows. Furthermore, none of the instances with 50 items could be solved by `MIX++` within one hour, whereas `B&C` solves all instances except K5040W08 and K5050W08 within 4 seconds, while for K5040W08, resp., K5050W08 it takes 14, resp., 29 seconds. Interestingly, all but four instances are solved right after preprocessing, by proving infeasibility after the addition of the cutoff constraint.

5.6 Results for the MKIP instances based on the SAC-94 Library

Tables 4 to 6 compare the results obtained by `B&C` to the results obtained by the best setting of the general purpose bilevel solver presented in (Fischetti et al. 2016b) (`MIX++`). Both solvers have been run on the same machine with a time limit of one hour, though `MIX++` used four (instead of one) threads. As in Table 3, the tables report the value of the best solution found (BestSol), the lower bound (LB), the optimality gap (%gap), and the runtime in seconds (t[s]). Additionally, the number of items ($|N|$), leader constraints (#LC) and follower constraints (#FC) is given.

Table 4 reports results for instances of type `-100`, i.e., with single-knapsack follower. We see that depending on the underlying instance from which they have been created, they pose different difficulties to `MIX++`. For example, instances **`weing*`** are solved in less than three seconds (except **`weing8`**). Instances **`weish*`** are particularly hard for `MIX++`, more than half remaining unsolved within the time limit. Looking at **`hp*`** and **`pb*`** also reveals that the performance of `MIX++` is highly influenced by the number of variables and constraints. On the other hand, our `B&C` approach manages to solve all instances to optimality in at most four seconds, thus greatly outperforming `MIX++` for every instance. For about half of the instances, the heuristic solution obtained during preprocessing is the optimal one.

Table 5 addresses instances of type `-50`. For `MIX++`, they do not seem much more difficult than instances of type `-100`: half of the instances based on **`weish*`** cannot be solved within the time limit and, for the remaining ones, runtimes are similar to those of the associated instances of type `-100`. Thus, for `MIX++` the underlying instance seems to have a bigger impact on runtime than the number of follower constraints. For our `B&C` approach, instead, these instances are more difficult than the ones of type `-100`. This is not too surprising, as these instances have a multidimensional knapsack as follower problem, thus the preprocessing procedure is less effective. Moreover, the solution of the follower problem now consists of heuristically solving a MILP instead of a single knapsack problem. In any case, our approach manages to solve all but three of the instances to optimality—in more than 50% of the cases within four seconds. Again, for about half of the instances, the solution found in the first phase of the heuristic is the optimal one.

The three unsolved instances (within the time limit of 3600 seconds) are **`weish22, weish27`** and **`weish29`**. For these three instances the gap is at most 1.15%, compared to a gap of up to 73% for `MIX++`. We reran these three instance with a larger time limit, and all of them could be solved to optimality within 3900 seconds.

There seems to be no clear influence of the number of items and constraints on the performance of our approach, e.g., **`pb5`** with 20 items and 5 leader and follower constraints takes 12 times as long as **`pb6`** that has 40 items and 15 leader and follower constraints. Solver `MIX++` turns out to be faster than `B&C` only for instance **`pb5-50`** (49 vs 301 seconds).

Finally, Table 6 reports results for type `-0`. For `MIX++`, the results are very similar to the previous ones, and 16 out of the 30 instances **`weish*`** can be solved within the time limit of 3600 seconds. Our approach `B&C` manages to outperform `MIX++` for every instance, though it is not able to solve to optimality seven instances. However, for these unsolved instances (**`weish22`**, **`weish23`**, **`weish25`**, **`weish26`**, **`weish27`**, **`weish28`**, and **`weish29`**) the gap is at most 9.12%, compared with gaps of 25% to 75% for `MIX++`. In general, instances of class `-0` seem more difficult than instances of class `-50` (and of course, also class `-100`), thus the number/ratio of leader/follower constraints seems to influence the difficulty of the problem, and this effect seems not just to be restricted to the case, when the follower has just a single-knapsack constraint.

Moreover, the number of follower constraints also seem to influence the effectiveness of the heuristic, as for type `-0`, only for 15 out of 45 instances, the solution of the heuristic was the optimal one. Again, we reran the unsolved `-0` instances with a larger time limit, and all of them could be solved to proven optimality within 19000 seconds, except `weish27-0` that required about 30000 seconds.

6 Conclusions

In this article we have considered interdiction games in which the follower subproblem satisfies a certain monotonicity property. We have shown that this property is fulfilled by important classes of interdiction games, including the (single and multiple) knapsack problem, the facility location problem, and the prize-collecting traveling salesman (or Steiner tree) problem—just to mention a few.

