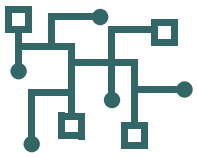


ISE

Industrial and
Systems Engineering

Valid Inequalities for Mixed Integer Bilevel Linear Optimization Problems

Sahar Tahernejad

Lindo Systems, Inc., Chicago, IL

Ted K. Ralphs

Department of Industrial and System Engineering, Lehigh University, Bethlehem, PA

COR@L Technical Report 20T-013

Valid Inequalities for Mixed Integer Bilevel Linear Optimization Problems

Sahar Tahernejad*[1] and Ted K. Ralphs†[2]

[1]Lindo Systems, Inc., Chicago, IL
[2]Department of Industrial and System Engineering, Lehigh University, Bethlehem, PA

October 30, 2020

Abstract

Despite the success of branch-and-cut methods for solving mixed integer bilevel linear optimization problems (MIBLPs) in practice, there have remained some gaps in the theory surrounding these methods. In this paper, we take a first step towards laying out a theory of valid inequalities and cutting-plane methods for MIBLPs that parallels the existing theory for mixed integer linear optimization problems (MILPs). We provide a general scheme for classifying valid inequalities and illustrate how the known classes of valid inequalities fit into this categorization. We also introduce new classes of valid inequalities—one based on a generalization of Chvátal inequalities for MILPs and several more based on a notion of parametric inequality similar to subadditive inequalities for MILPs. Finally, we compare the performance of all classes discussed in the paper using the open source solver `MibS`.

1 Introduction

Bilevel optimization (and multilevel optimization, more generally) provides a framework for modeling and solution of optimization problems in which decisions are made in multiple time stages by multiple (possibly competing) decision-makers (DMs). Such optimization problems arise in a wide array of applications, with ever more coming to light as computational tools for solution of such problems become more widely available. For an overview of bilevel optimization and its applications, we refer the reader to Dempe [2002].

Bilevel optimization problems are difficult to solve both in an empirical and theoretical sense, even in the simple case of (continuous) bilevel linear optimization problems (BLPs), in which the constraints and objective functions must all be linear and the variables are all continuous. The class of problems known as *mixed integer bilevel linear optimization problems* (MIBLPs) is that

*sahar@lindo.com
†ted@lehigh.edu

in which the constraints and objective functions must be linear, but some variables may also be required to take on integer values. While BLPs are hard for the complexity class NP, MIBLPs are hard for the class Σ_2^p, one level higher in the so-called polynomial time hierarchy [Stockmeyer, 1976].

The first algorithm proposed for solution of MIBLPs was the (pure) branch-and-bound algorithm of Bard and Moore [1990]. Two decades later, DeNegre and Ralphs [2009] introduced the idea that this basic framework could be used successfully as the basis for a branch-and-cut algorithm much like those that revolutionized the solution of mixed integer linear optimization problems (MILPs). Just as in the MILP case, approximation of the convex hull of feasible solutions by valid inequalities has since proved to be an effective means by which to improve bounds obtained by the (typically very weak) linear optimization problem (LP) relaxation employed in the original branch-and-bound algorithm. The key component necessary for any branch-and-cut algorithm to be effective is a method of separating the solution to the LP relaxation from the convex hull of feasible solutions by the dynamic generation of strong valid inequalities [Gomory, 1958, Cook et al., 1990, Balas et al., 1993a].

Much of the theory that has been developed for the MILP case and has been exploited so successfully in the development of MILP solvers can be generalized to the MIBLP case. Naturally, however, these is no free lunch—the separation problem for the convex hull of feasible solutions of an MIBLP is both theoretically and practically much more difficult than in the MILP case. The computational results do in fact bear this conclusion out, as will be discussed in the last section of the paper.

The main contributions of this paper are as follows. In Section 2, we lay the foundations for a theory of valid inequalities paralleling the one that already exists in the MILP case. Building on this foundation, Section 3 provides an overview of the known techniques for generation of valid inequalities for MIBLPs and describes how they fit into the basic framework. Section 4 introduces methods for strengthening valid inequalities using principles of duality and introduces new classes of strong valid inequalities generated using these principles. Finally, Section 6 describes computational experiments carried out to test the relative effectiveness of these inequalities using the purpose-built open-source solver `MibS` [DeNegre et al., 2019], described in [Tahernejad et al., 2020].

1.1 Mixed Integer Bilevel Optimization

To describe the class of MIBLPs more formally, let $x \in X$ represent the set of variables controlled by the *first-level* DM or *leader* and $y \in Y$ denote the set of variables controlled by the *second-level* DM or *follower*, where $X = \mathbb{Z}_+^{r_1} \times \mathbb{R}_+^{n_1 - r_1}$ and $Y = \mathbb{Z}_+^{r_2} \times \mathbb{R}_+^{n_2 - r_2}$. The general form of an MIBLP is

$$\min_{x \in X} \{cx + \Xi(x)\}, \tag{MIBLP}$$

where the function Ξ is a *risk function* that encodes the part of the objective value of x that depends on the response to x in the second level. This function may have different forms, depending on the precise variant of the bilevel problem being solved. In this paper, we focus on the so-called *optimistic* case [Loridan and Morgan, 1996], in which

$$\Xi(x) = \min \left\{ d^1 y \;\middle|\; y \in \mathcal{P}_1(x), y \in \operatorname{argmin}\{d^2 y \;|\; y \in \mathcal{P}_2(x) \cap Y\} \right\}, \tag{RF}$$

where

$$\mathcal{P}_1(x) = \left\{ y \in \mathbb{R}_+^{n_2} \;\middle|\; G^1 y \geq b^1 - A^1 x \right\}$$

is a parametric family of polyhedra containing points satisfying the linear constraints of the first-level problem with respect to a given $x \in \mathbb{R}^{n_1}$ and

$$\mathcal{P}_2(x) = \left\{ y \in \mathbb{R}^{n_2}_+ \;\middle|\; G^2 y \geq b^2 - A^2 x \right\}$$

is a second parametric family of polyhedra containing points satisfying the linear constraints of the second-level problem with respect to a given $x \in \mathbb{R}^{n_1}$. The input data is $A^1 \in \mathbb{Q}^{m_1 \times n_1}$, $G^1 \in \mathbb{Q}^{m_1 \times n_2}$, $b^1 \in \mathbb{Q}^{m_1}$, $A^2 \in \mathbb{Q}^{m_2 \times n_1}$, $G^2 \in \mathbb{Q}^{m_2 \times n_2}$ and $b^2 \in \mathbb{Q}^{m_2}$. As is customary, we define $\Xi(x) = \infty$ when either $x \notin X$ or the problem on the right-hand side of (RF) is infeasible. We consider the general formulation that allows participation of the second-level variables in the first-level constraints. The matrix G^1 is defined as a matrix of zeros (with appropriate dimensions) in the case in which the first-level constraints are independent of the second-level variables.

Another formulation for MIBLPs involves the *value function* of the second-level problem instead of the risk function. This formulation is given by

$$\min \left\{ cx + d^1 y \;\middle|\; x \in X, y \in \mathcal{P}_1(x) \cap \mathcal{P}_2(x) \cap Y, d^2 y \leq \phi(b^2 - A^2 x) \right\}, \tag{MIBLP-VF}$$

where ϕ represents the value function of the second-level problem and is defined as

$$\phi(\beta) = \min \left\{ d^2 y \;\middle|\; G^2 y \geq \beta, y \in Y \right\} \quad \forall \beta \in \mathbb{R}^{m_2}. \tag{VF}$$

This MILP yields the optimal value of the second-level problem corresponding to each right-hand side $\beta \in \mathbb{R}^{m_2}$.

A wide variety of special cases of MIBLPs have been studied in the literature. *Interdiction problems* are one of the most important classes among these special cases. In these problems, with respect to each second-level variable, there exists a binary first-level variable (i.e., $n_1 = n_2 = n$) whose sole responsibility is preventing the second-level variable from getting a non-zero value to promote the second-level objective value. This class of problems can be formulated as

$$\min \left\{ d^1 y \;\middle|\; x \in \mathcal{P}_1^{INT} \cap \mathbb{B}^n, y \in \operatorname{argmax}\{ d^1 y \;|\; y \in \mathcal{P}_2^{INT}(x) \cap Y \} \right\}, \tag{MIPINT}$$

where

$$\mathcal{P}_1^{INT} = \left\{ x \in \mathbb{R}^n_+ \;\middle|\; A^1 x \geq b^1 \right\},$$
$$\mathcal{P}_2^{INT}(x) = \left\{ y \in \mathbb{R}^n_+ \;\middle|\; G^2 y \geq b^2, y \leq \operatorname{diag}(u)(e - x) \right\},$$

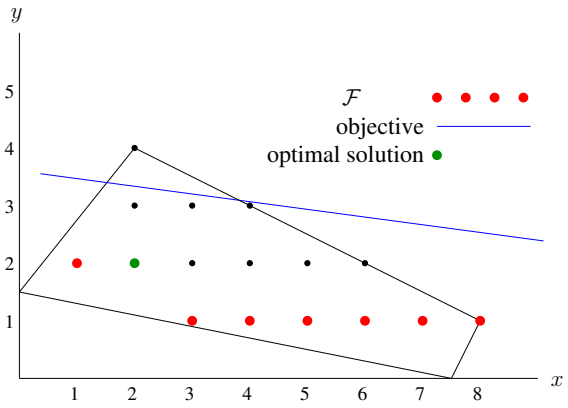
$u \in \mathbb{R}^n_+$ represents the upper bound vector for the second-level variables and e is the n-dimensional vector of ones. As we will explain in Section 3, `MibS` has specialized methods for (MIPINT) and other special classes of MIBLPs.

We now introduce notation used in the remainder of the paper. Dropping the integrality constraints and the optimality constraint of the second-level problem from (MIBLP-VF) results in an LP with feasible region

$$\mathcal{P} = \left\{ (x, y) \in \mathbb{R}^{n_1 \times n_2}_+ \;\middle|\; y \in \mathcal{P}_1(x) \cap \mathcal{P}_2(x) \right\}.$$

This set includes all $(x, y) \in \mathbb{R}^{n_1 \times n_2}_+$ that satisfy the linear constraints of the first- and second-level problems. The subset of $\mathcal{P}$ containing points that also satisfy the integrality constraints in both levels is

$$\mathcal{S} = \mathcal{P} \cap (X \times Y).$$



$$\begin{aligned}
\min_{x \in \mathbb{Z}_+} \quad & -x - 10y \\
\text{s.t.} \quad & y \in \operatorname{argmin} \{ y : \\
& -5x + 4y \leq 6 \\
& x + 2y \leq 10 \\
& 2x - y \leq 15 \\
& 2x + 10y \geq 15 \\
& y \in \mathbb{Z}_+ \}
\end{aligned}$$

Figure 1: The feasible region and optimal solution of the the example from [Moore and Bard, 1990].

With respect to a given $x \in \mathbb{R}^{n_1}_+$, the *rational reaction set* is defined as

$$\mathcal{R}(x) = \operatorname{argmin} \left\{ d^2 y \;\middle|\; y \in \mathcal{P}_2(x) \cap Y \right\}$$

and contains all $y \in Y$ that are optimal to the second-level problem arising from fixing the first-level variables to x. Note that (x, y) may not be bilevel feasible, even if $y \in \mathcal{R}(x)$ if either $x \notin X$ or $y \notin \mathcal{P}_1(x)$. As such, the conditions for bilevel feasibility of $(x, y) \in \mathbb{R}^{n_1 \times n_2}_+$ can be stated as

Feasibility Condition 1. $x \in X$.

Feasibility Condition 2. $y \in \mathcal{P}_1(x) \cap \mathcal{R}(x)$.

Based on these conditions, the *bilevel feasible region* is

$$\mathcal{F} = \{(x, y) \in X \times Y \mid y \in \mathcal{P}_1(x) \cap \mathcal{R}(x)\}.$$

The problem (MIBLP) can thus be re-cast as the optimization problem

$$\min_{(x,y) \in \mathcal{F}} cx + d^1 y. \tag{MIBLP-F}$$

The bilevel feasible region and optimal solution of the well-known example from [Moore and Bard, 1990], shown in Figure 1, illustrates these concepts.

The sets $\mathcal{F}$, $\mathcal{P}$ and $\mathcal{S}$ can also be defined for each node t of the tree resulting from employing the branch-and-cut algorithm for solving (MIBLP). The feasible region of node t differs from the root node by

1. changes to the bounds of variables; and
2. addition of valid inequalities.

Hence, the sets $\mathcal{F}$, $\mathcal{P}$ and $\mathcal{S}$ corresponding to node t are defined as

$$\mathcal{F}^t = \left\{(x,y) \in \mathcal{F} \cap \Pi^t \;\middle|\; l_x^t \leq x \leq u_x^t, l_y^t \leq y \leq u_y^t\right\}, \tag{1}$$

$$\mathcal{P}^t = \left\{(x,y) \in \mathcal{P} \cap \Pi^t \;\middle|\; l_x^t \leq x \leq u_x^t, l_y^t \leq y \leq u_y^t\right\}$$

and

$$\mathcal{S}^t = \mathcal{P}^t \cap (X \times Y),$$

where Π^t represents the polyhedron that is the intersection of the added valid inequalities at node t (which may possibly remove points in $\mathcal{F}$) and possibly at its ancestors. The vectors l_x^t (l_y^t) and u_x^t (u_y^t) denote, respectively, the vectors of lower and upper bounds for the first-level (second-level) variables.

