Intersection Cuts for Bilevel Optimization

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

The exact solution of bilevel optimization problems is a very challenging task that received more and more attention in recent years, as witnessed by the flourishing recent literature on this topic. In this paper we present ideas and algorithms to solve to proven optimality generic Mixed-Integer Bilevel Linear Programs (MIBLP's) where all constraints are linear, and some/all variables are required to take integer values. In doing so, we look for a general-purpose approach applicable to any MIBLP (under mild conditions), rather than ad-hoc methods for specific cases. Our approach concentrates on minimal additions required to convert an effective branch-and-cut MILP exact code into a valid MIBLP solver, thus inheriting the wide arsenal of MILP tools (cuts, branching rules, heuristics) available in modern solvers.

1 Introduction

A general bilevel optimization problem is defined as

$$\min_{x\in\mathbb{R}^{n_1}, y\in\mathbb{R}^{n_2}} F(x,y) \tag{1}$$

$$G(x,y) \le 0 \tag{2}$$

$$y \in \arg\min_{y'\in\mathbb{R}^{n_2}} \{ f(x,y') : g(x,y') \le 0 \}, \tag{3}$$

where $F, f : \mathbb{R}^{n_1+n_2} \to \mathbb{R}$, $G : \mathbb{R}^{n_1+n_2} \to \mathbb{R}^{m_1}$, and $g : \mathbb{R}^{n_1+n_2} \to \mathbb{R}^{m_2}$. Let $n = n_1 + n_2$ denote the total number of decision variables.

We will refer to $F(x,y)$ and $G(x,y) \le 0$ as the *leader* objective function and constraints, respectively, and to (3) as the *follower* subproblem. In case the follower subproblem has multiple optimal solutions, we assume that one with minimum leader cost among those with $G(x,y) \le 0$ is chosen—i.e. we consider the *optimistic* version of bilevel optimization.

By defining the follower value function for a given $x \in \mathbb{R}^{n_1}$

$$\Phi(x) = \min_{y\in\mathbb{R}^{n_2}} \{ f(x,y) : g(x,y) \le 0 \}, \tag{4}$$

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

one can restate the bilevel optimization problem as follows:

$$\min F(x,y) \tag{5}$$
$$G(x,y) \leq 0 \tag{6}$$
$$g(x,y) \leq 0 \tag{7}$$
$$(x,y) \in \mathbb{R}^n \tag{8}$$
$$f(x,y) \leq \Phi(x). \tag{9}$$

Note that the above optimization problem would be hard (both theoretically and in practice) even if one would assume convexity of F, G, f and g (which would imply that of Φ), due to the intrinsic nonconvexity of (9).

Dropping condition (9) leads the so-called *High Point Relaxation* (HPR). As customary in the bilevel context, we assume that HPR is feasible and bounded, and that the minimization problem in (4) is bounded for each feasible solution of HPR—while its feasibility follows directly from the definition of HPR. As HPR contains all the follower constraints, any HPR solution (x, y) satisfies $f(x, y) \geq \Phi(x)$, hence (9) actually implies $f(x, y) = \Phi(x, y)$.

A point $(x, y) \in \mathbb{R}^n$ will be called *bilevel infeasible* if it violates (9). A point $(x, y) \in \mathbb{R}^n$ will be called *bilevel feasible* if it is satisfies all constraints (6)–(9).

2 Literature overview

In this paper we will mainly focus on Mixed-Integer Bilevel Linear Programs (MIBLP's) where some/all variables are required to be integer, and all HPR constraints (plus objective function) are linear.

The first generic branch-and-bound approach to the MIBLP's has been given in [7], where the authors propose to solve HPR embedded into a branch-and-bound scheme and basically enumerate bilevel feasible solutions. Recently, [4, 5] proposed a sound branch-and-cut approach that builds upon the ideas from [7] and cuts off integer bilevel infeasible solutions, by adding cuts that exploit the integrality property of the leader and the follower variables. The authors provide an open-source MIBLP solver `MibS` [8]. More recently, [3] again propose to embed HPR into a branch-and-bound tree, bilevel infeasible solutions being cut off by adding a continuous follower subproblem into HPR, each time a new bilevel infeasible solution is detected. Continuous follower subproblems are then reformulated using KKT conditions and linearized in a standard way. Another generic approach for MIBLP's is a branch-and-sandwich method in [6], where the authors propose novel ideas for deriving lower and upper bounds of the follower's value function.

As this is usually the case with intersection cuts for MILPs, our IC's for MIBLP's also use disjunctive arguments. Disjunctive cuts in connection to bilevel linear programming have been investigated in [1], where the continuous follower subproblem is reformulated using KKT conditions, and disjunctive cuts are used to enforce complementary slackness conditions.