For this large and important family of problems, we have proposed a new class of *interdiction cuts* that generalize those previously used in the literature. Building on these cuts, we have developed a Benders-like framework with some important enhancing ingredients. We have discussed additional families of modified/lifted interdiction cuts, as well as new dominance-based valid inequalities. For all classes of cuts, we have proposed exact and/or heuristic separation procedures, and we have used them to develop an effective solver. Finally, we have introduced a preprocessing procedure based on a new heuristic single-level compact MILP formulation.

We have computationally demonstrated that our new solver significantly outperforms very recent methods from the literature. In particular, we have tested our approach on 360 knapsack interdiction instances from the recent literature, and have proved the optimality for all of them—including for the 27 previously unsolved ones. Our algorithm needs at most 84 seconds for solving any of these instances (for only four of these 360 instances, it takes more than 10 seconds), outperforming previous approaches from literature by up to 4 orders of magnitude. Computational tests on new random instances based on 0/1 multidimensional knapsack problems have also been performed in order to assess the dependency of our approach on the number of leader and follower constraints. Also for this kind of instances, our approach outperforms by orders of magnitude the state-of-the art general bilevel solver recently proposed in Fischetti et al. (2016b).

Future work should address the extension of our approach to the non-monotone case, as well as the customization of our solution method to special classes of monotone IGs, including the facility location and prize collecting applications outlined in Section 2.

Acknowledgements

This research was funded by the Vienna Science and Technology Fund (WWTF) through project ICT15-014. The work of M. Fischetti and M. Monaci was also supported by MiUR, Italy, through the PRIN-2015 project. The work of I. Ljubić and M. Sinnl was also supported by the Austrian Research Fund (FWF, Project P 26755-N19). Thanks are due to two anonymous referees for their helpful comments.

References

Balas E (1989) The prize collecting traveling salesman problem. *Networks* 19(6):621–636.

Balas E (2007) *The Prize Collecting Traveling Salesman Problem and its Applications*, 663–695 (Boston, MA: Springer US).

Bazgan C, Toubaline S, Tuza Z (2011) The most vital nodes with respect to independent set and vertex cover. *Discrete Applied Mathematics* 159(17):1933 – 1946.

Bienstock D, Goemans MX, Simchi-Levi D, Williamson D (1993) A note on the prize collecting traveling salesman problem. *Mathematical Programming* 59(1-3):413–420.

Brown G, Carlyle M, Harney R, Skroch E, Wood K (2009) Interdicting a nuclear-weapons project. *Operations Research* 57(4):866–877.

Brown G, Carlyle M, Salmeron J, Wood K (2005) Analyzing the vulnerability of critical infrastructure to attack and planning defenses. *Tutorials in Operations Research: Emerging Theory, Methods, and Applications* 102–123.

Brown G, Carlyle M, Salmeron J, Wood K (2006) Defending critical infrastructure. *Interfaces* 36(6):530–544.

Caprara A, Carvalho M, Lodi A, Woeginger GJ (2013) A complexity and approximability study of the bilevel knapsack problem. *Proceedings of the 16th International Conference on Integer Programming and Combinatorial Optimization*, 98–109, IPCO'13 (Springer).

Caprara A, Carvalho M, Lodi A, Woeginger GJ (2016) Bilevel knapsack with interdiction constraints. *INFORMS Journal on Computing* 28(2):319–333.

Cormican K, Morton D, Wood K (1998) Stochastic network interdiction. *Operations Research* 46(2):184 – 197.

DeNegre S (2011) *Interdiction and Discrete Bilevel Linear Programming*. Ph.D. thesis, Lehigh University.

Dinitz M, Gupta A (2013) Packing interdiction and partial covering problems. *Proceedings of the 16th International Conference on Integer Programming and Combinatorial Optimization*, 157–168, IPCO'13 (Springer-Verlag).

Fischetti M, Leitner M, Ljubić I, Luipersbeck M, Monaci M, Resch M, Salvagnin D, Sinnl M (2016a) Thinning out steiner trees: a node-based model for uniform edge costs. To appear on *Mathematical Programming Computation*. DOI: 10.1007/s12532-016-0111-0.

Fischetti M, Ljubić I, Monaci M, Sinnl M (2016b) An improved branch-and-cut algorithm for mixed-integer bilevel linear programs Under revision.

Fischetti M, Ljubić I, Monaci M, Sinnl M (2016c) Intersection cuts for bilevel optimization. Louveaux Q, Skutella M, eds., *Integer Programming and Combinatorial Optimization: 18th International Conference, IPCO 2016, Liège, Belgium, June 1-3, 2016, Proceedings, Springer International Publishing*, 77–88 (extended version submitted for publication), URL `http://www.dei.unipd.it/~fisch/papers/intersection_cuts_for_bilevel_optimization.pdf`.