In what follows, we make the following assumptions.

Assumption 1. *$\mathcal{P}$ is bounded.*

This assumption ensures the boundedness of (MIBLP), but it is made primarily for ease of presentation and can be straightforwardly relaxed.

Assumption 2. $\left\{r \in \mathbb{R}_+^{n_2} \;\middle|\; G^2 r \geq 0, d^2 r < 0\right\} = \emptyset$.

This second assumption prevents the unboundedness of the second-level problem (regardless of the first-level solution) and can be checked in a pre-processing step.

Assumption 3. *All first-level variables with at least one non-zero coefficient in the second-level problem (so-called linking variables) are integer, i.e.,*

$$L = \left\{i \in \{1,\ldots,n_1\} \;\middle|\; A_i^2 \neq 0\right\} \subseteq \{1,...,r_1\},$$

where A_i^2 represents the i^{th} column of matrix A^2.

This third assumption guarantees that the optimal solution of (MIBLP) is attainable whenever the optimal solution value is finite [Vicente et al., 1996]. The linking variables are structurally important and recognition of this can streamline the process of solving MIBLPs in several ways. Theorem 1 formally states the special role played by the linking variables.

Theorem 1. *For the vectors x^1 and $x^2 \in \mathbb{R}_+^{n_1}$ with $x_L^1 = x_L^2 \in \mathbb{Z}^L$, we have*

$$\phi(b^2 - A^2 x^1) = \phi(b^2 - A^2 x^2),$$

where x_L^1 and x_L^2 represent the subvector of x^1 and x^2, respectively, corresponding to the linking variables.

The importance of this theorem is that it shows that when the linking variables are fixed, (MIBLP) (which is a non-linear optimization problem in general, due to the non-linearity of the optimality constraint of the second-level problem) becomes an MILP. This is because $\phi(b^2 - A^2 x)$ is a constant whenever the linking variables are fixed. Corollary 1 states this formally.

Corollary 1. *For $\gamma \in \mathbb{Z}^L$, we have*

$$\min\left\{cx + d^1y \mid (x,y) \in \mathcal{F}, x_L = \gamma\right\} = \min\left\{cx + d^1y \mid (x,y) \in \mathcal{S}, d^2y \leq \phi(b^2 - A^2x), x_L = \gamma\right\}. \tag{UB}$$

Hence, the best bilevel feasible solution $(x,y) \in \mathcal{F}$ with $x_L = \gamma \in \mathbb{Z}^L$ can be obtained by solving the problem (*UB*) which can be solved by using an MILP solver.

2 Theoretical Foundations

The theory of valid inequalities for polyhedra and other closed convex sets is well-known and supported by deep foundations developed over several decades (see, e.g., Grötschel et al. [1993]). Although much of the theory was developed specifically with the case of MILPs in mind, it can be applied in other contexts, such as to the solution of MIBLPs. The main tools used in applying this theory to MILPs are *convexification* and the well-known result of Grötschel et al. [1993] that optimization over a closed convex set is polynomially equivalent to the so-called *separation problem* associated with the set. Essentially, we optimize over the convex hull of feasible solutions rather than the original feasible region, thereby transforming the original non-convex problem into an equivalent convex one.

2.1 Convexification

It may not be immediately obvious that convexification and separation can be employed in the context of MIBLPs, so we first show that MIBLPs can, in theory, be solved by a so-called *cutting plane method*. Showing this formally involves showing that convexifying the feasible region does not change the optimal solution value. This can be done in several steps.

Proposition 1. *Under Assumption 3, $\mathcal{F}$ is closed.*

Proof. Let $\mathcal{S}_\gamma$ be the feasible region of the problem (*UB*) for $\gamma \in \mathbb{Z}^L$. Under Assumption 3, $\mathcal{F}$ is the union of (possibly infinite) disjoint sets $\mathcal{S}_\gamma$ for $\gamma \in \mathcal{F}_{x_L} = \text{proj}_{x_L}(\mathcal{F})$, i.e.,

$$\mathcal{F} = \bigcup_{\gamma \in \mathcal{F}_{x_L}} \mathcal{S}_\gamma. \tag{2}$$

Furthermore, under Assumption 3, for all $(\hat{x}, \hat{y}) \in \mathbb{R}^{n_1+n_2}$, there is at least one neighborhood that intersects at most one of the sets $\mathcal{S}_\gamma$ with $\gamma \in \mathcal{F}_{x_L}$. The radius $\bar{r}$ of this neighborhood can be defined as $0 < \bar{r} < \min\left\{||x_L - \hat{x}_L||^2 \mid x_L \neq \hat{x}_L, x_L \in \mathcal{F}_{x_L}\right\}$ (such $\bar{r}$ exists due to Assumption 3). This results that the collection of sets $\mathcal{S}_\gamma$ for $\gamma \in \mathcal{F}_{x_L}$ is a locally finite collection of $\mathbb{R}^{n_1+n_2}$ under Assumption 3. This and (2) follow that [Munkres, 2014]

$$\text{cl}(\mathcal{F}) = \text{cl}\left(\bigcup_{\gamma \in \mathcal{F}_{x_L}} \mathcal{S}_\gamma\right) = \bigcup_{\gamma \in \mathcal{F}_{x_L}} \text{cl}\left(\mathcal{S}_\gamma\right). \tag{3}$$

We have $\mathrm{cl}(\mathcal{S}_\gamma) = \mathcal{S}_\gamma$ since this set is the feasible region of (UB), which is an MILP. Therefore, from (3), we have

$$\mathrm{cl}(\mathcal{F}) = \bigcup_{\gamma \in \mathcal{F}_{x_L}} \mathcal{S}_\gamma = \mathcal{F}.$$

Closedness of $\mathcal{F}$ follows. □

Theorem 2. *Under Assumptions 1 and 3,* $\mathrm{conv}(\mathcal{F})$ *is a rational polyhedron.*

Proof. Basu et al. [2018] showed that when input data are rational, the closure of $\mathcal{F}$ can be written as the finite union of MILP representable sets (with rational data). Since $\mathcal{F}$ is closed by Proposition 1, we have that

$$\mathcal{F} = \mathrm{cl}(\mathcal{F}) = \bigcup_{i=1}^{k} L(\mathcal{S}_i), \tag{4}$$

where k is a scalar, L denotes a linear transformation (specifically projection) and $\mathcal{S}_i$ is the feasible region of an MILP for $i = 1, ..., k$.

From the fundamental theorem of integer programming [Meyer, 1974] and (4), it follows that

$$\begin{aligned} \mathrm{conv}(\mathcal{F}) &= \mathrm{conv}\left(\bigcup_{i=1}^{k} L(\mathcal{S}_i)\right) = \mathrm{conv}\left(\bigcup_{i=1}^{k} \mathrm{conv}(L(\mathcal{S}_i))\right) \\ &= \mathrm{conv}\left(\bigcup_{i=1}^{k} L(\mathrm{conv}(\mathcal{S}_i))\right) = \mathrm{conv}\left(\bigcup_{i=1}^{k} \mathcal{Q}_i\right), \end{aligned} \tag{5}$$

where $\mathcal{Q}_i = L(\mathrm{conv}(\mathcal{S}_i))$ is a rational polyhedron for $i = 1, ..., k$.

Furthermore, since $\mathcal{Q}_i$ is bounded for $i = 1, ...k$ by Assumption 1, the result follows from the fact that the convex hull of the union of a finite number of bounded polyhedra is a polyhedron [Balas, 1985, 1998]. □

Theorem 3. *Under Assumption 3, we have that*

$$\min_{(x,y)\in\mathcal{F}} cx + d^1 y = \min_{(x,y)\in\mathrm{conv}(\mathcal{F})} cx + d^1 y. \tag{6}$$

Proof. Since the objective function is linear, we have [Conforti et al., 2014]

$$\inf_{(x,y)\in\mathcal{F}} cx + d^1 y = \inf_{(x,y)\in\mathrm{conv}(\mathcal{F})} cx + d^1 y$$

and the infimum of $cx + d^1 y$ is attained over $\mathcal{F}$ if and only if it is attained over $\mathrm{conv}(\mathcal{F})$. This follows the result because the infimum of $cx + d^1 y$ is attained over $\mathcal{F}$ under Assumption 3. □

The problem (6) would be an LP in principle, if we knew a complete description of $\mathrm{conv}(\mathcal{F})$. In such case, a solution of the MIBLP could be obtained by producing an extremal solution to this LP by, e.g., the simplex algorithm. Since we generally do not know a complete description of $\mathrm{conv}(\mathcal{F})$ and cannot construct one efficiently (this would be at least as difficult as solving the original optimization problem), the cutting plane method is to employ the well-known technique of generating an approximation of $\mathrm{conv}(\mathcal{F})$ by solving the separation problem to generate valid inequalities, as described next.

2.2 Cutting Plane Method

Although they are well-known, we first review several standard definitions and results for convenience before describing the cutting plane method in broad outline.

Definition 1. *A* valid inequality *for the set $\mathcal{F}$ is a triple $(\alpha^x, \alpha^y, \beta)$, where $(\alpha^x, \alpha^y) \in \mathbb{Q}^{n_1+n_2}$ is the* coefficient vector *and $\beta \in \mathbb{Q}$ is a* right-hand side*, such that*

$$\mathcal{F} \subseteq \left\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \alpha^x x + \alpha^y y \geq \beta\right\}.$$

It is easy to see that any inequality valid for $\mathcal{F}$ is also valid for the convex hull of $\mathcal{F}$. The so-called *separation problem* for $\text{conv}(\mathcal{F})$ is to generate an inequality valid for $\text{conv}(\mathcal{F})$, but violated by a given vector $(\hat{x}, \hat{y}) \notin \text{conv}(\mathcal{F})$ or to show that the vector is actually a member of $\text{conv}(\mathcal{F})$. Formally, we define the problem as follows.

Definition 2. *The* separation problem *for $\text{conv}(\mathcal{F})$ with respect to a given $(\hat{x}, \hat{y}) \in \mathbb{R}^{n_1+n_2}$ is to determine whether or not $(\hat{x}, \hat{y}) \in \text{conv}(\mathcal{F})$ and if not, to produce an inequality $(\alpha^x, \alpha^y, \beta) \in \mathbb{Q}^{n_1+n_2+1}$ valid for $\text{conv}(\mathcal{F})$ and for which $\alpha^x \hat{x} + \alpha^y \hat{y} < \beta$.*

The process of solving an MIBLP by a standard cutting plane method is initiated by solving a convex relaxation of the original problem (MIBLP). In contrast to MILPs, the problem obtained by removing integrality constraints at both levels is not a relaxation (its feasible region does not necessarily contain $\mathcal{F}$). On the other hand, we have that $\mathcal{F} \subseteq \mathcal{S} \subseteq \mathcal{P}$, so the problem of optimizing over either $\mathcal{P}$ or $\mathcal{S}$ is a valid relaxation. In `MibS`, $\mathcal{P}$ is used as the feasible region of that starting relaxation and the relaxation problem is

$$\min_{(x,y)\in\mathcal{P}} cx + d^1 y. \tag{LR}$$

In the remainder of the paper, we take this as the initial relaxation.

A valid inequality for $\mathcal{F}$ that is violated by an infeasible point of the relaxation is called a *cut* (or *cutting plane*).

Definition 3. *A* cut *is an inequality valid for $\mathcal{F}$, but violated by some $(\hat{x}, \hat{y}) \in \mathcal{P} \setminus \mathcal{F}$ (typically the optimal solution to a relaxation defined with respect to $\mathcal{P}$).*

Let $(\hat{x}, \hat{y})$ be an extremal optimal solution of (LR). Then the cutting plane method consists of the following loop.

1. Determine whether $(\hat{x}, \hat{y}) \in \text{conv}(\mathcal{F})$ or not. Although determining whether a given arbitrary point is in $\text{conv}(\mathcal{F})$ is difficult in general, we exploit the fact that $(\hat{x}, \hat{y})$ is an extremal member of $\mathcal{P}$. Such a point must be a member of $\mathcal{F}$ in order to be a member of $\text{conv}(\mathcal{F})$. Therefore, we need only determine whether it satisfies the constraints that were relaxed. In this case, we must check whether $(\hat{x}, \hat{y}) \in X \times Y$ and $\hat{y} \in \mathcal{R}(\hat{x})$. If so, then $(\hat{x}, \hat{y}) \in \mathcal{F}$ (and $(\hat{x}, \hat{y})$ is an optimal solution), so the solution process is terminated. Otherwise $(\hat{x}, \hat{y}) \notin \mathcal{F}$ and we move to Step 2.

2. Separate $(\hat{x}, \hat{y}) \notin \mathcal{F}$ from $\mathcal{F}$. To do so, we generate a cut $(\alpha^x, \alpha^y, \beta) \in \mathbb{Q}^{n_1+n_2+1}$ which separates $(\hat{x}, \hat{y})$ from $\mathcal{F}$. The existence of such an inequality is guaranteed when $(\hat{x}, \hat{y})$ is an extreme point of $\mathcal{P}$. It is possible that such inequality is not found in cases where the separation problem is being solved approximately. If no inequality is found, the method terminates with a lower bound. Otherwise, move to Step 3.

3. Add the constraint $\alpha^x x + \alpha^y y \geq \beta$ to the relaxation, thereby strengthening it, and repeat the previous steps until either the method terminates or until one of a specified set of termination criteria, e.g., the number of iterations exceeds some pre-defined limit, is satisfied.