3 Bilevel-free sets

The following result is valid for generic bilevel problems and was implicit in some early references (including [9]) where it was only used as a guide for branching.

Lemma 3.1. *For any $\hat{y} \in \mathbb{R}^{n_2}$, the set*

$$S(\hat{y}) = \{(x,y) \in \mathbb{R}^n : f(x,y) \geq f(x,\hat{y}),\, g(x,\hat{y}) \leq 0\} \tag{10}$$

does not contain any bilevel feasible point in its interior.

Proof. It is enough to prove that no bilevel feasible (x, y) exists such that $f(x, y) > f(x, \hat{y})$ and $g(x, \hat{y}) < 0$. We will in fact prove a tighter result where the latter condition is replaced by $g(x, \hat{y}) \leq 0$, as this will be required in the proof of the next theorem. Indeed, for any bilevel feasible solution (x, y) with $g(x, \hat{y}) \leq 0$, one has $f(x, y) \leq \Phi(x) = \min_{y'}\{f(x, y') : g(x, y') \leq 0\} \leq f(x, \hat{y})$. □

In some relevant settings, the above result can be strengthened to obtain the following enlarged bilevel-free set.

Theorem 3.1. *Assume that $g(x,y)$ is integer for all HPR solutions (x,y). Then, for any $\hat{y} \in \mathbb{R}^{n_2}$, the extended set*

$$S^+(\hat{y}) = \{(x,y) \in \mathbb{R}^n : f(x,y) \geq f(x,\hat{y}),\ g(x,\hat{y}) \leq 1\} \tag{11}$$

does not contain any bilevel feasible point in its interior, where 1 denotes a vector of all ones.

Proof. To be in the interior of $S^+(\hat{y})$, a bilevel feasible (x,y) should satisfy $f(x,y) > f(x,\hat{y})$ and $g(x,\hat{y}) < 1$. By assumption, the latter condition can be replaced by $g(x,\hat{y}) \leq 0$, hence the claim follows from the proof of previous lemma. □

As far as we know, the above result is new. In spite of its simplicity, it will play a fundamental role in our solution method.

4 Mixed-integer bilevel linear programming

In the remaining part of the paper we will focus on the case where some/all variables are required to be integer, and all HPR constraints (plus objective function) are linear. This leads to the following Mixed-Integer Bilevel Linear Program (MIBLP):

$$\begin{aligned}
\min\ & F(x,y) && (12)\\
& G(x,y) \leq 0 && (13)\\
& g(x,y) \leq 0 && (14)\\
& (x,y) \in \mathbb{R}^n && (15)\\
& f(x,y) \leq \Phi(x) && (16)\\
& x_j \text{ integer}, \quad \forall j \in J_1 && (17)\\
& y_j \text{ integer}, \quad \forall j \in J_2, && (18)
\end{aligned}$$

where F, G, f, g are now assumed to be affine functions, sets $J_1 \subseteq \{1, \cdots, n_1\}$ and $J_2 \subseteq \{1, \cdots, n_2\}$ identify the (possibly empty) indices of the integer-constrained variables in x and y, respectively, and the value function reads

$$\Phi(x) = \min_{y \in \mathbb{R}^{n_2}} \{f(x,y) : g(x,y) \leq 0,\ y_j \in \mathbb{Z}\ \ \forall j \in J_2\}. \tag{19}$$

Dropping (16) leads to the HPR, which is a MILP in this setting. Dropping integrality conditions as well leads to the LP relaxation of HPR, namely (12)–(15), an LP which will be denoted by $\overline{\text{HPR}}$.

Our main goal is to solve the above MIBLP by using a standard simplex-based branch-and-cut algorithm where the hard constraint (16) is enforced, on the fly, by adding cutting planes. The minimal requisite for the correctness of such an approach is the ability of cutting any *vertex*, say (x^*, y^*), of $\overline{\text{HPR}}$ which satisfies the integrality requirements (17)–(18) but is bilevel infeasible because

$$f(x^*, y^*) > \Phi(x^*), \tag{20}$$

thus preventing a wrong update of the incumbent. To this end, we will propose a novel application of Balas' intersection cuts [2] in the MIBLP context.

5 A new family of cuts for MIBLP

Intersection cuts (IC's) for a given (x^*, y^*) require the definition of two sets: (1) a cone pointed at (x^*, y^*) that contains all the bilevel feasible solutions, and (2) a convex set S^* that contains (x^*, y^*) but no bilevel feasible solutions in its interior. The reader is referred to [2] for technical details.

As customary in mixed-integer programming, IC's are generated for vertices (x^*, y^*) of an LP relaxation of the problem to be solved, so a suitable cone is just the corner polyhedron associated with the corresponding optimal basis. All relevant information in this cone is readily available in the "optimal tableau" and requires no additional computational effort.