Freville A, Plateau G (1990) Hard 0-1 multiknapsack test problems for size reduction methods. *Investigation Operativa* 1:251–270.

Halldórsson MM (2000) Approximations of weighted independent set and hereditary subset problems. *Journal of Graph Algorithms and Applications* 4(1):1–16.

Israeli E, Wood RK (2002) Shortest-path network interdiction. *Networks* 40(2):97–111.

Khuri S, Baeck T, Heitkoetter J (1994) SAC94 Suite: Collection of Multiple Knapsack Problems. `http://www.cs.cmu.edu/Groups/AI/areas/genetic/ga/test/sac/0.html`.

Kleniati PM, Adjiman CS (2015) A generalization of the branch-and-sandwich algorithm: From continuous to mixed-integer nonlinear bilevel problems. *Computers & Chemical Engineering* 72:373 – 386.

Ljubić I, Weiskircher R, Pferschy U, Klau GW, Mutzel P, Fischetti M (2006) An algorithmic framework for the exact solution of the prize-collecting steiner tree problem. *Mathematical Programming* 105(2-3):427–449.

Martello S, Pisinger D, Toth P (1999) Dynamic programming and strong bounds for the 0-1 knapsack problem. *Management Science* 45(3):414–424.

Martello S, Toth P (1990) *Knapsack problems: algorithms and computer implementations* (John Wiley & Sons, Inc.).

Moore J, Bard J (1990) The mixed integer linear bilevel programming problem. *Operations Research* 38(5):911–921.

Morton DP, Pan F, Saeger KJ (2007) Models for nuclear smuggling interdiction. *IIE Transactions* 39(1):3–14.

Petersen CC (1967) Computational experience with variants of the Balas algorithm applied to the selection of R&D projects. *Management Science* 13(9):736–750.

Prodon A, DeNegre S, Liebling TM (2010) Locating leak detecting sensors in a water distribution network by solving prize-collecting steiner arborescence problems. *Mathematical Programming* 124(1-2):119–141.

Ralphs T (2015) Bilevel integer optimization: Theory and algorithms, talk at *22nd International Symposium on Mathhematical Programming*.

Senju S, Toyoda Y (1968) An approach to linear programming with 0-1 variables. *Management Science* 15(4):B196–B207.

Shih W (1979) A branch and bound method for the multiconstraint zero-one knapsack problem. *Journal of the Operational Research Society* 30(4):369–378.

Smith JC, Lim C (2008) *Algorithms for Network Interdiction and Fortification Games*, 609–644 (New York, NY: Springer New York).

Tang Y, Richard JPP, Smith JC (2015) A class of algorithms for mixed-integer bilevel min–max optimization. *Journal of Global Optimization* 1–38.

Vansteenwegen P, Souffriau W, Van Oudheusden D (2011) The orienteering problem: A survey. *European Journal of Operational Research* 209(1):1–10.

Von Stackelberg H (1952) *The theory of the market economy* (Oxford University Press).

Washburn A, Wood K (1995) Two-person zero-sum games for network interdiction. *Operations Research* 43(2):243–251.

Weingartner HM, Ness DN (1967) Methods for the solution of the multidimensional 0/1 knapsack problem. *Operations Research* 15(1):83–103.

Wood RK (2010) *Bilevel Network Interdiction Models: Formulations and Solutions* (John Wiley & Sons, Inc.).

Xu P (2012) *Three essays on bilevel optimization algorithms and applications*. Ph.D. thesis, Iowa State University.

Xu P, Wang L (2014) An exact algorithm for the bilevel mixed integer linear programming problem under three simplifying assumptions. *Computers & Operations Research* 41:309–318.

Zeng B, An Y (2014) Solving bilevel mixed integer program by reformulations and decomposition. available at `http://www.optimization-online.org/DB_FILE/2014/07/4455.pdf`.

Zenklusen R (2010) Matching interdiction. *Discrete Applied Mathematics* 158(15):1676 – 1690.

Table 3: Results for instance set D compared to results obtained by the state-of-the-art general purpose bilevel solver presented in (Fischetti et al. 2016b).