Whether or not this method converges finitely depends on exactly how the valid inequalities are generated and what properties they are guaranteed to have. In the case of MILPs, finite cutting plane algorithms for the pure integer and general cases under mild assumptions were, respectively, given by Gomory [1958] and Del Pia and Weismantel [2012] (see [Gade and Küçükyavuz, 2011] for more detailed discussion on the convergence of the cutting plane algorithm). Since the feasible region $\mathcal{P}$ of our assumed initial relaxation is a polyhedron, the bounding problem (LR) is an LP and can be solved by standard algorithms. Any extremal optimal solution of $\mathcal{P}$ is either a member of $\mathcal{F}$ or is *not* contained in $\text{conv}(\mathcal{F})$ and can be separated from it by an inequality valid for $\text{conv}(\mathcal{F})$, as described above. Hence, MIBLPs can, in principle, be solved by a cutting plane method.

2.3 Improving Valid Inequalities

Traditionally, cutting plane methods have been described theoretically as generating only inequalities valid for the entire feasible set. In practice, however, it is well-known that the addition of inequalities removing feasible solutions is not problematic, as long as this does not change the *optimal solution value* of the original problem. In the case of MIBLP, inequalities removing subsets of $\mathcal{F}$ are used routinely and we thus formally define a notion of valid inequality that allows this.

When $(x^*, y^*) \in \mathcal{F}$ and we have that $cx^* + d^1y^* \leq \min_{(x,y)\in(\mathcal{G}\cap\mathcal{F})} cx + d^1y$ for some set $\mathcal{G} \subseteq \mathbb{R}^{n_1+n_2}$, the optimal solution of (MIBLP-F) will be

$$\min_{(x,y)\in\text{conv}(\mathcal{F})} cx + d^1y = \min\left\{cx^* + d^1y^*, \min_{(x,y)\in\text{conv}(\mathcal{F})\setminus\mathcal{G}} cx + d^1y\right\}.$$

In this case, although $\mathcal{G} \cap \text{conv}(\mathcal{F})$ may contain a subset of the feasible region, it can be removed because it does not include any *improving solutions* relative to (x^*, y^*). Based on this discussion, the definition of valid inequality can be generalized as follows.

Definition 4 (Improving Valid Inequality)**.** *An* improving valid inequality *for the set $\mathcal{F}$ with respect to an incumbent $(x^*, y^*) \in \mathcal{F}$, is a triple $(\alpha^x, \alpha^y, \beta) \in \mathbb{R}^{n_1+n_2+1}$ such that*

$$\left\{(x, y) \in \mathcal{F} \;\middle|\; cx + d^1y < cx^* + d^1y^*\right\} \subseteq \left\{(x, y) \in \mathbb{R}^{n_1\times n_2} \;\middle|\; \alpha^x x + \alpha^y y \geq \beta\right\}.$$

In the remainder of paper, the term "valid inequality" is taken to mean "improving valid inequality" unless otherwise stated.

2.4 General Classes

Although many of the classes of valid inequalities for MILPs that have been proposed are for a particular subclass of MILP and exploit specific combinatorial structure, there exist a variety of methods that produce inequalities valid for general MILPs. The reader is referred to Marchand et al. [2002], Wolter [2006], and Cornuéjols [2008] for broad overviews of these approaches and the relationships between the various classes.

Although the general classes employed in solving generic MILPs were conceived specifically for that purpose, the theoretical basis for many of the classes does not actually depend on any specific properties of MILP and they can thus be easily employed in other settings. *Disjunctive inequalities* and *intersection cuts* are two such families of valid inequalities for MILPs that can be easily generalized to the MIBLP case. We describe each of these classes here at a high level in the context of MIBLPs and then discuss specific applications in Section 3.

Disjunctive Inequalities. Disjunctive programming is both a modeling paradigm and a set of algorithmic technique introduced by Balas [1979] based on the concept of what we call a *valid disjunction*.

Definition 5 (Valid Disjunction)**.** *A collection of disjoint sets $X_i \subseteq \mathbb{R}^{n_1+n_2}$ for $i = 1, ..., k$ represents a* valid disjunction *for $\mathcal{F}$ if*

$$\mathcal{F} \subseteq \bigcup_{i=1}^{k} X_i.$$

In this context, the collection $\{X_i\}_{1 \leq i \leq k}$ is called a *disjunctive set*. The branch-and-cut algorithm depends crucially on the identification of valid disjunctions that are violated by the solution to some relaxation. The identified disjunctions are used both for branching and cutting, two essential elements of the branch-and-cut algorithm.

Just as with valid inequalities, the basic notion of valid disjunction can be modified to allow for the possibility that the disjunction does not contain the entire set $\mathcal{F}$, but possibly eliminates some solutions known to be suboptimal. This yields the concept of an *improving valid disjunction*.

Definition 6 (Improving Valid Disjunction)**.** *A collection of disjoint sets $X_i \subseteq \mathbb{R}^{n_1+n_2}$ for $i = 1, ..., k$ represents an* improving valid disjunction *for $\mathcal{F}$ with respect to $(x^*, y^*) \in \mathcal{F}$ if*

$$\left\{(x, y) \in \mathcal{F} \;\middle|\; cx + d^1 y < cx^* + d^1 y^*\right\} \subseteq \bigcup_{i=1}^{k} X_i.$$

In the remainder of paper, the term "valid disjunction" is taken to mean "improving valid disjunction" unless otherwise stated.

Definition 7 (Disjunctive Inequality)**.** *A* disjunctive (valid) inequality *for the set $\mathcal{F}$ with respect to $\mathcal{Q} \supseteq \mathcal{F}$ and a valid disjunction $\{X_i\}_{1 \leq i \leq k}$ is a triple $(\alpha^x, \alpha^y, \beta) \in \mathbb{R}^{n_1+n_2+1}$ such that*

$$\mathcal{Q} \bigcap \left(\bigcup_{i=1}^{k} X_i \right) \subseteq \left\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \alpha^x x + \alpha^y y \geq \beta\right\}.$$

A subclass of the disjunctive inequalities arises from the disjunctions known as *split disjunctions* that have only two terms and are defined as follows.

Definition 8 (Split Disjunction)**.** *Let* $(\pi^x, \pi^y, \pi_0) \in \mathbb{Z}^{n_1+n_2+1}$ *be such that* $\pi_i^x = 0$ *for* $i \geq r_1 + 1$ *and* $\pi_i^y = 0$ *for* $i \geq r_2 + 1$ *(the coefficients of the continuous variables in both first and second levels are zero). Then when*

$$X_1 = \left\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \pi^x x + \pi^y y \leq \pi_0 - 1\right\} \text{ and } X_2 = \left\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \pi^x x + \pi^y y \geq \pi_0\right\}, \quad (7)$$

$\{X_1, X_2\}$ *is a valid disjunction for* $\mathcal{F}$ *called a* split disjunction*.*

The validity of the above disjunction arises from the fact that the inner product of any member of $\mathcal{F}$ with the coefficient vector is an integer and thus all members of $\mathcal{F}$ must belong to either X_1 or X_2. The disjunctive inequalities derived from such a disjunction are known as *split inequalities*.

Definition 9 (Split Inequality)**.** *A* split inequality *for the set* $\mathcal{F}$ *with respect to* $\mathcal{Q} \supseteq \mathcal{F}$ *and a split disjunction* $\{X_1, X_2\}$ *is any inequality valid for* $\mathcal{Q} \cap (X_1 \cup X_2)$*.*

The usual Chvátal inequalities, as defined in the theory of MILPs, are a subclass of the split inequalities in which $X_1 \cap \mathcal{Q} = \emptyset$. We introduce here a class we refer to as *generalized Chvátal inequalities* that differ slightly in form, but are similar in spirit to the usual Chvátal inequalities.

Definition 10 (Generalized Chvátal Inequality)**.** *Let a split disjunction* $\{X_1, X_2\}$ *for set* $\mathcal{F}$ *be given such that* $X_1 \cap \mathcal{Q} = \{(\hat{x}, \hat{y})\}$ *and* $(\hat{x}, \hat{y}) \notin \mathcal{F}$ *for some given* $\mathcal{Q} \supseteq \mathcal{F}$*. Then* (π^x, π^y, π_0) *in* (7) *is itself a valid inequality for the set* $\mathcal{F}$ *known as a* generalized Chvátal inequality *with respect to set* $\mathcal{Q}$*.*

The connection to the Chvátal inequalities in MILP should be clear. In the MILP case, the usual Chvátal inequalities are split inequalities for which we have a proof that X_1 does not contain any feasible solutions to the LP relaxation of the MILP (and hence does not contain any solutions to the MILP itself). We can thus easily conclude that the MILP feasible region is contained in X_2 and derive the associated valid inequality. Here, we extend this idea to allow that X_1 may contain a (single) solution to the relaxation (whose feasible region we can think of as being $\mathcal{Q}$), but that we have an independent proof that the single solution to the relaxation is not contained in the feasible region $\mathcal{F}$ of the MIBLP itself. We thus have a similar proof that $\mathcal{F}$ is contained in the set X_2, yielding a valid inequality as in the MILP case. Note that we could further extend this definition to include cases where X_1 contains a set of feasible solutions to the relaxation, all of which can be proven not to be in $\mathcal{F}$. However, as we currently have no practical application of such a definition, we use the simpler one here.

An obvious question that we address below is how to obtain a split disjunction (π^x, π^y, π_0) satisfying the requirements of the above definition in a practical way. In MILP, a relevant split disjunction can be derived from the tableau using the procedure of Gomory [Gomory, 1960]. One way of viewing this procedure is that we first derive an inequality valid for the feasible region of the LP relaxation by taking a combination of the inequalities in the original formulation. We ensure that the coefficients of the left-hand side satisfy the integrality requirements for a split disjunction either by rounding in some other fashion. When the right-hand side of an inequality derived in this way is fractional, we can round it up to obtain a valid inequality (see Gomory [1960], for more details).

A very similar procedure is possible in the case of MIBLP by taking $\mathcal{Q}$ in Definition 10 to be the LP relaxation $\mathcal{P}$ of (MIBLP) and $(\hat{x}, \hat{y}) \in (X \times Y) \setminus \mathcal{F}$ to be an extremal optimum to the LP relaxation. By taking a combination of the constraints binding at $(\hat{x}, \hat{y})$, we can derive a split disjunction satisfying Definition 10, as detailed later in Theorems 4 and 5.

Intersection Cuts. Another well-known procedure for deriving valid inequalities for MILPs that can be generalized to the MIBLP setting is that of generating a so-called *intersection cut*, originally suggested by Balas [1971]. This procedure has already been extended to more general classes of optimization problems by Bienstock et al. [2016] and was extended to MIBLPs by Fischetti et al. [2018, 2017]. In general, an intersection cut is an inequality valid for for the convex hull of a discrete set $\mathcal{D}$ contained within a polyhedral radial cone $\mathcal{V}(v)$ with vertex $v \notin \mathcal{D}$. Given a closed convex set $\mathcal{C}$ containing v, but no member of $\mathcal{D}$, in its interior, the intersection points of the extreme rays of $\mathcal{V}(v)$ with the set $\mathcal{C}$ lie on a uniquely defined hyperplane that defines an inequality valid for $\text{conv}(\mathcal{D})$. Typically, v arises as an extremal optimum of an LP relaxation of some discrete optimization problem and $\mathcal{V}$ is then the polyhedral cone described by only those inequalities in the description of the feasible region of the LP relaxation that are binding at v.

In the context of MIBLPs, improving valid inequalities can be generated by considering intersection cuts valid for the convex hull of just those feasible solutions that are improving with respect to a current incumbent. We typically wish to separate an extreme point $(\hat{x}, \hat{y})$ of $\mathcal{P}$, which is either not a member of $\mathcal{F}$ or is non-improving with respect to the incumbent. The radial cone is described by the inequalities in the description of $\mathcal{P}$ that are binding at $(\hat{x}, \hat{y})$ and we denote the cone by $\mathcal{V}(\hat{x}, \hat{y})$. This is the case, for example, when the LP relaxation has been solved by a simplex algorithm and $(\hat{x}, \hat{y})$ is an optimal basic feasible solution of the LP relaxation over $\mathcal{P}$ (usually, this LP will have been re-cast in standard form by introducing slack variables, but we describe the procedure here in the original space for simplicity).

Definition 11 (Intersection Cut)**.** *Let $\mathcal{V}(\hat{x}, \hat{y}) \supseteq \mathcal{P}$ be a radial cone with vertex $(\hat{x}, \hat{y})$, an extreme point of $\mathcal{P}$ and let $\mathcal{C} \subseteq \mathbb{R}^{n_1+n_2}$ be a convex set such that $(\hat{x}, \hat{y}) \in \text{int}(\mathcal{C})$ and $\text{int}(\mathcal{C}) \cap \mathcal{F} = \emptyset$. Then the triple $(\alpha^x, \alpha^y, \beta) \in \mathbb{R}^{n_1+n_2+1}$ represents an* intersection cut *with respect to a radial cone when $\{(x, y) \in \mathbb{R}^{n_1+n_2} \mid \alpha^x x + \alpha^y y = \beta\}$ is the unique hyperplane containing the points of intersection of $\mathcal{C}$ with the extreme rays of $\mathcal{V}(\hat{x}, \hat{y})$.*

Note that the size of the defined set $\mathcal{C}$ can impact the strength of the generated intersection cut, so the desire is to find the largest set $\mathcal{C}$ possible.