As to the convex set S^*, we propose to use the set defined in Lemma 3.1 (or, better, in Theorem 3.1 if applicable) by choosing

$$\hat{y} = \arg\min_y \{f(x^*, y) : g(x^*, y) \leq 0,\ y_j \in \mathbb{Z}\ \forall j \in J_2\} \tag{21}$$

(assuming this problem is not unbounded). Indeed, such a set S^* does not contain any bilevel feasible point in its interior, as required, while $(x^*, y^*) \in S^*$ because of (20) and $\Phi(x^*) = f(x^*, \hat{y})$ by definition. Note that $\hat{y}$ is well defined when (x^*, y^*) is a solution of HPR, and that S^* is a convex polyhedron in the MIBLP case.

However, an important property is still missing, namely, (x^*, y^*) must belong to the *interior* of S^* if we want to generate a violated intersection cut. This is always the case for MILBP's for which S^* is the *extended* set defined as in Theorem 3.1. This includes problems with *all-integer follower* where $J_2 = \{1, \cdots, n_2\}$, all g-coefficients are integer, and $j \in J_1$ for all x_j's appearing with nonzero coefficient in some follower constraint.

A relevant consequence of the above discussion is that, at least in the all-integer follower case, an exact branch-and-cut MIBLP solver can be obtained from a MILP solver by just adding a separation function for IC's based on the extended set $S^+(\hat{y})$ defined by (11) and (21). Indeed, observe that an exact MIBLP solver can be obtained by applying a general-purpose simplex-based MILP solver to HPR. To avoid the incumbent be updated with bilevel infeasible solutions, it is enough to cut any HPR solution (x^*, y^*) with $f(x^*, y^*) > \Phi(x^*)$. Without loss of generality, by disabling internal MILP heuristics, we can assume that (x^*, y^*) is a *vertex* of the current $\overline{\text{HPR}}$ so we can always cut it by an (locally-valid) IC as, by definition, (x^*, y^*) is in the interior of the extended $S^+(\hat{y})$ when $\hat{y}$ is defined as in (21). In addition, assuming that all leader's variables x are integer and bounded, the number of HPR solutions to cut is finite, so a finite number of branching nodes (and hence of IC's) will be generated, i.e., the method converges in a finite number of iterations.

In the heuristic attempt of producing violated IC's for a generic vertex (x^*, y^*) of the $\overline{\text{HPR}}$ polyhedron, one could also consider the following alternative definition of the point $\hat{y}$ that defines the bilevel-free sets

$$\begin{aligned}(\hat{y}, \hat{d}) = \arg\max_{y,d} \{d :\ & f(x^*, y) + \varphi\, d \leq f(x^*, y^*),\\ & g(x^*, y) + \gamma d \leq 0,\ y_j \in \mathbb{Z}\ \forall j \in J_2\},\end{aligned} \tag{22}$$

where $\varphi \in \mathbb{R}_+$ and $\gamma \in \mathbb{R}^{m_2}_+$ are suitable normalization factors, e.g., the Euclidean norm of the corresponding left-hand-side coefficient vectors. The rationale of this definition is that one wants to detect a bilevel-free set $S(\hat{y})$ whose closest face to (x^*, y^*) has a maximum distance from it.

Example. Figure 1 illustrates the application of IC's on an example given in [7], which is frequently used in the literature:

$$\min_{x \in \mathbb{Z}} \ -x - 10y \tag{23}$$

$$y \in \arg\min_{y' \in \mathbb{Z}} \{\ y' : \tag{24}$$

$$-25x + 20y' \leq 30 \tag{25}$$

$$x + 2y' \leq 10 \tag{26}$$

$$2x - y' \leq 15 \tag{27}$$

$$2x + 10y' \geq 15\ \}. \tag{28}$$

In this all-integer example, there are 8 bilevel feasible points (depicted as crossed squares in Figure 1), and the optimal bilevel solution is $(2, 2)$. The drawn polytope corresponds to the $\overline{\text{HPR}}$ feasible set.

We first apply the definition of the bilevel-free set from Lemma 3.1 with $\hat{y}$ defined as in (21). After solving the first $\overline{\text{HPR}}$, the point $A = (2, 4)$ is found. This point is bilevel infeasible, as for $x^* = 2$ we have $f(x^*, y^*) = y^* = 4$ while $\Phi(x^*) = 2$. From (21) we compute $\hat{y} = 2$ and the intersection cut derived from the associated $S(\hat{y})$ is depicted in Figure 1(a). In the next iteration, the optimal $\overline{\text{HPR}}$ solution moves to $B = (6, 2)$. Again, for $x^* = 6$, $f(x^*, y^*) = y^* = 2$ while $\Phi(x^*) = 1$. So we compute $\hat{y} = 1$ and generate the IC induced by the associated $S(\hat{y})$, namely $2x + 11y \leq 27$ (cf. Figure 1(b)). In the next iteration, the fractional point $C = (5/2, 2)$ is found and $\hat{y} = 1$ is again computed. In this case, C is not in the interior of $S(\hat{y})$ so we cannot generate an IC cut from C but we should proceed and optimize HPR to integrality by using standard MILP tools such as MILP cuts or branching. This produces the optimal HPR solution $(2, 2)$ which is bilevel feasible and hence optimal.