	MIX++ from Fischetti et al. (2016b)				B&C			
instance	BestSol	LB	%gap	t[s]	BestSol	LB	%gap	t[s]
K5030W01	2956	2956.0000	0.00	39.88	2956	CUTOFF	0.00	0.72
K5030W02	3529	3529.0000	0.00	86.77	3529	CUTOFF	0.00	0.59
K5030W03	2706	2706.0000	0.00	43.01	2706	CUTOFF	0.00	0.61
K5030W04	3201	3201.0000	0.00	73.32	3201	CUTOFF	0.00	0.65
K5030W05	4861	4861.0000	0.00	569.09	4861	CUTOFF	0.00	2.37
K5030W06	1997	1997.0000	0.00	12.12	1997	CUTOFF	0.00	0.47
K5030W07	2270	2270.0000	0.00	18.99	2270	CUTOFF	0.00	0.45
K5030W08	4902	4902.0000	0.00	1077.40	4902	4902.0000	0.00	3.58
K5030W09	2201	2201.0000	0.00	14.06	2201	CUTOFF	0.00	0.50
K5030W10	2668	2668.0000	0.00	19.00	2668	CUTOFF	0.00	0.75
K5030W11	2013	2013.0000	0.00	28.67	2013	CUTOFF	0.00	0.50
K5030W12	2534	2534.0000	0.00	11.42	2534	CUTOFF	0.00	0.33
K5030W13	3152	3152.0000	0.00	21.57	3152	CUTOFF	0.00	0.53
K5030W14	2184	2184.0000	0.00	23.05	2184	CUTOFF	0.00	0.43
K5030W15	2841	2841.0000	0.00	53.60	2841	CUTOFF	0.00	0.58
K5030W16	2102	2102.0000	0.00	12.57	2102	CUTOFF	0.00	0.47
K5030W17	3553	3553.0000	0.00	98.74	3553	CUTOFF	0.00	0.50
K5030W18	2602	2602.0000	0.00	19.66	2602	CUTOFF	0.00	0.50
K5030W19	5015	5015.0000	0.00	710.57	5015	CUTOFF	0.00	2.44
K5030W20	2496	2496.0000	0.00	11.95	2496	2496.0000	0.00	0.79
K5040W01	4254	3204.0000	24.68	TL	4254	CUTOFF	0.00	2.00
K5040W02	4423	4423.0000	0.00	2533.36	4423	CUTOFF	0.00	1.02
K5040W03	3440	3440.0000	0.00	1578.91	3440	CUTOFF	0.00	0.68
K5040W04	3574	3574.0000	0.00	1158.20	3574	CUTOFF	0.00	1.04
K5040W05	4646	3363.6302	27.60	TL	4529	CUTOFF	0.00	1.32
K5040W06	2606	2606.0000	0.00	233.58	2606	CUTOFF	0.00	0.99
K5040W07	3244	3244.0000	0.00	600.63	3244	CUTOFF	0.00	1.11
K5040W08	6345	2870.0000	54.77	TL	6174	6173.5586	0.00	14.44
K5040W09	3154	3154.0000	0.00	410.00	3154	CUTOFF	0.00	0.57
K5040W10	4382	4382.0000	0.00	3099.20	4382	CUTOFF	0.00	1.41
K5040W11	3389	3389.0000	0.00	1120.76	3389	CUTOFF	0.00	0.81
K5040W12	3817	3817.0000	0.00	593.61	3817	CUTOFF	0.00	0.56
K5040W13	4174	4174.0000	0.00	1126.49	4174	CUTOFF	0.00	0.85
K5040W14	3374	3374.0000	0.00	1090.91	3374	CUTOFF	0.00	0.75
K5040W15	3925	3164.8373	19.37	TL	3925	CUTOFF	0.00	0.56
K5040W16	2605	2605.0000	0.00	194.39	2605	CUTOFF	0.00	1.20
K5040W17	3996	3996.0000	0.00	2645.77	3996	CUTOFF	0.00	1.05
K5040W18	3342	3342.0000	0.00	918.78	3342	CUTOFF	0.00	0.52
K5040W19	5299	3167.0000	40.23	TL	5233	CUTOFF	0.00	1.68
K5040W20	2875	2875.0000	0.00	267.87	2875	CUTOFF	0.00	0.89
K5050W01	4244	2610.2294	38.50	TL	4189	CUTOFF	0.00	1.20
K5050W02	5280	2559.0000	51.53	TL	5106	CUTOFF	0.00	1.20
K5050W03	5483	2530.0283	53.86	TL	4769	CUTOFF	0.00	1.39
K5050W04	3999	2401.0000	39.96	TL	3723	CUTOFF	0.00	1.20
K5050W05	5109	2408.0000	52.87	TL	4998	CUTOFF	0.00	4.31
K5050W06	3558	2691.9300	24.34	TL	3558	CUTOFF	0.00	1.55
K5050W07	4521	2355.0000	47.91	TL	4390	CUTOFF	0.00	2.34
K5050W08	8215	2706.5175	67.05	TL	7862	7862.0000	0.00	29.65
K5050W09	4775	2521.0000	47.20	TL	4620	CUTOFF	0.00	1.21
K5050W10	5575	2682.1036	51.89	TL	5047	CUTOFF	0.00	2.13
K5050W11	3855	2287.0000	40.67	TL	3778	CUTOFF	0.00	1.63
K5050W12	4885	2738.0731	43.95	TL	4562	CUTOFF	0.00	1.61
K5050W13	4926	2816.0000	42.83	TL	4778	CUTOFF	0.00	1.27
K5050W14	5055	2249.0000	55.51	TL	4544	CUTOFF	0.00	1.19
K5050W15	4757	2240.7014	52.90	TL	4610	CUTOFF	0.00	1.17
K5050W16	4039	2222.0000	44.99	TL	3979	CUTOFF	0.00	1.52
K5050W17	5666	2672.0930	52.84	TL	5218	CUTOFF	0.00	1.24
K5050W18	4591	2858.0000	37.75	TL	4591	CUTOFF	0.00	1.13
K5050W19	6022	2717.5260	54.87	TL	5858	CUTOFF	0.00	2.06
K5050W20	4303	2247.0000	47.78	TL	4303	CUTOFF	0.00	2.57