3 Valid Inequalities for MIBLPs

As in MILP, the main aim of generating a valid inequality is to remove from the feasible region of the relaxation a solution which is not feasible for the original problem, along with as much of the surrounding region as possible. In the case of MILP, all such solutions that arise in a typical cutting plane method violate integrality conditions and we have simple methods of generating violated valid inequalities.

In the context of MIBLPs, the infeasible solutions that arise can violate either Feasibility Conditions 1 or 2 and we may hence want to remove any of the following types of points or set of

points.

C1. An extreme point $(\hat{x}, \hat{y})$ of $\mathcal{P}$ such that $(\hat{x}, \hat{y}) \notin X \times Y$.

C2. An extreme point $(\hat{x}, \hat{y})$ of $\mathcal{P}$ such that $(\hat{x}, \hat{y}) \in X \times Y$, but $\hat{y} \notin \mathcal{R}(\hat{x})$.

C3. All $(x, y) \in \mathbb{R}^{n_1 \times n_2}$ such that $x \in \mathcal{F}_x$ and $x_L = \gamma \in \mathbb{Z}^L$, where $\mathcal{F}_x = \text{proj}_x(\mathcal{F})$.

The cases C1 and C2 occur when the extreme point $(\hat{x}, \hat{y})$ resulting from solving the relaxation problem (LR) is not bilevel feasible. Note that case C1 also arises in cutting plane methods for MILPs, but case C2 is unique to MIBLPs and involves removing a point that is already integral. Case C3 arises when solving the problem (UB), after which all $(x, y) \in \text{conv}(\mathcal{F})$ with $x_L = \gamma \in \mathbb{Z}^L$ can be removed because they must be non-improving feasible solutions.

With respect to the goals of generating valid inequalities stated above, the set of applicable valid inequalities for MIBLPs can be classified roughly as follows, where U is a global upper bound on the optimal solution value

1. Feasibility cuts: Inequalities that are violated by an extreme point of $\mathcal{P}$ of the form described in C1, but are valid for
$$\text{conv}\left(\left\{(x, y) \in \mathcal{S} \;\middle|\; cx + d^1 y < U\right\}\right).$$
2. Optimality cuts: Inequalities that are violated by an extreme point of $\mathcal{P}$ of the form described in C2 (and possibly special cases of C1), but are valid for
$$\text{conv}\left(\left\{(x, y) \in \mathcal{F} \;\middle|\; cx + d^1 y < U\right\}\right).$$
3. Projected optimality cuts: Inequalities that are violated by the set of bilevel feasible solutions described in C3, but are valid for
$$\text{conv}\left(\left\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; x \in \mathcal{F}_x, cx + \Xi(x) < U\right\}\right).$$

Note that the presented classification is just to provide a rough idea about different classes of valid inequalities and the classes are not entirely distinct.

3.1 Feasibility Cuts

This set includes all inequalities valid for $\mathcal{S}$, the feasible region of the MILP relaxation

$$\min_{(x,y)\in\mathcal{S}} cx + d^1 y.$$

These are the very same cuts that are used in solving MILPs and include all general families, as well as any specialized MILP cuts valid for set $\mathcal{S}$ with particular structure. Since `MibS` utilizes the COIN-OR Cut Generation Library (CGL) [Cgl], all available separation routines in this package can also be employed in `MibS`. Most of these cuts are themselves derived based on the principles described earlier in Section 2.4 and fall into those general categories described there. The disjunctions utilized are typically split disjunctions, often those involving a single variable.

3.2 Optimality Cuts

Whereas the feasibility cuts are meant to restore the strength lost due to relaxation of integrality conditions, i.e., enforce the requirement that $(x, y) \in \mathcal{S}$, the optimality cuts are meant to approximately enforce the optimality condition for the second-level problem. It is this optimality condition that is relaxed (in addition to the integrality conditions) in order to obtain a tractable bounding problem. Optimality cuts are so named exactly because they approximate this non-linear inequality with linear inequalities. Note that because ϕ is non-convex and non-concave in general, this constraint cannot be approximated exactly by linear inequalities (all discussed optimality cuts in this section are linear). When combined with branching in a branch-and-cut algorithm, however, we can restrict the feasible region to areas in which this function is linear.

In the remainder of the section, we describe the optimality cuts implemented in `MibS`.

3.2.1 Integer No-Good Cut

Assumptions.

- $r_1 = n_1$ and $r_2 = n_2$.
- Vectors b^1 and b^2 and all matrices A^1, A^2, G^1 and G^2 are integer.

Theorem 4 ([DeNegre and Ralphs, 2009])**.** *Let $(\hat{x}, \hat{y}) \in \mathcal{S} \setminus \mathcal{F}$ be the optimal solution of (LR) and H_1 and H_2 denote the set of indices of the first- and second-level constraints, respectively, binding at $(\hat{x}, \hat{y})$. Then, under the stated assumptions, we have*

$$\alpha^x x + \alpha^y y \geq \beta \quad \forall (x, y) \in \mathcal{F},$$

where

$$\alpha^x = \sum_{i \in H_1} a_i^1 + \sum_{i \in H_2} a_i^2, \alpha^y = \sum_{i \in H_1} g_i^1 + \sum_{i \in H_2} g_i^2, \beta = \sum_{i \in H_1} b_i^1 + \sum_{i \in H_2} b_i^2 + 1$$

and $a_i^1, a_i^2, g_i^1, g_i^2$, b_i^1 and b_i^2 represent the i^{th} rows of A^1, A^2, G^1, G^2, b^1 and b^2, respectively. Furthermore, we have

$$\alpha^x \hat{x} + \alpha^y \hat{y} = \beta - 1,$$

so the inequality is violated by $(\hat{x}, \hat{y})$.

Proof. Let $(\alpha^x, \alpha^y, \beta) \in \mathbb{Z}^{n_1+n_2+1}$ be such that

1. $\{(x, y) \in \mathcal{P} \mid \alpha^x x + \alpha^y y = \beta - 1\} = \{(\hat{x}, \hat{y})\}$ and
2. $\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \mid \alpha^x x + \alpha^y y \geq \beta - 1\} \supseteq \mathcal{P}$.

We have that $\alpha^x x + \alpha^y y \in \mathbb{Z}$ for all $(x, y) \in X \times Y$. Furthermore,

$$\alpha^x x + \alpha^y y > \beta - 1 \; \forall (x, y) \in (\mathcal{P} \setminus \{(\hat{x}, \hat{y})\}) \supseteq \mathcal{F}.$$

It follows that $\alpha^x x + \alpha^y y \geq \beta$ for all $(x, y) \in \mathcal{F}$ and that $\alpha^x \hat{x} + \alpha^y \hat{y} < \beta$. Then, we need only observe that $(\alpha^x, \alpha^y, \beta)$ as chosen in the theorem satisfy the conditions 1 and 2 above. □

Illustrative Example. Figure 2 shows the generated integer no-good cut for removing the bilevel infeasible extreme point $(2, 4)$ of $\mathcal{P}$ in the example shown in Figure 1. The cut is obtained in two steps, as described above. We first sum the first two inequalities in the formulation (which are the ones binding at $(2, 4)$) to obtain the inequality $-2x + 3y \leq 8$, valid for $\mathcal{P}$. We then subtract one from the right-hand side to obtain the final cut $-2x + 3y \leq 7$. As one can observe, this cut does not remove any integer points from $\mathcal{P}$ except $(2, 4)$.

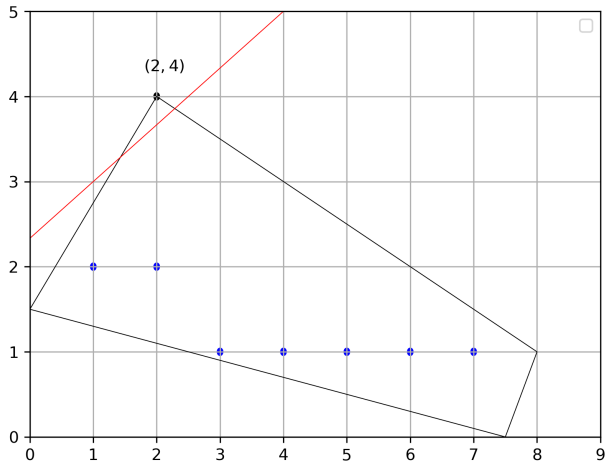


Figure 2: The generated integer no-good cut for the example shown in Figure 1

Discussion. This cut is a valid inequality for $\mathcal{F}$ with respect to the split disjunction (note that α^x, α^y and β are integer)

$$X_1 = \left\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \alpha^x x + \alpha^y y \leq \beta - 1\right\} \qquad X_2 = \left\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \alpha^x x + \alpha^y y \geq \beta\right\},$$

and since $\mathcal{F} \cap X_1 = \emptyset$, it is a generalized Chvátal inequality. The utilized split disjunction is defined by considering a combination of constraints of $\mathcal{P}$ binding at $(\hat{x}, \hat{y})$, as described earlier. The combination is guaranteed to yield a split disjunction because of our assumption that the constraint matrix is integral.

Application. This cut can be applied to remove the optimal solution $(\hat{x}, \hat{y}) \in \mathcal{S} \setminus \mathcal{F}$ of (LR) under the above assumptions, but it does not eliminate any other bilevel infeasible members of $\mathcal{S}$. It is easy to generate such cut because all required information for its generation can be obtained from the optimal tableau obtained when solving (LR).

3.2.2 Generalized Chvátal Inequality

Assumptions. None

Theorem 5. *Let $(\hat{x}, \hat{y}) \in \mathcal{S} \setminus \mathcal{F}$ be the optimal solution of (LR) and H_1 and H_2 denote the set of indices of the first- and second-level constraints, respectively, binding at $(\hat{x}, \hat{y})$. Let $u \in \mathbb{Q}^{m_1+m_2}$*

be such that $u_i = 0$ *for* $i \notin H_1 \cup H_2$, $u_i > 0$ *for* $i \in H_1 \cup H_2$, *and such that* $(\alpha^x, \alpha^y, \beta)$ *is a split disjunction where*

- $\alpha^x = u(A_1 + A_2)$,
- $\alpha^y = u(G_1 + G_2)$, *and*
- $\beta = u(b^1 + b_2) + 1$.

Then

$$\alpha^x x + \alpha^y y \geq \beta \quad \forall (x, y) \in \mathcal{F},$$

Furthermore, we have

$$\alpha^x \hat{x} + \alpha^y \hat{y} = \beta - 1,$$

so the inequality is violated by $(\hat{x}, \hat{y})$.

Proof. The inequality $(\alpha^x, \alpha^y, \beta - 1)$ is valid for $\mathcal{P}$, since it is derived as a positive combination of inequalities valid for $\mathcal{P}$. Furthermore, $(\alpha^x, \alpha^y, \beta)$ is a split disjunction for which $X_1 \cap \mathcal{P} = (\hat{x}, \hat{y})$, since there is a positive weight on all inequalities in $\mathcal{P}$ that are binding at $(\alpha^x, \alpha^y, \beta)$. Hence, $\mathcal{X}_1 \cap \mathcal{F} = \emptyset$ and $\alpha^x, \alpha^y, \beta)$ is valid for $\mathcal{F}$. □

Discussion. This cut is a form of the generalized Chvátal inequality introduced in Definition 10. In the stated form, it is not obvious how to generate the weight vector u, but the integer no-good inequality already introduced provides one possibility. Another possibility is discussed next.

Application. A potential way to generate such cuts systematically would be through the use of an auxiliary optimization problem similar to the cut-generating LP used in the case of lift-and-project for MILPs [Balas et al., 1993b], as follows.

$$\begin{aligned} \min_{u \in \mathbb{Q}^{m_1+m_2}} \quad & \alpha^x \hat{x} + \alpha^y \hat{y} - \beta \\ & \alpha^x = u(A^1_{H_1} + A^2_{H_2}) \\ & \alpha^y = u(G^1_{H_1} + G^2_{H_2}) \\ & \beta = u(b^1 + b^2) \\ & u \geq \epsilon (> 0) \\ & \alpha^x \in \mathbb{Z}^{r_1} \times \{0\}^{n_1 - r_1} \\ & \alpha^y \in \mathbb{Z}^{r_2} \times \{0\}^{n_2 - r_2} \end{aligned}$$

As written, the above problem is unbounded, so some kind of normalization would also be needed, but the appropriate normalization would probably be problem-specific.

3.2.3 Increasing Objective Cut

Assumptions.

- $x_L \subseteq \mathbb{B}^L$.
- $A^2 \leq 0$.

Theorem 6 ([DeNegre, 2011]). *Let $\hat{x} \in X$ and $\hat{y} \in \mathcal{R}(\hat{x})$. Then, under the above assumptions, we have*

$$d^2y \leq d^2\hat{y} + M\left(\sum_{i \in L:\hat{x}_i=0} x_i\right) \quad \forall (x,y) \in \mathcal{F}, \tag{8}$$

where $M \geq \max\{d^2y \mid (x,y) \in \mathcal{F}\} - d^2\hat{y}$. Furthermore, this inequality is violated by all $(x,y) \in \mathcal{S}\setminus\mathcal{F}$ with $x_i = 0$ for $\{i \in L \mid \hat{x}_i = 0\}$.