We next apply the definition of the enlarged bilevel-free set from Theorem 3.1 (whose assumption is fulfilled) with $\hat{y}$ defined again as in (21); see Figures 1(c) and (d). After the first iteration, the point $A = (2, 4)$ is cut off by a slightly larger $S^+(\hat{y} = 2)$, but with the same IC as before ($y \leq 2$). After the second iteration, from the bilevel infeasible point $B = (6, 2)$ we derive a larger set $S^+(\hat{y} = 1)$ and a stronger IC ($x + 6y \leq 14$). In the third iteration, solution $D = (2, 2)$ is found which is the optimal bilevel solution, so no branching at all is required in this example.

6 Informed no-good cuts

A known drawback of IC's is their dependency on the LP basis associated with the point to cut, which can create cut accumulation in the LP relaxation and hence shallow cuts and numerical issues. Moreover, IC's are not directly applicable if the point to cut is not a vertex of a certain LP relaxation of the problem at hand, as it happens e.g. when it is computed by the internal MILP heuristics.

We next describe a general-purpose variant of IC's whose derivation does not require any LP basis and is based on the well-known interpretation of IC's as disjunctive cuts. It turns out that the resulting inequality is valid and violated by any bilevel infeasible solution of HPR in the relevant special case where all x and y variables are binary.

Suppose we are given a point $\xi^* = (x^*, y^*) \in \mathbb{R}^n$ and a polyhedron $S^* = \{\xi \in \mathbb{R}^n : \alpha_i^T \xi \leq \alpha_{i0}, i = 1, \cdots, k\}$ whose interior contains ξ^* but no feasible points. Assume that variable-bound constraints $l \leq \xi \leq u$ are present, where some entries of l or u can be $-\infty$ or $+\infty$, respectively. Given ξ^*, define $L := \{j : \xi_j^* - l_j \leq u_j - \xi_j^*\}$ and $U := \{1, \cdots, n\} \setminus L$ and the corresponding linear mapping $\xi \mapsto \overline{\xi} \in \mathbb{R}^n$ with $\overline{\xi}_j := \xi_j - l_j$ for $j \in L$, and $\overline{\xi}_j := u_j - \xi_j$ for $j \in U$ (variable shift and complement).

By assumption, any feasible point ξ must satisfy the disjunction

$$\bigvee_{i=1}^{k} \{\, \xi \in \mathbb{R}^n : \sum_{j=1}^{n} \alpha_{ij} \xi_j \geq \alpha_{i0} \,\}, \tag{29}$$

whereas ξ^* violates all the above inequalities. Now, each term of (29) can be rewritten in terms of $\overline{\xi}$ as

$$\sum_{j=1}^{n} \overline{\alpha}_{ij}\, \overline{\xi}_j \geq \overline{\beta}_i := \alpha_{i0} - \sum_{j \in L} \alpha_{ij} l_j - \sum_{j \in U} \alpha_{ij} u_j, \tag{30}$$

with $\overline{\alpha}_{ij} := \alpha_{ij}$ if $j \in L$, $\overline{\alpha}_{ij} = -\alpha_{ij}$ otherwise. If $\overline{\beta}_i > 0$ for all $i = 1, \cdots, k$, one can normalize the above inequalities to get $\sum_{j=1}^{n} (\overline{\alpha}_{ij} / \overline{\beta}_i)\, \overline{\xi}_j \geq 1$ and derive the valid disjunctive cut in the $\overline{\xi}$ space