Table 4: Results for instance set `SAC` compared to results obtained by the state-of-the-art general purpose bilevel solver presented in (Fischetti et al. 2016b).

				`MIX++ Fischetti et al. (2016b)`				`B&C`			
instance	$\lvert N \rvert$	#LC	#FC	BestSol	LB	%gap	t[s]	BestSol	LB	%gap	t[s]
hp1-100	28	3	1	1536	1536.0000	0.00	13.06	1536	CUTOFF	0.00	0.05
hp2-100	35	3	1	3015	3015.0000	0.00	304.14	3015	3015.0000	0.00	2.55
pb1-100	27	3	1	1536	1536.0000	0.00	11.08	1536	CUTOFF	0.00	0.04
pb2-100	34	3	1	1902	1902.0000	0.00	251.04	1902	CUTOFF	0.00	0.10
pb4-100	29	1	1	52329	52329.0000	0.00	5.42	52329	CUTOFF	0.00	0.04
pb5-100	20	9	1	1799	1799.0000	0.00	37.29	1799	1799.0000	0.00	1.66
pb6-100	40	29	1	1389	1389.0000	0.00	250.25	1389	1389.0000	0.00	0.85
pb7-100	37	29	1	670	565.0000	15.67	TL	656	656.0000	0.00	0.59
pet2-100	10	9	1	38833	38833.0000	0.00	0.97	38833	CUTOFF	0.00	0.03
pet3-100	15	9	1	1080	1080.0000	0.00	1.90	1080	CUTOFF	0.00	0.04
pet4-100	20	9	1	2505	2505.0000	0.00	4.07	2505	CUTOFF	0.00	0.09
pet5-100	28	9	1	3025	3025.0000	0.00	18.06	3025	CUTOFF	0.00	0.06
pet6-100	39	4	1	3936	3936.0000	0.00	419.09	3936	CUTOFF	0.00	0.07
pet7-100	50	4	1	5935	5166.0000	12.96	TL	5723	CUTOFF	0.00	0.11
sento1-100	60	29	1	1686	1225.0000	27.34	TL	1610	1610.0000	0.00	3.84
sento2-100	60	29	1	752	457.0000	39.23	TL	738	CUTOFF	0.00	0.46
weing1-100	28	1	1	6205	6205.0000	0.00	0.70	6205	CUTOFF	0.00	0.05
weing2-100	28	1	1	16705	16705.0000	0.00	0.89	16705	CUTOFF	0.00	0.05
weing3-100	28	1	1	37936	37936.0000	0.00	1.28	37936	37936.0000	0.00	0.13
weing4-100	28	1	1	42958	42958.0000	0.00	0.65	42958	CUTOFF	0.00	0.06
weing5-100	28	1	1	6205	6205.0000	0.00	0.52	6205	CUTOFF	0.00	0.05
weing6-100	28	1	1	8103	8103.0000	0.00	0.60	8103	CUTOFF	0.00	0.05
weing7-100	105	1	1	15646	15646.0000	0.00	2.70	15646	CUTOFF	0.00	0.34
weing8-100	105	1	1	212854	212854.0000	0.00	151.36	212854	212854.0000	0.00	1.31
weish01-100	30	4	1	1121	1121.0000	0.00	14.61	1121	1121.0000	0.00	0.19