Discussion. This cut is a disjunctive inequality for $\mathcal{F}$ with respect to the valid disjunction defined by

$$X_1 = \left\{(x,y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \sum_{i \in L:\hat{x}_i=0} x_i = 0, d^2y \leq d^2\hat{y}\right\} \text{ and}$$

$$X_2 = \left\{(x,y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \sum_{i \in L:\hat{x}_i=0} x_i \geq 1, d^2y \leq d^2\hat{y} + M\left(\sum_{i \in L:\hat{x}_i=0} x_i\right)\right\}.$$

The validity of this disjunction for $\mathcal{F}$ arises from

$$\phi(b^2 - A^2x) \leq \phi(b^2 - A^2\hat{x}) \; \forall x \in X \text{ such that } \hat{y} \in \mathcal{P}_2(x).$$

Application. Under the desired assumptions, the optimal solution $(\hat{x},\hat{y}) \in \mathcal{S} \setminus \mathcal{F}$ of (LR) can be removed by generating the inequality (8) with respect to $\hat{x}$ and $\hat{y} \in \mathcal{R}(\hat{x})$. Moreover, this cut removes all bilevel infeasible solutions $(x,y) \in \mathcal{S}$ with $x_i = 0$ for $\{i \in L \mid \hat{x}_i = 0\}$. Note that in addition to the increasing objective cut, there may be stronger disjunctive inequalities, which can be generated with respect to the same valid disjunction.

3.2.4 Benders Cut

Assumptions.

- $x_L \subseteq \mathbb{B}^L$.
- Corresponding to each linking variable x_i, there exists y_i so that $x_i = 1$ results $y_i = 0$ and this is the only restriction from the second-level constraints in which the linking variables participate.
- The second-level variables coefficients are not greater than 0 in the second-level constraints in which the linking variables do not participate.

Theorem 7. *Let $(\hat{x}, \hat{y}) \in \mathcal{F}$. Then, under the above assumptions, we have*

$$d^2 y \leq \sum_{i \notin L} d_i^2 \hat{y}_i + \sum_{i \in L} d_i^2 \hat{y}_i (1 - x_i) \quad \forall (x, y) \in \mathcal{F}. \tag{9}$$

Furthermore, this inequality is violated by all $(x, y) \in \mathcal{S} \setminus \mathcal{F}$ with $x_L = \hat{x}_L$.

Proof. Let $(\tilde{x}, \tilde{y}) \in \mathcal{F}$ and $y' \in Y$ such that $y'_i = \hat{y}_i$ if $i \notin L$ and

$$y'_i = \hat{y}_i (1 - \tilde{x}_i) = \begin{cases} 0 & \text{if } i \in L \text{ and } \tilde{x}_i = 1, \\ \hat{y}_i & \text{if } i \in L \text{ and } \tilde{x}_i = 0. \end{cases}$$

By the definition of y', we have $y' \leq \hat{y}$, which results that y' satisfies the second-level constraints in which the linking variables are not present. It also satisfies the other constraints with respect to $\tilde{x}$ because for $i \in L$, $y'_i = 0$ if $\tilde{x}_i = 1$. Hence, $y' \in \mathcal{P}_2(\tilde{x}) \cap Y$ and since $\tilde{y} \in \mathcal{R}(\tilde{x})$, we have

$$d^2 \tilde{y} \leq d^2 y' = \sum_{i \notin L} d_i^2 \hat{y}_i + \sum_{i \in L} d_i^2 \hat{y}_i (1 - \tilde{x}_i).$$

This follows that the inequality (9) is valid for all bilevel feasible solutions. Moreover, since $(\hat{x}, \hat{y}) \in \mathcal{F}$, we have $\hat{y}_i = 0$ for all $\{i \in L \mid \hat{x}_i = 1\}$. It follows that

$$d^2 \hat{y} = \sum_{i \notin L} d_i^2 \hat{y}_i + \sum_{i \in L} d_i^2 \hat{y}_i (1 - \hat{x}_i). \tag{10}$$

Furthermore, for $(x, y) \in \mathcal{S} \setminus \mathcal{F}$ with $x_L = \hat{x}_L$, we have

$$d^2 y > d^2 \hat{y}. \tag{11}$$

(10) and (11) result that the inequality (9) is violated by all $(x, y) \in \mathcal{S} \setminus \mathcal{F}$ with $x_L = \hat{x}_L$. □

Discussion. Caprara et al. [2016] proposed this cut for the knapsack interdiction problems, but the benders cut can be employed for more general problems as was illustrated in Theorem 7. This cut is a disjunctive inequality for $\mathcal{F}$ with respect to the valid disjunction

$$X_1 = \left\{ (x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \sum_{i \in L : \hat{x}_i = 0} x_i + \sum_{i \in L : \hat{x}_i = 1} (1 - x_i) = 0, d^2 y \leq d^2 \hat{y} \right\}$$

$$X_2 = \left\{ (x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \sum_{i \in L : \hat{x}_i = 0} x_i + \sum_{i \in L : \hat{x}_i = 1} (1 - x_i) \geq 1, d^2 y \leq \sum_{i \notin L} d_i^2 \hat{y}_i + \sum_{i \in L} d_i^2 \hat{y}_i (1 - x_i) \right\}.$$

Application. Under the stated assumptions, the Benders cut can be exploited to remove the optimal solution $(\hat{x}, \hat{y}) \in \mathcal{S} \setminus \mathcal{F}$ of (LR) with respect to the bilevel feasible solution $(\hat{x}, \hat{y})$, where $\hat{y} \in \mathcal{R}(\hat{x})$. Moreover, this cut removes all bilevel infeasible solutions $(x, y) \in \mathcal{S}$, with $x_L = \hat{x}_L$.

3.2.5 Intersection Cut

Assumptions.

- $A^2x + G^2y - b^2 \in \mathbb{Z}^{m_2}$ for all $(x, y) \in \mathcal{S}$.
- $d^2 \in \mathbb{Z}^{n_2}$.

Theorem 8 ([Fischetti et al., 2018], [Fischetti et al., 2017]). *Let $(\hat{x}, \hat{y}) \notin \mathcal{F}$ be the optimal solution of (LR) and $y^* \in Y$ satisfy these conditions:*

- $d^2y^* < d^2\hat{y}$.
- $G^2y^* \geq b^2 - A^2\hat{x}$.

Then, under the stated assumptions, we have

$$\alpha^x x + \alpha^y y \geq \beta \quad \forall (x, y) \in \mathcal{F}, \tag{12}$$

where the inequality (12) is the intersection cut associated with the sets $\mathcal{V}(\hat{x}, \hat{y})$ and

$$\mathcal{C} = \left\{(x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; d^2y \geq d^2y^*, G^2y^* \geq b^2 - A^2x - 1\right\}, \tag{13}$$

Furthermore, the inequality (12) is violated by $(\hat{x}, \hat{y})$.

Two distinct types of cuts can be derived from the above formula, obtained by computing y^* in two different ways.

Intersection Cut I. This type is obtained simply by selecting any $y^* \in \mathcal{R}(\hat{x})$. Note that in this case, the assumption $d^2 \in \mathbb{Z}^{m_2}$ is not required.

Intersection Cut II. In this type of intersection cut, y^* is taken to be an optimal solution of the following MILP.

$$\begin{aligned} y^* \in \text{argmin} & \sum_{i=1}^{m_2} w_i \\ & d^2y \leq d^2\hat{y} - 1 \\ & G^2y + (A^2\hat{x} - L)w \geq b^2 - L \\ & y \in Y \\ & w \in \{0, 1\}^{m_2}, \end{aligned} \tag{14}$$

where $L_i = \sum_{j=1}^{n_1} \min\{A^2_{ij} l_{x_j}, A^2_{ij} u_{x_j}\}$ for $i = 1, ..., m_2$ and A^2_{ij} represents the element of i^{th} row and j^{th} column of A^2.

Note that due to constraint (14), $w^*_i = 0$ (w^* represents the optimal value of w) guarantees that $g^2_i y^* \geq b^2_i - a^2_i x$ for all bilevel feasible solutions. Hence, this inequality can be removed from the definition of set $\mathcal{C}$ in (13) in this case. By utilizing such property, the defined set $\mathcal{C}$ for intersection cut type II is hopefully larger than the one for type I and may provide a stronger cut. However, it should also be noted that finding y^* for type II cuts requires solving an MILP, while in intersection cut I, no additional effort is needed.

Illustrative Example. Figure 3 shows the generated intersection cut (the red line) for removing the extreme point (2,4) of $\mathcal{P}$ in the example shown in Figure 1. The blue cone and the green dotted region show $\mathcal{V}(2,4))$ and set $\mathcal{C}$, respectively. Note that in this case, the generated cuts and sets $\mathcal{C}$ are the same for both described types of intersection cut.

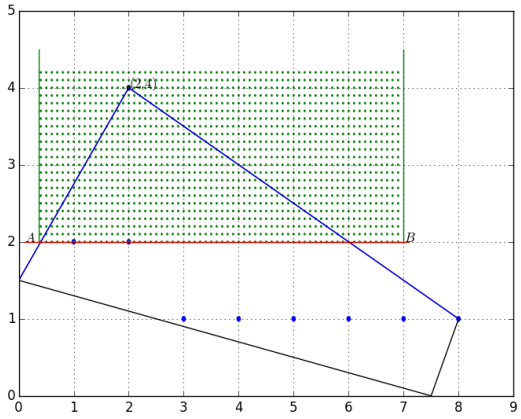


Figure 3: The generated intersection cut for the example shown in Figure 1

Discussion. As one can observe in Theorem 8 (and also Theorems 9 and 10), the intersection cuts can also be exploited in the context of MIBLPs.

Application. In case of existing $\hat{y}$ with the desired conditions (such $\hat{y}$ exists certainly when $(\hat{x},\hat{y}) \in \mathcal{S}$), this cut can be applied to remove the optimal solution $(\hat{x},\hat{y}) \notin \mathcal{F}$ of (LR) under the above assumptions. The generated cut is valid for the set $\mathcal{V}(\hat{x},\hat{y}) \setminus \text{int}(\mathcal{C})$ (note that $\mathcal{F} \subseteq \mathcal{V}(\hat{x},\hat{y}) \setminus \text{int}(\mathcal{C})$), however, it does not remove all points belong to the set $\mathcal{V}(\hat{x},\hat{y}) \cap \text{int}(\mathcal{C})$ necessarily.

3.2.6 Watermelon Intersection Cut

Assumptions.

- $A^2x + G^2y - b^2 \in \mathbb{Z}^{m_2}$ for all $(x,y) \in \mathcal{S}$.
- $d^2 \in \mathbb{Z}^{n_2}$.

Theorem 9 ([Fischetti et al., 2017]). *Let $(\hat{x},\hat{y}) \notin \mathcal{F}$ be the optimal solution of (LR) and $\Delta\hat{y} \in \mathbb{Z}^{r_2} \times \mathbb{R}^{n_2-r_2}$ satisfy these conditions:*

- $d^2\Delta\hat{y} < 0$.
- $G^2(\hat{y} + \Delta\hat{y}) \geq b^2 - A^2\hat{x}$.
- $\hat{y} + \Delta\hat{y} \geq 0$.

Then, under the stated assumptions, we have

$$\alpha^x x + \alpha^y y \geq \beta \quad \forall (x,y) \in \mathcal{F}, \tag{15}$$

where the inequality (15) *is the intersection cut generated associated with the sets* $\mathcal{V}(\hat{x},\hat{y})$ *and*

$$\mathcal{C} = \left\{ (x,y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; G^2(y+\Delta\hat{y}) \geq b^2 - A^2 x - 1, y + \Delta\hat{y} \geq -1 \right\}. \tag{16}$$

Furthermore, the inequality (15) *is violated by* $(\hat{x},\hat{y})$.

Generating a watermelon intersection cut requires finding $\Delta\hat{y} \in \mathbb{Z}^{r_2} \times \mathbb{R}^{n_2 - r_2}$ described in Theorem 9. Such vector can be achieved by solving the MILP

$$\begin{aligned}
\Delta\hat{y} \in \text{argmax} & \sum_{i=1}^{m_2} w_i + \sum_{i=1}^{n_2} v_i \\
& d^2 \Delta y \leq -1 \\
& G^2 \Delta y \geq b^2 - A^2\hat{x} - G^2\hat{y} \\
& \Delta y \geq -\hat{y} \\
& G^2 \Delta y \geq w \\
& \Delta y \geq v \\
& \Delta y \in \mathbb{Z}^{r_2} \times \mathbb{R}^{n_2 - r_2} \\
& w \leq 0 \\
& v \leq 0.
\end{aligned}$$

In the same vein as the intersection cut II, with the aim of finding a larger convex set, the optimal values of the vectors w and v can be employed to drop some of the inequalities in the definition of set (16). $w_i^* = 0$ ($v_i^* = 0$) means that $g_i^2(y + \Delta\hat{y}) \geq b_i^2 - a_i^2 x$ ($y_i + \Delta\hat{y}_i \geq 0$) for all bilevel feasible solutions, so this inequality can be removed from the definition of set (16).

Illustrative Example. Figure 4 shows the generated watermelon intersection cut (the red line) for removing the extreme point (2,4) of $\mathcal{P}$ in the example shown in Figure 1. The blue cone and the green dotted region show $\mathcal{V}(2,4)$ and set $\mathcal{C}$, respectively.

Discussion. As one can observe in Figures 3 and 4 (and also Figure 5), none of the sets $\mathcal{C}$ is not the subset of another, so we cannot say that one type of intersection cuts always dominates the others.