$$\sum_{j=1}^{n} \overline{\gamma}_j \overline{\xi}_j \geq 1, \tag{31}$$

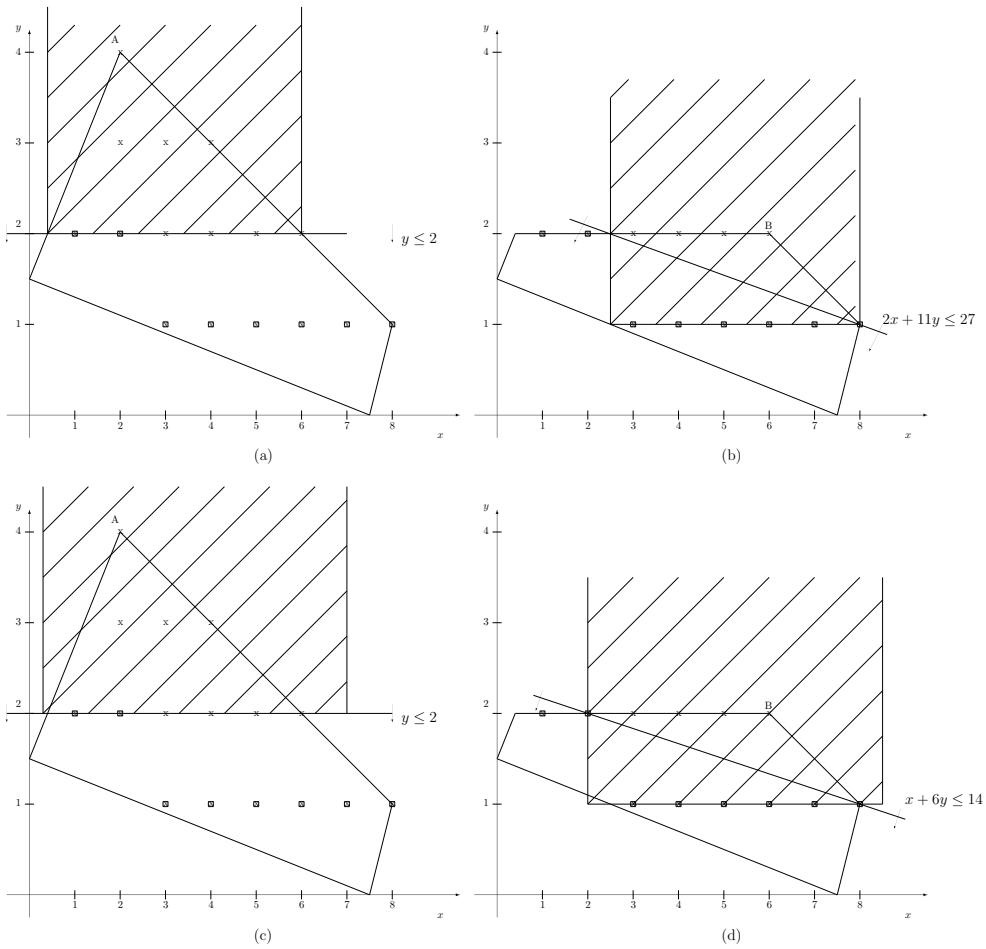

Figure 1: Illustration of the effect of alternative intersection cuts for a notorious example from [7]. Shaded regions correspond to the bilevel-free sets for which the cut is derived.

where $\overline{\gamma}_j := \max\{\overline{\alpha}_{ij}/\overline{\beta}_i : i = 1, \cdots, k\}$, and then one can transform it back to the ξ space in the obvious way. It is easy to see that, in case $\xi^*_j \in \{l_j, u_j\}$ for all $j = 1, \cdots, n$, the above cut is indeed valid (because $\overline{\beta} > 0$) and obviously violated as $\overline{\xi}^* = 0$. In all other cases, the above cut separation is just heuristic.

Inequality (31) will be called *Informed No-Good* (ING) cut as it can be viewed as a strengthening of the following no-good cut often used for bilevel problems with all-binary variables—and in many other Constraint Programming (CP) and Mathematical Programming (MP) contexts:

$$\sum_{j \in L} \xi_j + \sum_{j \in U} (1 - \xi_j) \ge 1. \tag{32}$$

The cut above corresponds to the very generic choice

$$S^* = \{\xi \in \mathbb{R}^n : \xi_j \le 1 \, \forall j \in L, \; 1 - \xi_j \le 1 \, \forall j \in U\}$$

and is violated by ξ^* but is satisfied by any other binary point, hence resulting into a very weak cut. To the best our knowledge, ING cuts are new; they will hopefully be useful in other CP and MP contexts.

7 Preliminary computational results

To evaluate the performance of our new cuts, we embedded them within the general-purpose MILP solver IBM ILOG Cplex 12.6.2 using callbacks, resulting into a branch-and-cut (B&C) MIBLP approach. Internal Cplex's heuristics as well preprocessing have been deactivated in all experiments. IC separation is applied at the root node on all LP solutions (in the so-called usercut callback), while for the remaining nodes it is only applied to integer solutions (lazycut callback). For fractional solutions, IC's whose normalized violation is too small are just skipped. All generated cuts are treated as local cuts (even if no-good and ING cuts would be globally valid) as this reduces the node LP size and significantly improves node throughput. To improve the quality of IC cuts, the bilevel-free set is enlarged by removing all its defining inequalities $\alpha^T(x, y) \leq \alpha_0$ (say) such that imposing the reverse condition $\alpha^T(x, y) \geq \alpha_0$ would trivially lead to an infeasible HPR relaxation due to the current bounds on the x and y variables (this step turns out to be very important for the success of our method). More implementation details will be given in the full paper.