weish02-100	30	4	1	1293	1293.0000	0.00	16.76	1293	CUTOFF	0.00	0.08
weish03-100	30	4	1	1601	1601.0000	0.00	10.12	1601	1601.0000	0.00	0.20
weish04-100	30	4	1	1268	1268.0000	0.00	5.18	1268	CUTOFF	0.00	0.13
weish05-100	30	4	1	1315	1315.0000	0.00	5.08	1315	CUTOFF	0.00	0.11
weish06-100	40	4	1	1369	1369.0000	0.00	335.60	1369	CUTOFF	0.00	0.22
weish07-100	40	4	1	1407	1407.0000	0.00	574.35	1407	CUTOFF	0.00	0.12
weish08-100	40	4	1	1369	1369.0000	0.00	210.40	1369	CUTOFF	0.00	0.12
weish09-100	40	4	1	1645	1645.0000	0.00	88.94	1645	1645.0000	0.00	0.51
weish10-100	50	4	1	2146	2146.0000	0.00	809.66	2146	2146.0000	0.00	0.43
weish11-100	50	4	1	2827	2827.0000	0.00	331.23	2827	2827.0000	0.00	0.88
weish12-100	50	4	1	2146	2146.0000	0.00	621.03	2146	2146.0000	0.00	0.38
weish13-100	50	4	1	2369	2369.0000	0.00	628.74	2369	2369.0000	0.00	0.63
weish14-100	60	4	1	2648	1825.1101	31.08	TL	2625	2625.0000	0.00	1.67
weish15-100	60	4	1	2138	1759.0000	17.73	TL	2138	CUTOFF	0.00	0.20
weish16-100	60	4	1	2336	1435.1534	38.56	TL	2285	2285.0000	0.00	1.21
weish17-100	60	4	1	1010	808.5564	19.94	TL	991	CUTOFF	0.00	0.15
weish18-100	70	4	1	1986	1348.8854	32.08	TL	1945	CUTOFF	0.00	0.19
weish19-100	70	4	1	3874	1779.0000	54.08	TL	3741	3740.8875	0.00	1.80
weish20-100	70	4	1	2142	1310.0881	38.84	TL	2075	CUTOFF	0.00	0.24
weish21-100	70	4	1	2535	1453.7849	42.65	TL	2451	CUTOFF	0.00	0.24
weish22-100	80	4	1	3719	1524.1072	59.02	TL	3325	CUTOFF	0.00	0.78
weish23-100	80	4	1	4177	1602.8044	61.63	TL	3906	3906.0000	0.00	1.15
weish24-100	80	4	1	2190	1277.6197	41.66	TL	2111	CUTOFF	0.00	0.23
weish25-100	80	4	1	2445	1155.6013	52.74	TL	2392	CUTOFF	0.00	0.31
weish26-100	90	4	1	4266	1627.1253	61.86	TL	3799	CUTOFF	0.00	1.25
weish27-100	90	4	1	4077	1545.9016	62.08	TL	3565	CUTOFF	0.00	0.99
weish28-100	90	4	1	4441	1635.0000	63.18	TL	3896	CUTOFF	0.00	0.84
weish29-100	90	4	1	4514	1690.2451	62.56	TL	3997	3997.0000	0.00	1.55
weish30-100	90	4	1	2267	1504.7394	33.62	TL	2226	CUTOFF	0.00	0.30