Application. In a similar way as the intersection cut, under the stated assumptions and also existing the desired $\Delta\hat{y}$, this cut can be utilized to separate a subset of $\mathcal{V}(\hat{x},\hat{y}) \cap \text{int}(\mathcal{C})$ which definitely includes the optimal solution $(\hat{x},\hat{y}) \notin \mathcal{F}$ of (LR) (the desired $\Delta\hat{y}$ exists certainly when $(\hat{x},\hat{y}) \in \mathcal{S}$)).

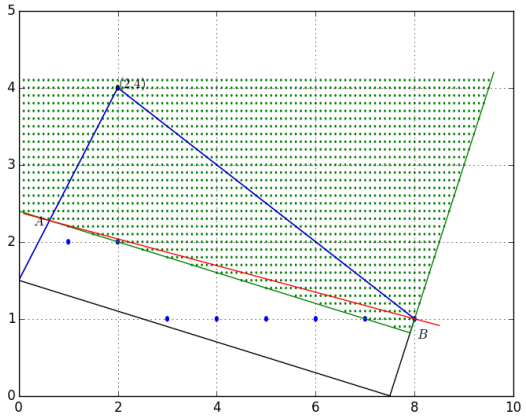


Figure 4: The generated watermelon intersection cut for the example shown in Figure 1

3.2.7 Hypercube Intersection Cut

Assumptions.

- $x_L \subseteq \mathbb{Z}^L$.

Theorem 10 ([Fischetti et al., 2017])**.** *Let $(\hat{x}, \hat{y}) \in \mathcal{P}$ be the optimal solution of (LR) with $\hat{x}_L \in \mathbb{Z}^L$. Then, under the stated assumption, we have*

$$\alpha^x x + \alpha^y y \geq \beta \quad \forall (x, y) \in \mathcal{F} \text{ such that } x_L \neq \hat{x}_L, \tag{17}$$

where the inequality (17) is the intersection cut generated associated with the sets $\mathcal{V}(\hat{x}, \hat{y})$ and

$$\mathcal{C} = \left\{ (x, y) \in \mathbb{R}^{n_1 \times n_2} \;\middle|\; \hat{x}_i - 1 \leq x_i \leq \hat{x}_i + 1 \quad \forall i \in L \right\}.$$

Furthermore, the inequality (17) is violated by $(\hat{x}, \hat{y})$.

Illustrative Example. Figure 5 shows the generated hypercube intersection cut (the red line) for removing the extreme point (2,4) of $\mathcal{P}$ in the example shown in Figure 1. The blue cone and the green dotted region show $\mathcal{V}(2, 4)$ and set $\mathcal{C}$, respectively.

Discussion. Since this cut eliminates the solution to (LR), even when it is bilevel feasible, the problem (*UB*) must first be solved with $\gamma = \hat{x}_L$ prior to generating such cut. By doing this, all possible removed bilevel feasible solutions will be non-improving.

Application. Under the above assumption, the optimal solution $(\hat{x}, \hat{y}) \notin \mathcal{F}$ of (LR) with $\hat{x}_L \in \mathbb{Z}^L$ can be removed by generating the hypercube intersection cut with respect to this point. The generated cut may also be violated by a subset of set $\mathcal{F}$ with the same linking part as $\hat{x}$, but (as shown in Figure 5) it does not necessarily remove all $(x, y) \in \mathcal{P}$ with $x_L = \hat{x}_L$.

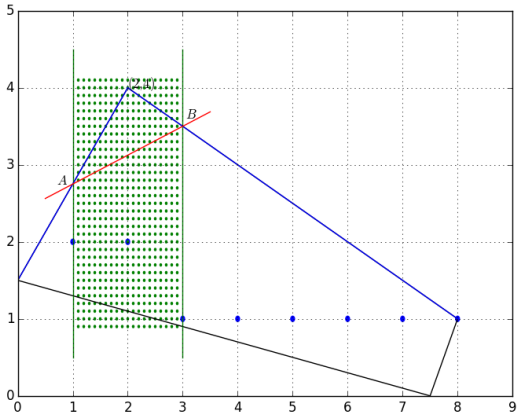


Figure 5: The generated hypercube intersection cut for the example shown in Figure 1

3.3 Projected Optimality Cuts

The only projected optimality cut in `MibS` is the generalized no-good cut described in this section.

3.3.1 Generalized No-good Cut

Assumptions.

- $x_L \subseteq \mathbb{B}^L$.

Theorem 11. *Let $\gamma \in \mathbb{B}^L$. Then, under the desired assumption, we have*

$$\sum_{i \in L:\gamma_i=0} x_i + \sum_{i \in L:\gamma_i=1} (1 - x_i) \geq 1 \quad \forall (x, y) \in \mathcal{F} \text{ such that } x_L \neq \gamma. \tag{18}$$

Furthermore, the inequality (18) is violated by all $(x, y) \in \mathcal{P}$ with $x_L = \gamma$.

Proof. When $x_L = \gamma$, we have $\sum_{i \in L:\gamma_i=0} x_i = 0$ and $\sum_{i \in L:\gamma_i=1}(1 - x_i) = 0$, which follows that

$$x_L = \gamma \Rightarrow \sum_{i \in L:\gamma_i=0} x_i + \sum_{i \in L:\gamma_i=1} (1 - x_i) = 0. \tag{19}$$

Furthermore, when $x_L \neq \gamma$, at least one of the following happens

- $\exists i \in L$ such that $\gamma_i = 0$ and $x_i = 1 \Rightarrow \sum_{i \in L:\gamma_i=0} x_i \geq 1$.
- $\exists i \in L$ such that $\gamma_i = 1$ and $x_i = 0 \Rightarrow \sum_{i \in L:\gamma_i=1}(1 - x_i) \geq 1$.

Hence, we have

$$x_L \neq \gamma \Rightarrow \sum_{i \in L:\gamma_i=0} x_i + \sum_{i \in L:\gamma_i=1} (1 - x_i) \geq 1. \tag{20}$$

The result follows from (19) and (20). □

Discussion. The generalized no-good cut is a generalization of the no-good cut suggested by DeNegre [2011]. His proposed cut is employed to remove all solutions with the same first-level values, while the new version strengthens the former one since it separates the solutions with the same linking part. Moreover, since the no-good cut is valid only for the problems in which all first-level variables are binary, the assumption of the generalized one is less strict.

As with the hypercube intersection cut, the corresponding problem (UB) should be solved prior to generating a generalized no-good cut. In contrast with the hypercube intersection cuts, it is guaranteed that the generalized no-good cut eliminates all solutions with the target linking values. Note, however, that the hypercube intersection cut is applicable for more general MIBLPs.

Application : Under the above assumption, after solving the problem (UB) with $\gamma = \hat{x}_L$, the generalized no-good cut can be applied to remover the optimal solution $(\hat{x}, \hat{y}) \in \mathcal{P} \setminus \mathcal{F}$ of (LR) with $\hat{x}_L \in \mathbb{B}^L$ and all other solutions $(x, y) \in \mathcal{P}$ with $x_L = \hat{x}_L$. At a more general level, this cut can be generated whenever problem (UB) is solved to avoid visiting the solutions with the same linking part again.

4 Strong Valid Inequalities

As in the MILP case, there is a limit to the strength of inequalities that can be obtained through procedures such as the ones describe in the last section. This is particularly true in the case of MIBLP. In the MILP case, there exist methods for strengthening known valid inequalities by solving auxiliary optimization problems. Specifically, lifting is a procedure for computing "optimal" coefficients that can be used to strengthen a given base inequality. Lifting can be applied directly to inequalities valid for MIBLPs in the obvious way. Here, we suggest a related methodology to strengthen the right-hand side of a given inequality, while leaving the left-hand side coefficients unmodified.

4.1 General Principle

The method we propose is based on the straightforward idea that a valid inequality $(\alpha^x, \alpha^y, \beta)$ can be strengthened if we are able to prove that β can be replaced with β^* for some $\beta^* > \beta$. The methodology for generating such a proof is general and can be applied broadly in many settings beyond the narrow one described here. In particular, it can be applied to the solution of traditional single-level MILPs. Figure 6 shows how decreasing the right-hand side (recall that this cut was given in "$\leq$" form, so the strengthening is a decrease in the right-hand side) of the integer no-good cut described in Figure 2 results in a stronger valid inequality. The strengthening is accomplished by computing the right-hand side that ensures the associated hyperplane supports $\mathcal{F}$ (the new right-hand side is 4 in this case). This computation is obviously an optimization problem over $\mathcal{F}$ and thus intractable in general, but in the remainder of this section and the next, we discuss ways in which the computation can be carried out approximately to improve tractability.

In the remainder of this subsection, we consider a general closed feasible set $\mathcal{G} \subset \mathbb{R}^n$ and then apply the principles discussed specifically to the MIBLP case in subsection 4.2. As we have already

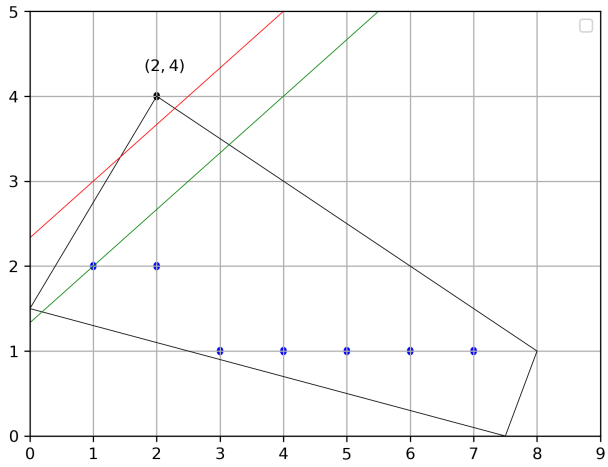


Figure 6: Strengthening the integer no-good cut

indicated, the largest β for which $(\alpha, \beta) \in \mathbb{Q}^n \times \mathbb{Q}$ is a valid inequality for $\mathcal{G}$ is the unique maximum β such that the hyperplane $\{x \in \mathbb{R}^n \mid \alpha x = \beta\}$ supports $\text{conv}(\mathcal{G})$ (i.e, has non-empty intersection with $\mathcal{G}$). For convenience, we define a function, which we refer to as an *inverse value function*, that returns this largest value of the right-hand side as a function of the left-hand side vector, as follows.

Definition 12. *The* inverse value function $\Gamma_{\mathcal{G}}$ *is the function that returns the largest value of* $\beta \in \mathbb{Q}$ *such that* (α, β) *is valid for* $\text{conv}(\mathcal{G})$*, given* $\alpha \in \mathbb{Q}^n$*.*

It is straightforward to observe that $\Gamma_{\mathcal{G}}(\alpha)$ can be computed by solving an optimization problem.

Theorem 12. *For any* $\mathcal{G} \subseteq \mathbb{Q}^n$ *and* $\alpha \in \mathbb{Q}^n$*,*

$$\Gamma_{\mathcal{G}}(\alpha) = \min_{x \in \mathcal{G}} \alpha x = \min_{x \in \text{conv}(\mathcal{G})} \alpha x. \tag{21}$$

In other words, $\Gamma_{\mathcal{G}}$ can also be interpreted as a function that returns the optimal value of an optimization problem as a function of the objective vector. The origin of the name *inverse value function* is from the fact that this function plays a role here similar to the function that is more typically called a *value function*, which is a function of the right-hand side rather than the objective function [Hassanzadeh and Ralphs, 2014]. The word "inverse" is to indicate the connection to the inverse optimization problem, which is closely related to the problem of computing $\Gamma_{\mathcal{G}}^{-1}(\beta)$ [Bulut and Ralphs, 2015].

We have the following corollary that states the connection between $\Gamma_{\mathcal{G}}$ and valid inequalities more directly.

Corollary 2. *An inequality* $(\alpha, \beta) \in \mathbb{Q}^n \times \mathbb{Q}$ *is valid for* $\mathcal{G}$ *if and only if* $\beta \leq \Gamma_{\mathcal{G}}(\alpha)$*. The associated face supports* $\mathcal{G}$ *if and only if* $\beta = \Gamma_{\mathcal{G}}(\alpha)$*.*

Note that we could replace $\mathcal{G}$ with a subset of $\mathcal{G}$ containing only improving solutions in order to obtain a similar condition for improving valid inequalities.

Corollary 2 tells us that the strongest possible right-hand side would be obtained by computing the exact value of $\Gamma_{\mathcal{G}}(\alpha)$, but that we may instead use any *lower bound* to ensure validity of the inequality. There are two obvious strategies to calculating such a lower bound. First, we can replace $\mathcal{G}$ with a larger set (essentially relaxing the optimization problem). This is a common strategy used by many cut generation procedures. Second, we can compute a lower bound by executing a time- or node-limited branch-and-bound procedure. The latter is the approach we take here. As an aside, this latter procedure can technically be viewed as an implementation of the former, since the union of the feasible regions of the subproblems associated with the leaf nodes of the branch-and-bound tree constitute a set containing $\mathcal{G}$ over which we are optimizing to obtain the dual bound associated with the branch-and-bound tree. The proposed strategy creates an obvious tradeoff between the resource limit imposed and the strength of the resulting inequality. The fine-tuning of this tradeoff is crucial to the effectiveness of the algorithm and our procedure for doing this in the case of MIBLP is described in Section 6.

A question we have not addressed yet is exactly how the left-hand side vectors that are inputs to this strengthening procedure are generated. This is a largely empirical question that we discuss in the specific case of MIBLP in the next subsection.