We first compared our code with the one in [3] on the testbed proposed therein. All such instances turned out to be very easy, both for our approach and for `MibS`. More precisely, each instance could be solved in less than a second by our code and in at most 3 seconds by `MibS`, i.e., both codes were 2-3 orders of magnitude faster than the one in [3]. Therefore we addressed more difficult instances, obtained according to the following procedure.

We took a familiar testbed (MILPLIB 3.0) that contains instances that are easily solvable by modern MILP solvers (except instance `seymour` which is very hard even as a MILP). As we planned to also run the open-source MIBLP solver `MibS` [8] to check our code, we skipped all instances involving equations or continuous variables, as well as those involving noninteger coefficients—all the above cases being not supported by the current release of `MibS`. This produced a set of 10 basic 0-1 MILP instances, that we converted into bilevel problems by labeling the first $Y\%$ (rounded up) variables as y's, and the remaining ones as x's. In our test, we considered the three cases with $Y \in \{10, 50, 90\}$ leading to instances named `name-0.1.mps`, `name-0.5.mps`, and `name-0.9.mps`, respectively. All constraints in the resulting model belong to the follower subproblem, as `MibS` cannot handle leader-specific constraints $G(x, y) \leq 0$, while the objective function is used as the leader's objective $F(x, y)$. Finally, the follower's objective is defined as $f(x, y) = -F(x, y)$.

In Table 1, we use `MibS` to assess the computational difficulty of the instances we generated. The table also reports results for our basic B&C code (with IC's but not ING cuts) when run in single-thread mode and with internal Cplex cuts disabled. Note that the two solvers cannot be compared directly, as they are based on a different underlying MILP code, namely: Cplex for our code, and COIN-OR (BLIS) plus Cplex for `MibS`. For both codes, we report in Table 1 the following values: the best obtained upper bound (UB), the best obtained lower bound (LB), the final percentage gap (%gap) calculated as (UB - LB) / UB $\times$ 100. Computing times (t.[s]) are wall-clock seconds on an Intel Xeon E5-2670v2 @ 2.5Ghz computer with 12GB ram. The timelimit was set to 600 sec.s as larger values produced memory issues for some instances where the number of tree nodes is very large. If the time-limit was reached, this is notified as "TL" in the time column. These results clearly indicate that we managed to generate a testbed which is sufficiently challenging for state-of-the-art MIBLP solvers.

Table 2 compares four settings for our code: (1) only no-good cuts are generated, (2) only ING cuts are generated, (3) only IC's are generated, and (4) IC's are generated for fractional solutions at root node, while only ING cuts generated for integer ones. Note that all settings lead to an exact method as all instances in our testbed are pure binary. All versions were run in 4-thread opportunistic mode, without disabling internal Cplex cuts, on a Intel Xeon E3-1220V2 quadcore PC @ 3.10GHz with 16GB of RAM. Setting (1) is intended to assess the difficulty of the created data set for a method built on top of Cplex, but using the most basic MIBLP cuts (no-good). Setting (2) is intended to measure the performance improvement obtained by replacing generic no-good cuts with bilevel-specific ING cuts, while the impact of IC's is addressed in setting (3). Finally, setting (4) combines IC's and ING cuts to limit the negative effect of cut accumulation in the LP basis.

For each of the four setting and for each instance, in Table 2 we report the same information as in Table 1, plus the overall number of branch-and-bound nodes (#nodes).