Table 5: Results for instance set `SAC` compared to results obtained by the state-of-the-art general purpose bilevel solver presented in (Fischetti et al. 2016b). The optimal solution value (obtained with a larger time limit) for `weish22-50` is 1372, for `weish27-50` is 1290, and for `weish29-50` is 1205.

				`MIX++` from Fischetti et al. (2016b)				`B&C`			
instance	$\|N\|$	#LC	#FC	BestSol	LB	%gap	t[s]	BestSol	LB	%gap	t[s]
hp1-50	28	2	2	1536	1536.0000	0.00	21.21	1536	CUTOFF	0.00	0.08
hp2-50	35	2	2	2912	2912.0000	0.00	263.13	2912	CUTOFF	0.00	20.95
pb1-50	27	2	2	1536	1536.0000	0.00	12.90	1536	CUTOFF	0.00	0.86
pb2-50	34	2	2	1787	1787.0000	0.00	181.58	1787	CUTOFF	0.00	1.72
pb5-50	20	5	5	1625	1625.0000	0.00	48.58	1625	1624.9853	0.00	301.03
pb6-50	40	15	15	634	634.0000	0.00	119.05	634	634.0000	0.00	26.02
pb7-50	37	15	15	423	423.0000	0.00	1073.40	423	423.0000	0.00	106.16
pet2-50	10	5	5	38833	38833.0000	0.00	1.38	38833	CUTOFF	0.00	0.07
pet3-50	15	5	5	905	905.0000	0.00	1.23	905	CUTOFF	0.00	0.07
pet4-50	20	5	5	2445	2445.0000	0.00	3.67	2445	2445.0000	0.00	0.92
pet5-50	28	5	5	3025	3025.0000	0.00	45.83	3025	CUTOFF	0.00	0.38
pet6-50	39	3	2	3936	3936.0000	0.00	474.10	3936	CUTOFF	0.00	0.98
pet7-50	50	3	2	5873	5031.3240	14.33	TL	5723	CUTOFF	0.00	14.05
sento1-50	60	15	15	1102	1102.0000	0.00	2235.55	1102	1102.0000	0.00	76.06
sento2-50	60	15	15	522	338.0000	35.25	TL	503	503.0000	0.00	10.32
weish01-50	30	3	2	1097	1097.0000	0.00	23.81	1097	1097.0000	0.00	1.22
weish02-50	30	3	2	1293	1293.0000	0.00	28.99	1293	CUTOFF	0.00	0.20
weish03-50	30	3	2	619	619.0000	0.00	9.78	619	619.0000	0.00	0.30
weish04-50	30	3	2	1027	1027.0000	0.00	5.57	1027	1027.0000	0.00	0.21
weish05-50	30	3	2	1215	1215.0000	0.00	8.22	1215	1215.0000	0.00	0.20
weish06-50	40	3	2	1369	1369.0000	0.00	373.87	1369	CUTOFF	0.00	1.55
weish07-50	40	3	2	1407	1407.0000	0.00	804.42	1407	CUTOFF	0.00	0.52
weish08-50	40	3	2	1369	1369.0000	0.00	372.00	1369	CUTOFF	0.00	0.32
weish09-50	40	3	2	1568	1568.0000	0.00	108.72	1568	1568.0000	0.00	0.54
weish10-50	50	3	2	785	785.0000	0.00	232.30	785	785.0000	0.00	2.20
weish11-50	50	3	2	584	584.0000	0.00	58.67	584	584.0000	0.00	2.16
weish12-50	50	3	2	778	778.0000	0.00	242.39	778	778.0000	0.00	2.64
weish13-50	50	3	2	742	742.0000	0.00	140.11	742	742.0000	0.00	2.25
weish14-50	60	3	2	1041	811.1739	22.08	TL	1020	1020.0000	0.00	40.00
weish15-50	60	3	2	1931	1931.0000	0.00	3110.28	1931	1931.0000	0.00	4.99
weish16-50	60	3	2	2198	1474.0000	32.94	TL	2172	2172.0000	0.00	7.45
weish17-50	60	3	2	991	819.0525	17.35	TL	991	CUTOFF	0.00	0.06
weish18-50	70	3	2	2113	948.0000	55.13	TL	1945	CUTOFF	0.00	0.15
weish19-50	70	3	2	1194	599.8071	49.76	TL	1095	1095.0000	0.00	202.63
weish20-50	70	3	2	2274	1021.0000	55.10	TL	2075	CUTOFF	0.00	0.42
weish21-50	70	3	2	2601	1263.0000	51.44	TL	2451	CUTOFF	0.00	0.80
weish22-50	80	3	2	1522	504.0716	66.88	TL	1372	1358.8669	0.96	TL
weish23-50	80	3	2	1309	522.0000	60.12	TL	1236	1236.0000	0.00	1026.43
weish24-50	80	3	2	2360	889.4432	62.31	TL	2111	CUTOFF	0.00	0.23
weish25-50	80	3	2	2576	920.3113	64.27	TL	2392	CUTOFF	0.00	0.42
weish26-50	90	3	2	1384	402.0000	70.95	TL	1243	1242.8969	0.00	2913.69
weish27-50	90	3	2	1470	391.0000	73.40	TL	1290	1275.2127	1.15	TL
weish28-50	90	3	2	1513	444.5101	70.62	TL	1358	1357.9082	0.00	2079.98
weish29-50	90	3	2	1401	405.0000	71.09	TL	1205	1196.8229	0.68	TL
weish30-50	90	3	2	2356	1221.8923	48.14	TL	2226	CUTOFF	0.00	0.09

Table 6: Results for instance set `SAC` compared to results obtained by the state-of-the-art general purpose bilevel solver presented in (Fischetti et al. 2016b). The optimal solution value (obtained with a larger time limit) for `weish22-0` is 1372, for `weish23-0` is 1236, for `weish25-0` is 1079, for `weish26-0` is 1243, for `weish27-0` is 1290, for `weish28-0` is 1358, and for `weish29-0` is 1205.