4.2 Strong Valid Inequalities for MIBLP

In the specific case of an MIBLP that is itself solved by a branch-and-bound algorithm, further improvements to the tractability of the procedure suggested above can be made by generating inequalities that are only locally valid with respect to some subproblem in the branch-and-bound tree. The problem of generating an improved right-hand side should be easier in such cases, since the optimization is then over a reduced feasible region. With respect to node t and $(\alpha^x, \alpha^y) \in \mathbb{Q}^{n_1+n_2}$, the function $\Gamma_{\mathcal{F}^t}(\alpha^x, \alpha^y)$ is the inverse value function function that generates strengthened valid inequalities. Theorem 13 captures the use of such a function to generate a strong valid inequality for node t, given a left-hand side vector.

Theorem 13. *Let $(\alpha^x, \alpha^y) \in \mathbb{Q}^{n_1+n_2}$ be given and let $\beta \in \mathbb{R}$ be the lower bound resulting from a resource-limited branch-and-bound procedure for attempting to solve*

$$\min_{(x,y)\in\mathcal{F}^t} \alpha^x x + \alpha^y y. \tag{22}$$

Then $(\alpha^x, \alpha^y, \beta)$ is valid for $\mathcal{F}^t$. The associated face supports $\mathcal{F}^t$ if and only if β is the optimal value of (22).

The empirical question of how the left-hand side vectors that are the input to this strengthening procedure are obtained is a crucial one. In our experiments, we have so far used left-hand sides from two different sources, which are described next. In Section 4.2.1, we describe how we employ the described idea for generating *strengthened integer no-good cuts*, which are exactly as one would expect—first generate an integer no-good cut and then strengthen it using the procedure. In Section 4.2.2, we describe so-called *bound cuts* in which the left-hand side is fixed to d^2. The bound cuts are linear cuts that attempt to replicate some of the strength lost when relaxing the second-level optimality condition to obtain the LP relaxation that is the basis for the branch-and-cut procedure.

4.2.1 Strengthened Integer No-good Cut

The idea of the strengthened integer no-good cut is both to strengthen the typically weak integer no-good cut and to remove some of the difficulties surrounding the scaling that may be required to obtain integer coefficients, as needed for validity of the original cut.

Theorem 14. *Let $(\hat{x}, \hat{y}) \notin \mathcal{F}$ be the optimal solution of (LR) and H_1 and H_2 denote the set of indices of the first- and second-level constraints, respectively, binding at $(\hat{x}, \hat{y})$. Then we have*

$$\alpha^x x + \alpha^y y \geq \beta \quad \forall (x, y) \in \mathcal{F},$$

where

$$\alpha^x = \sum_{i \in H_1} a_i^1 + \sum_{i \in H_2} a_i^2, \alpha^y = \sum_{i \in H_1} g_i^1 + \sum_{i \in H_2} g_i^2, \beta \leq \Gamma_{\mathcal{F}}(\alpha^x, \alpha^y)$$

and a_i^1, a_i^2, g_i^1 and g_i^2 represent the i^{th} rows of A^1, A^2, G^1 and G^2, respectively.

As we have already mentioned, computing β is an optimization problem over $\mathcal{F}$ and we have a branch-and-bound procedure for solving this problem. Therefore, we can straightforwardly apply the procedure that is the basis for Theorem 13. Note that for the inequality to be violated by $(\hat{x}, \hat{y})$, we must have

$$\beta > \alpha^x \hat{x} + \alpha^y \hat{y}$$

so we need to obtain a lower bound improving upon this value in order to ensure violation. Naturally, these inequalities can also be derived with respect to individual subproblems arising in the branch-and-bound tree, but for notational simplicity, the above theorem describes only the globally valid version.

It is important to mention that using this procedure avoids the need for the restrictive assumptions of the original integer no-good cut, as described in Section 3.2.1. According to Theorem 4, this inequality requires the integrality of all first- and second-level variables, as well as that of the constraint coefficients and the right-hand side vectors. This is due to the method used for computing the right-hand side for a given left-hand side vector. None of these assumptions are needed for validity, however, when the strengthening procedure is applied, since validity is assured by direct computation of a valid right-hand side.

4.2.2 Bound Cut

It is clear that the relaxation of the optimality constraint of the second-level problem is typically the primary reason for the weakness of the LP relaxation used in the branch-and-bound procedure for solving (MIBLP). In order to tighten the relaxation, an obvious approach is thus to impose a valid upper bound on the value of the second-level objective function. Such a bound can be viewed as a (linear) relaxation of the original optimality constraint of the second-level problem. The strongest version of such a cut, which we refer to as a *bound inequality* at node t, is defined as

$$-d^2 y \geq \Gamma_{\mathcal{F}^t}(0, -d^2).$$

Obviously, computing $\Gamma_{\mathcal{F}^t}(0, -d^2)$ is a bilevel problem in itself (though simpler than the original one). Nevertheless, using a resource-limited branch-and-bound is a viable approach to improving tractability and obtaining a valid inequality. In the next section, we discuss a further improvement in which we view the inequality as being *parametric* and are able to compute strengthened bounds for subproblems deeper in the branch-and-bound tree based on computations performed in ancestors using a technique that essentially amounts sensitivity analysis.

5 Parametric Inequalities

In this section, we describe a procedure for automatically and inexpensively strengthening the inequalities obtained by the procedure in the previous section. Specifically we focus on deriving new inequalities that are locally valid for the subproblems obtained by branching. As previously, the technique has broad application, but we describe the general framework using the notation from (MIBLP) to avoid introducing too much additional notation.

5.1 General Principle

A *parametric cut* is a cut whose coefficients and/or right-hand side are functions of the input data. The subadditive inequalities arising in integer programming are a classic example [Nemhauser and Wolsey, 1988]. Some known classes of valid inequalities, such as Gomory cuts, also have a parametric form. Here, we describe a general notion of parametric valid inequality in which the left-hand side remains constant but the right-hand side is expressed as a parametric function of (some of) the input data. The parameterization we consider is of the variable bounds, which are the parts of the input to the subproblems that are modified as branching occurs.

As previously, let a left-hand side vector $(\alpha^x, \alpha^y) \in \mathbb{Q}^{n_1+n_2}$ be given. The goal is to generate strong valid inequalities with this left-hand side, but with the right-hand side determined by an easily computable function of the input data for different nodes in the branch-and-bound tree. The so-called *non-parametric approach* would be simply to employ Theorem 13 at each node t, beginning the optimization from scratch. However, by essentially warm-starting the optimization procedure, we can inexpensively compute an improved right-hand side after bounds on the variables have been modified due to branching. This is the so-called *parametric approach*.

The core idea is to construct a *nodal dual function* (with respect to (α^x, α^y)) that allows us to quickly obtain a bound on the optimal value of a subproblem of the form (22). The nodal dual function is thus a function that bounds the so-called value function of the parametric family of optimization problem arising as subproblems in a branch-and-bound procedure for solving (22) (for a detailed discussion of value functions in general, as they pertain to MIBLPs, see [Bolusani et al., 2020, Bolusani and Ralphs, 2020]).

Specifically, we consider functions of the variable bounds, as follows. At the risk of abusing notation, let $l_x, u_x \in \mathbb{R}^{n_1}$ and $l_y, u_y \in \mathbb{R}^{n_2}$ denote vectors of lower and upper bounds for the first- and second-level variables. We then define the nodal dual function as follows.

Definition 13. *A* nodal dual function $F^\alpha : \mathbb{R}^{2n_1+2n_2} \to \mathbb{R}$ *with respect to* (α^x, α^y) *is any function*

satisfying

$$F^\alpha(l_x, u_x, l_y, u_y) \leq \min_{(x,y)\in\mathcal{F}} \{\alpha^x x + \alpha^y y \mid l_x \leq x \leq u_x, l_y \leq y \leq u_y\} \; \forall l_x, u_x \in \mathbb{R}^{n_1}, l_y, u_y \in \mathbb{R}^{n_2}$$

Here, the superscript α indicates that the function is defined with respect to a particular fixed left-hand side vector. If F^α is a nodal dual function with respect to $(\alpha^x, \alpha^y) \in \mathbb{Q}^{n_1+n_2}$, then

$$F^\alpha(l_x^t, u_x^t, l_y^t, u_y^t) \leq \min_{(x,y)\in\mathcal{F}^t} \alpha^x x + \alpha^y y$$

Theorem 15 thus tells us that such a nodal dual function can be exploited to obtain a parametric family of inequalities valid for any given node in the branch-and-bound tree.

Theorem 15. *Let $F^\alpha : \mathbb{R}^{2n_1+2n_2} \to \mathbb{R}$ be a nodal dual function associated with $(\alpha^x, \alpha^y) \in \mathbb{Q}^{n_1+n_2}$. Then, at node t, in the branch-and-bound tree for solving the original problem* (MIBLP)*, we have*

$$\alpha^x x + \alpha^y y \geq F^\alpha(l_x^t, u_x^t, l_y^t, u_y^t) \quad \forall (x, y) \in \mathcal{F}^t.$$

In this way, we obtain a different right-hand side for each node t with respect to the same left-hand side by plugging in the set of variable bounds that define the subproblem at node t into the nodal dual function. Of course, the strength of the resulting inequality at a given node depends strongly on how well the nodal dual function approximates the true optimal values of that subproblems. We discuss how the nodal dual functions arise next.

5.2 Application

While Theorem 15 provides the main theoretical justification for the parametric approach, it does not specify how the desired nodal dual function(s) (or the left-hand sides) should be generated. To build the function associated with a given left-hand side, our approach is precisely the same as the approach for obtaining valid right-hand side values in Section 4—we first attempt to solve problem (22) using a resource-limited branch-and-bound procedure in the root node ($t = 0$) of the branch-and-bound tree for the original MIBLP and then exploit the information from the obtained branch-and-bound tree to build the nodal dual function F^α.

More specifically, let problem (22) be solved (partially or completely) for $t = 0$ by a separate branch-and-bound algorithm, such as that of `MibS` itself. Since (22) is a bilevel optimization problem itself, we use related notation to describe this problem and *its* solution procedure, but carefully distinguish objects associated with solution of (22) from objects associated with solution of the original problem (MIBLP). As such, let $(\alpha^x, \alpha^y) \in \mathbb{Q}^{n_1+n_2}$ be a given left-hand side vector and define the following sets with respect to (22) at the root node ($t = 0$) and the associated set K of leaf nodes of the resource-limited branch-and-bound procedure associated with it.

- $\mathcal{F}_\alpha^k$ is the feasible region of the bilevel subproblem associated with leaf node k;
- $\mathcal{P}_\alpha^k$ is the feasible region of the LP relaxation of leaf node k; and
- $\mathcal{S}_\alpha^k = \mathcal{P}_\alpha^k \cap (X \times Y)$.

We assume that when solving (22) at the root node ($t = 0$), no improving valid inequalities that may remove part of the bilevel feasible region are generated. This is because while we may be able to prove that addition of such inequalities are valid when the goal is only to solve the original problem to optimality, they may remove parts of the feasible region that are needed in describing the nodal dual function. For similar reasons, the nodes in set K must include nodes that would ordinarily be discarded after pruning. To emphasize this, we classify the leaf nodes constituting the set K as follows (the classification is not explicitly referenced in what follows, however).

- K_1: The set of nodes fathomed because the relaxation problem is infeasible ($\mathcal{P}^k_\alpha = \emptyset$).
- K_2 : The set of nodes fathomed because the objective value of the relaxation is greater than the current global upper bound.
- K_3 : The set of nodes whose processing was terminated because resource limits were exceeded (either time or node limits).
- K_4 : The set of nodes fathomed because the optimal solution of relaxation problem was bilevel feasible.
- K_5 : The set of nodes fathomed because the linking variables were fixed and the node was pruned after solving (UB) (with objective function $\alpha^x x + \alpha^y y$).

Theorem 16 states formally how information obtained when processing these leaf nodes can then be used to construct the nodal dual function F^α.

Theorem 16. *Let problem (22) be solved by a resource-constrained branch-and-bound for $t = 0$, as described above. Moreover, let the vectors $LB^k \in \mathbb{R}^{n_1+n_2}$ and $UB^k \in \mathbb{R}^{n_1+n_2}$ denote, respectively, the intersection of the bounds l_x, u_x, l_y, and u_y with the bounds describing the leaf node $k \in K$. Then, if we define*

$$F^\alpha(l_x, u_x, l_y, u_y) = \min_{k \in K} z^k,$$

where

$$z^k = \begin{cases} \min\left\{\alpha^x x + \alpha^y y \;\middle|\; (x,y) \in \mathcal{S}^k_\alpha, d^2 y \le \phi(b^2 - A^2 x), LB^k \le (x,y) \le UB^k\right\} & \text{if } LB^k_L = UB^k_L \\ \min\left\{\alpha^x x + \alpha^y y \;\middle|\; (x,y) \in \mathcal{P}^k_\alpha, LB^k \le (x,y) \le UB^k\right\} & \text{otherwise,} \end{cases}$$

we have that F^α is a nodal dual function.