Table 1: Instance difficulty when using two different MIBLP solvers

	Mibs				B&C with IC's			
name	UB	LB	%gap	t.[s]	UB	LB	%gap	t.[s]
fast0507-0.1	-	173	100.00	TL	12553	173	98.62	TL
fast0507-0.5	-	173	100.00	TL	61503	174	99.72	TL
fast0507-0.9	-	173	100.00	TL	109916	109916	0.00	7
lseu-0.1	1120	1120	0.00	4	1120	1120	0.00	2
lseu-0.5	2400	1205	49.79	TL	2263	1235	45.43	TL
lseu-0.9	5838	1171	79.94	TL	5838	1275	78.75	TL
p0033-0.1	3089	3089	0.00	0	3089	3089	0.00	0
p0033-0.5	3095	3095	0.00	0	3095	3095	0.00	0
p0033-0.9	4679	4679	0.00	90	4679	4679	0.00	3
p0201-0.1	12615	7859	37.70	TL	12465	7931	36.37	TL
p0201-0.5	14220	7832	44.92	TL	13910	7925	43.03	TL
p0201-0.9	15025	7809	48.03	TL	15025	7925	47.25	TL
p0282-0.1	261188	258435	1.05	TL	260781	260067	0.27	TL
p0282-0.5	276338	258432	6.48	TL	272659	259331	4.89	TL
p0282-0.9	724572	258427	64.33	TL	636846	284519	55.32	TL
p0548-0.1	-	317	100.00	TL	10982	8691	20.86	TL
p0548-0.5	-	317	100.00	TL	22450	8620	61.60	TL
p0548-0.9	-	317	100.00	TL	48959	8694	82.24	TL
p2756-0.1	-	2691	100.00	TL	12765	2734	78.58	TL
p2756-0.5	-	2691	100.00	TL	23976	2723	88.64	TL
p2756-0.9	-	2691	100.00	TL	35867	2733	92.38	TL
seymour-0.1	-	407	100.00	TL	480	407	15.21	TL
seymour-0.5	-	407	100.00	TL	823	408	50.43	TL
seymour-0.9	-	407	100.00	TL	1251	1251	0.00	2
stein27-0.1	18	18	0.00	0	18	18	0.00	1
stein27-0.5	19	19	0.00	7	19	19	0.00	3
stein27-0.9	24	20	16.67	TL	24	24	0.00	0
stein45-0.1	30	30	0.00	103	30	30	0.00	32
stein45-0.5	33	31	6.06	TL	32	32	0.00	205
stein45-0.9	40	31	22.50	TL	40	40	0.00	0

The influence of IC's to the performance of the B&C can be measured by comparing the quality of lower bounds of the setting (3), with the settings (1) and (2). In 14, respectively 11 cases, the LBs obtained by IC's are strictly stronger than those obtained by pure no-good and ING cuts, respectively. The quality of lower bounds when IC's are combined with ING cuts remains roughly the same across all instances. As expected, the setting (1) exhibits the worst performance with 22 instances remaining unsolved within the given time-limit. ING cuts perform better (in particular considering the quality of lower bounds), but still with 20 instances remaining unsolved. Both settings with IC's and IC's with ING cuts manage to solve 12 instances to optimality. The number of enumerated branch-and-bound nodes varies strongly between the instances, even between those being derived from the same MIPLIB source. This indicates that, despite the fact that some instances are derived from the identical HPR formulation, the difficulty is mainly determined by the structure of the follower subproblem.

Table 2: Comparison of different settings of our B&C approach.