				`MIX++` from Fischetti et al. (2016b)				`B&C`			
instance	$\|N\|$	#LC	#FC	BestSol	LB	%gap	t[s]	BestSol	LB	%gap	t[s]
hp1-0	28	1	3	1467	1467.0000	0.00	12.86	1467	CUTOFF	0.00	0.66
hp2-0	35	1	3	2278	2278.0000	0.00	377.55	2278	2278.0000	0.00	4.47
pb1-0	27	1	3	1467	1467.0000	0.00	11.39	1467	CUTOFF	0.00	0.70
pb2-0	34	1	3	1784	1784.0000	0.00	145.71	1784	CUTOFF	0.00	4.14
pb5-0	20	1	9	1417	1417.0000	0.00	26.44	1417	CUTOFF	0.00	16.63
pb6-0	40	1	29	292	292.0000	0.00	19.42	292	292.0000	0.00	14.37
pb7-0	37	1	29	185	185.0000	0.00	31.28	185	CUTOFF	0.00	2.35
pet2-0	10	1	9	25295	25295.0000	0.00	0.39	25295	CUTOFF	0.00	0.06
pet3-0	15	1	9	905	905.0000	0.00	0.66	905	CUTOFF	0.00	0.22
pet4-0	20	1	9	1935	1935.0000	0.00	2.71	1935	1935.0000	0.00	1.04
pet5-0	28	1	9	2195	2195.0000	0.00	9.17	2195	CUTOFF	0.00	0.13
pet6-0	39	1	4	3683	3683.0000	0.00	330.08	3683	CUTOFF	0.00	1.43
pet7-0	50	1	4	5636	4986.0000	11.53	TL	5459	CUTOFF	0.00	9.51
sento1-0	60	1	29	552	552.0000	0.00	856.95	552	552.0000	0.00	78.60
sento2-0	60	1	29	226	226.0000	0.00	226.93	226	CUTOFF	0.00	1.07
weish01-0	30	1	4	923	923.0000	0.00	13.91	923	923.0000	0.00	0.84
weish02-0	30	1	4	1108	1108.0000	0.00	18.34	1108	1108.0000	0.00	0.56
weish03-0	30	1	4	619	619.0000	0.00	4.90	619	619.0000	0.00	0.26
weish04-0	30	1	4	465	465.0000	0.00	5.49	465	465.0000	0.00	1.80
weish05-0	30	1	4	443	443.0000	0.00	4.71	443	443.0000	0.00	1.82
weish06-0	40	1	4	1283	1283.0000	0.00	340.99	1283	1283.0000	0.00	7.69
weish07-0	40	1	4	1185	1185.0000	0.00	184.75	1185	1185.0000	0.00	4.63
weish08-0	40	1	4	1283	1283.0000	0.00	387.82	1283	1283.0000	0.00	5.55
weish09-0	40	1	4	532	532.0000	0.00	83.48	532	532.0000	0.00	6.63
weish10-0	50	1	4	785	785.0000	0.00	280.40	785	785.0000	0.00	10.96
weish11-0	50	1	4	584	584.0000	0.00	92.28	584	584.0000	0.00	2.15
weish12-0	50	1	4	778	778.0000	0.00	314.01	778	778.0000	0.00	3.36
weish13-0	50	1	4	742	742.0000	0.00	218.14	742	742.0000	0.00	3.62
weish14-0	60	1	4	1046	786.0000	24.86	TL	1020	1020.0000	0.00	85.27
weish15-0	60	1	4	759	759.0000	0.00	2484.18	759	759.0000	0.00	57.82
weish16-0	60	1	4	876	644.0000	26.48	TL	828	828.0000	0.00	278.70
weish17-0	60	1	4	36	36.0000	0.00	0.76	36	CUTOFF	0.00	0.08
weish18-0	70	1	4	2139	749.0000	64.98	TL	1927	CUTOFF	0.00	22.28
weish19-0	70	1	4	1170	600.3016	48.69	TL	1095	1095.0000	0.00	1283.08
weish20-0	70	1	4	1086	526.0000	51.57	TL	964	964.0000	0.00	773.93
weish21-0	70	1	4	988	569.0000	42.41	TL	904	903.9480	0.00	1770.06
weish22-0	80	1	4	1465	485.9626	66.83	TL	1374	1292.7368	5.91	TL
weish23-0	80	1	4	1361	485.0000	64.36	TL	1248	1161.4429	6.94	TL
weish24-0	80	1	4	2401	571.7136	76.19	TL	2094	CUTOFF	0.00	27.41
weish25-0	80	1	4	1181	406.0000	65.62	TL	1090	1001.5398	8.12	TL
weish26-0	90	1	4	1484	365.2379	75.39	TL	1243	1145.6534	7.83	TL
weish27-0	90	1	4	1431	412.0000	71.21	TL	1296	1177.8500	9.12	TL
weish28-0	90	1	4	1482	434.0000	70.72	TL	1358	1280.7280	5.69	TL
weish29-0	90	1	4	1368	385.0000	71.86	TL	1206	1110.1846	7.94	TL
weish30-0	90	1	4	829	314.2100	62.10	TL	724	CUTOFF	0.00	91.42