	No-good cuts only					ING cuts only					IC's only					IC's and ING cuts				
name	UB	LB	%gap	t.[s]	#nodes	UB	LB	%gap	t.[s]	#nodes	UB	LB	%gap	t.[s]	#nodes	UB	LB	%gap	t.[s]	#nodes
fast0507-0.1	12547	173	98.62	TL	2766	12548	173	98.62	TL	11k	12550	173	98.62	TL	4451	12552	173	98.62	TL	5371
fast0507-0.5	61485	173	99.72	TL	2699	61485	173	99.72	TL	5215	-	5440	100.00	TL	33k	-	5440	100.00	TL	33k
fast0507-0.9	109928	173	99.84	TL	2697	109928	173	99.84	TL	864	109916	109916	0.00	4	2	109916	109916	0.00	4	2
lseu-0.1	1120	1120	0.00	0	38	1120	1120	0.00	0	40	1120	1120	0.00	0	39	1120	1120	0.00	0	40
lseu-0.5	2314	1219	47.32	TL	141k	2263	1324	41.49	TL	1M	2263	1318	41.76	TL	2M	2274	1323	41.82	TL	1M
lseu-0.9	5838	1213	79.22	TL	128k	5838	1355	76.79	TL	2M	5838	1384	76.29	TL	2M	5838	1385	76.28	TL	2M
p0033-0.1	3089	3089	0.00	0	2	3089	3089	0.00	0	2	3089	3089	0.00	0	2	3089	3089	0.00	0	2
p0033-0.5	3095	3095	0.00	0	42	3095	3095	0.00	0	45	3095	3095	0.00	0	41	3095	3095	0.00	0	43
p0033-0.9	4679	4679	0.00	9	11k	4679	4679	0.00	1	4646	4679	4679	0.00	1	4071	4679	4679	0.00	1	3355
p0201-0.1	12610	7802	38.13	TL	126k	12495	7915	36.65	TL	794k	12345	7945	35.64	TL	944k	12345	7922	35.83	TL	738k
p0201-0.5	13925	7803	43.96	TL	117k	13910	7932	42.98	TL	922k	13920	7944	42.93	TL	1M	13850	7945	42.64	TL	965k
p0201-0.9	15025	7804	48.06	TL	115k	15025	7925	47.25	TL	718k	15025	7933	47.20	TL	722k	15025	7927	47.24	TL	716k
p0282-0.1	260781	258431	0.90	TL	102k	260781	258448	0.89	TL	2M	260781	258449	0.89	TL	3M	260781	258448	0.89	TL	2M
p0282-0.5	274422	258432	5.83	TL	120k	274422	258447	5.82	TL	2M	274422	258448	5.82	TL	3M	274422	258447	5.82	TL	2M
p0282-0.9	685640	258432	62.31	TL	124k	638243	258446	59.51	TL	2M	639964	271734	57.54	TL	15M	644113	271734	57.81	TL	15M
p0548-0.1	11100	8691	21.70	TL	123k	11100	8691	21.70	TL	365k	11348	8691	23.41	TL	1M	11348	8691	23.41	TL	494k
p0548-0.5	22083	8691	60.64	TL	64k	22078	8691	60.64	TL	76k	22083	8691	60.64	TL	74k	22083	8691	60.64	TL	70k
p0548-0.9	50162	8691	82.67	TL	103k	50162	8691	82.67	TL	220k	50253	9147	81.80	TL	42k	50253	9147	81.80	TL	44k
p2756-0.1	14540	3124	78.51	TL	20k	14430	3124	78.35	TL	38k	13936	3124	77.58	TL	65k	14500	3124	78.46	TL	32k
p2756-0.5	25654	3124	87.82	TL	19k	25654	3124	87.82	TL	44k	23931	3124	86.95	TL	66k	24181	3124	87.08	TL	49k
p2756-0.9	36449	3124	91.43	TL	17k	35242	3124	91.14	TL	182k	34092	3124	90.84	TL	95k	34703	3124	91.00	TL	175k
seymour-0.1	477	415	13.00	TL	29k	477	415	13.00	TL	25k	477	415	13.00	TL	27k	478	415	13.18	TL	25k
seymour-0.5	823	415	49.57	TL	31k	816	415	49.14	TL	37k	823	415	49.57	TL	34k	814	415	49.02	TL	39k
seymour-0.9	1252	415	66.85	TL	31k	1252	415	66.85	TL	23k	1251	1251	0.00	5	2	1251	1251	0.00	5	2
stein27-0.1	18	18	0.00	0	1202	18	18	0.00	0	1244	18	18	0.00	0	1209	18	18	0.00	0	1247
stein27-0.5	19	19	0.00	1	7377	19	19	0.00	1	7060	19	19	0.00	1	6465	19	19	0.00	1	7001
stein27-0.9	24	19	20.83	TL	110k	24	24	0.00	2	13k	24	24	0.00	0	2	24	24	0.00	0	2
stein45-0.1	30	30	0.00	4	14k	30	30	0.00	4	14k	30	30	0.00	5	13k	30	30	0.00	5	14k
stein45-0.5	32	32	0.00	176	211k	32	32	0.00	31	133k	32	32	0.00	37	161k	32	32	0.00	47	202k
stein45-0.9	40	30	25.00	TL	158k	40	40	0.00	234	1M	40	40	0.00	0	2	40	40	0.00	0	2

References

[1] C. Audet, J. Haddad, and G. Savard. Disjunctive cuts for continuous linear bilevel programming. *Optimization Letters*, 1(3):259–267, 2007.

[2] E. Balas. Intersection cuts–a new type of cutting planes for integer programming. *Operations Research*, 19(1):19–39, 1971.

[3] M. Caramia and R. Mari. Enhanced exact algorithms for discrete bilevel linear problems. *Optimization Letters*, 9(7):1447–1468, 2015.

[4] S. DeNegre. *Interdiction and Discrete Bilevel Linear Programming*. PhD thesis, Lehigh University, 2011.

[5] S. DeNegre and T. K. Ralphs. A branch-and-cut algorithm for integer bilevel linear programs. In *Operations research and cyber-infrastructure*, pages 65–78. Springer, 2009.

[6] P.-M. Kleniati and C. S. Adjiman. A generalization of the branch-and-sandwich algorithm: From continuous to mixed-integer nonlinear bilevel problems. *Computers & Chemical Engineering*, 72:373 – 386, 2015.

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

[8] T. K. Ralphs. MibS. `https://github.com/tkralphs/MibS`.

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

Acknowledgment

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 the University of Padova (Progetto di Ateneo "Exploiting randomness in Mixed Integer Linear Programming"), and by MiUR, Italy (PRIN project "Mixed-Integer Nonlinear Optimization: Approaches and Applications"). The work of I. Ljubić and M. Sinnl was also supported by the Austrian Research Fund (FWF, Project P 26755-N19). The authors thank Ted Ralphs for his technical support and instructions regarding `MibS`, and Massimiliano Caramia for providing the instances used in [3].