Proof. Since none of the cuts generated while solving problem (22) with $t = 0$ remove any bilevel feasible solutions, we have that

$$\{(x,y) \in \mathcal{F} \mid l_x \le x \le u_x, l_y \le y \le u_y\} \subseteq \bigcup_{k \in K} \left\{(x,y) \in \mathcal{F}^k_\alpha \;\middle|\; LB^k \le (x,y) \le UB^k\right\},$$

which results

$$\begin{aligned} &\min\left\{\alpha^x x + \alpha^y y \mid (x,y) \in \mathcal{F}, l_x \le x \le u_x, l_y \le y \le u_y\right\} \\ &\ge \min_{k \in K}\left\{\min\left\{\alpha^x x + \alpha^y y \;\middle|\; (x,y) \in \mathcal{F}^k_\alpha, LB^k \le (x,y) \le UB^k\right\}\right\}. \end{aligned} \tag{23}$$

Moreover, we have

$$\begin{aligned}&\min\left\{\alpha^x x+\alpha^y y \;\middle|\; (x,y)\in\mathcal{F}_\alpha^k, LB^k\le(x,y)\le UB^k\right\}\\&=\min\left\{\alpha^x x+\alpha^y y \;\middle|\; (x,y)\in\mathcal{S}_\alpha^k, d^2y\le\phi(b^2-A^2x), LB^k\le(x,y)\le UB^k\right\}\text{ if } LB_L^k=UB_L^k,\end{aligned}\tag{24}$$

and

$$\begin{aligned}&\min\left\{\alpha^x x+\alpha^y y \;\middle|\; (x,y)\in\mathcal{F}_\alpha^k, LB^k\le(x,y)\le UB^k\right\}\ge\\&\min\left\{\alpha^x x+\alpha^y y \;\middle|\; (x,y)\in\mathcal{P}_\alpha^k, LB^k\le(x,y)\le UB^k\right\}\text{ for all } k\in K.\end{aligned}\tag{25}$$

The result follows from (23), (24) and (25). □

6 Computational Results

Some experiments were conducted in order to evaluate the performance of different valid inequalities for MIBLPs described in previous sections. Four different data sets were employed in our experiments as follows.

- `INTERD-DEN`: This set was generated by DeNegre [2011] and contains 320 knapsack interdiction problems. These problems originate from the *the Multiple Criteria Decision Making library* [Figueira, 2000] and has the same structure as (MIPINT). The number of first-level variables ($n_1 = n_2$) varies in $n_1 \in \{10, 11, ..., 19, 20, 30, 40, 50\}$ and the number of first- and second-level constraints are 1 and $n_1 + 1$, respectively. There are 20 instances corresponding to each level of n_1, except 40 instances for $n_1 \in \{10, 20\}$. Due to the difficulty of the instances with $n_1 = 50$, we excluded them in the experiments.
- `IBLP-DEN`: This set was also generated by DeNegre [2011]. It contains 50 problems comprised of 10 instances with $(n_1, n_2) = (5, 10)$, 10 instances with $(n_1, n_2) = (10, 10)$, 10 instances with $(n_1, n_2) = (15, 5)$ and 20 instances with $(n_1, n_2) = (15, 5)$. All first- and second-level variables are integer and the number of first- and second-level constraints are 0 and 20, respectively.
- `IBLP-ZHANG`: This set was generated by Zhang and Ozaltın [2017] and includes 30 instances with binary first-level variables, integer second-level variables and no first-level constraints. The number of first-level variables varies in $n_1 \in \{50, 60, 70, 80, 90\}$ and the number of second-level variables is set to $n_2 = n_1 + 20$. There are 3 instances with $m_2 = 6$ and also 3 instances with $m_2 = 7$ corresponding to each level of n_1.
- `IBLP-FIS`: This set was generated by Fischetti et al. [2018] and originates from `MILPLIB 3.0` [Bixby et al., 1998]. This set includes 57 instances in which all variables are binary and there are no first-level constraints. Furthermore, n_1 and n_2 vary in ranges $3 - 78734$ and $2 - 78733$, respectively. Due to the memory limit, we did not consider 3 instances of this set in our experiments.

`MibS` 1.1.1 was employed for conducting all experiments and all computational results we report were generated on compute nodes running the Linux (Debian 8.7) operating system with dual

AMD Opteron 6128 processors and 32 GB RAM. All experiments were run sequentially and the time limit was 3600 seconds. SYMPHONY was employed as the MILP solver, while preprocessing and primal heuristics were disabled and the pseudocost branching strategy was used to choose the best variable among the branching candidates in `MibS`. In all numerical experiments, the generation of generic MILP cuts by CGL in `MibS` was disabled and the cuts were generated only when the optimal solution of relaxation was infeasible, but satisfied integrality requirements. Furthermore, all other parameters of `MibS` were set to their default values as described in [Tahernejad et al., 2020], unless otherwise noted.

6.1 Comparing Performance of MIBLP Cuts

In this section, we compare the performance of different valid inequalities for the described data sets. We also investigate the impact of the `branchStrategy` parameter of `MibS` on the performance of cuts.

For plotting each figure shown in Sections 6.1.1–6.1.3, we first solved all instances of its corresponding data set with all considered methods in that figure. Then, we selected only the problems that (i)could be solved by at least one method in 3600 seconds, (ii)whose solution time exceeds 5 seconds for at least one method and (iii)call the cut generator at least one time during the solution process. Furthermore, the plotted figures in these sections show *virtual best performance profiles* with solution time or number of processed nodes as the criteria. In a virtual best performance profile, for a specific method, the point `(x,y)` shows that for the fraction `y` of all instances, the value of the investigating criterion by employing this method is less than or equal to `x` times of the best value among all methods.

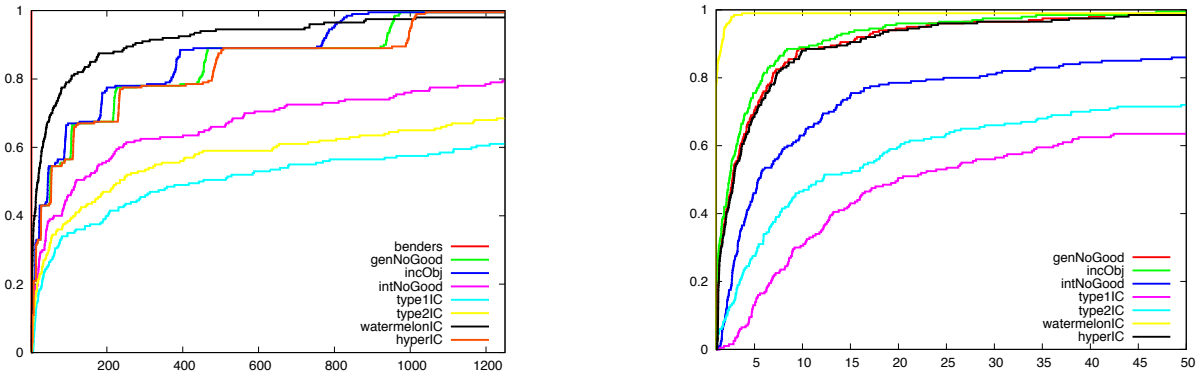
6.1.1 `INTERD-DEN` Set

In order to evaluate the impact of different MIBLP cuts on the instances of this set, we employed eight different methods:

- `benders`: The benders cut was on.
- `genNoGood`: The strengthened no-good cut was on.
- `incObj`: The increasing objective cut was on.
- `intNoGood`: The integer no-good cut was on.
- `type1IC`: The intersection cut I was on.
- `type2IC`: The intersection cut II was on.
- `watermelonIC`: The watermelon intersection cut was on.
- `hyperIC`: The hypercube intersection cut was on.

Note that since the required solution times of the `benders` method for some of the accepted instances were very small, we added 1 second to the required solution times of all methods for all accepted instances.

In the first set of experiments, the `branchStrategy` parameter was set to `fractional`. The performance profile shown in Figure 7a compares all eight described methods with the solution time as the performance measure (for 200 accepted instances). This figure shows the superiority of the benders cut over the other cuts. In order to get a better understanding of the performance of the other cuts, Figure 7b was plotted which compares the performance of all methods other than `benders`. This figure shows that `watermelonIC` is the runner-up after the `benders` method and `incObj` takes the third place.



(a) Comparing all cuts

(b) Excluding the `benders` method

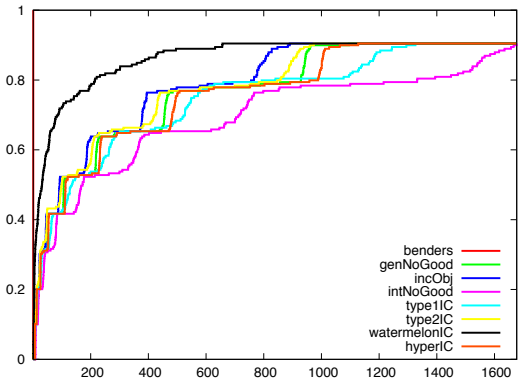
Figure 7: Comparing the performance of different cuts on the `INTERD-DEN` set (with the `fractional` branching strategy)

The `branchStrategy` parameter was set to `linking` in the second set of experiments. The performance profile shown in Figure 8a (for 199 accepted instances) compares the solution time of the eight described methods and it shows that the benders and watermelon intersection cuts perform better comparing with the other cuts. Figure 8b was plotted in the same way as Figure 7b and it shows that `type2IC` and `incObj` are the best methods after `benders` and `watermelonIC`. By comparing the Figures 7b and 8b, one can observe that the type of employed branching strategy can affect the performance of cuts.

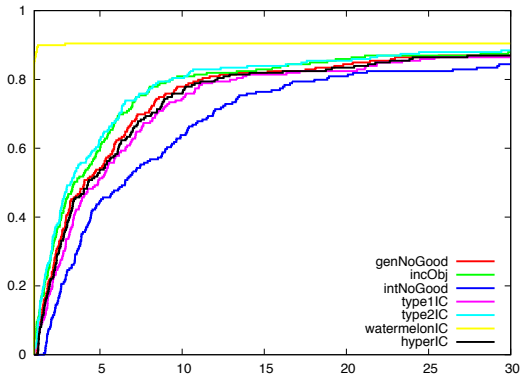
We also investigated the impact of using the benders and watermelon intersection cuts together and the results are shown in Figure 9 (for both branching strategies). Figures 9a and 9b show that the performance of benders cut cannot be improved by combining it with watermelon intersection cut in none of the `fractional` and `linking` branching strategies.

6.1.2 `IBLP-DEN` Set

A set of experiments with five different methods, from the ones described in Section 6.1.1, were conducted on the `IBLP-DEN` set to assess the performance of different cuts on the instances of this set. These methods are `intNoGood`, `type1IC`, `type2IC`, `watermelonIC` and `hyperIC`. Each of these

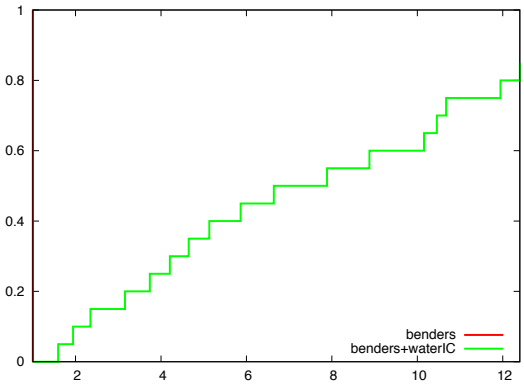


(a) Comparing all cuts

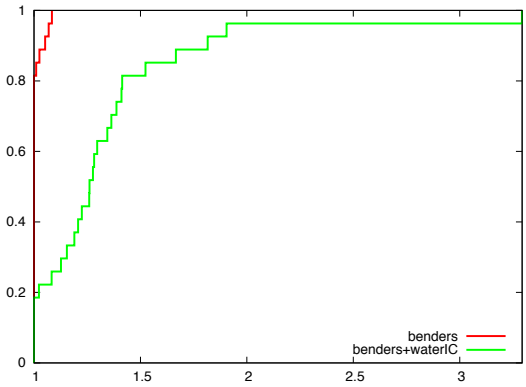


(b) Excluding the `benders` method

Figure 8: Comparing the performance of different cuts on the `INTERD-DEN` set (with the `linking` branching strategy)



(a) `Fractional` branching strategy

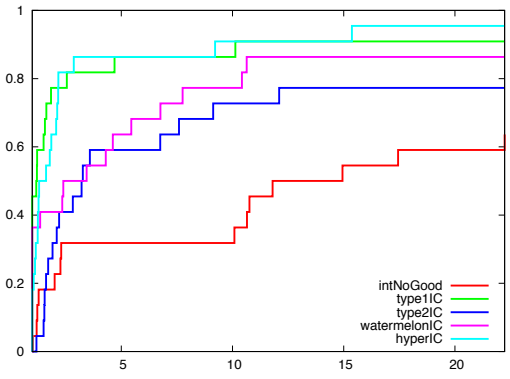


(b) `Linking` branching strategy

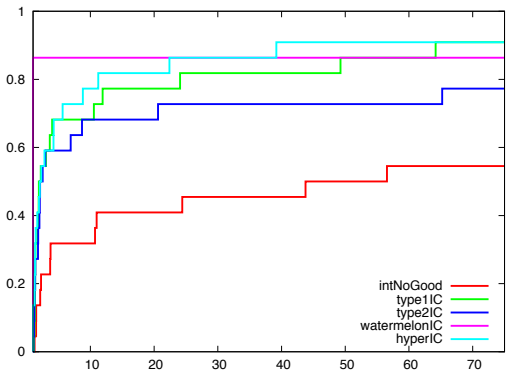
Figure 9: Impact of employing both benders and watermelon intersection cuts on the `INTERD-DEN` set

methods was tested with both `fractional` and `linking` branching strategies, but since the number of accepted instances for the `linking` strategy was 14 and the number of generated cuts for the majority of these accepted problems was very small, it was not appropriate for representing the performance of cuts, so we just report the results for the `fractional` branching strategy.

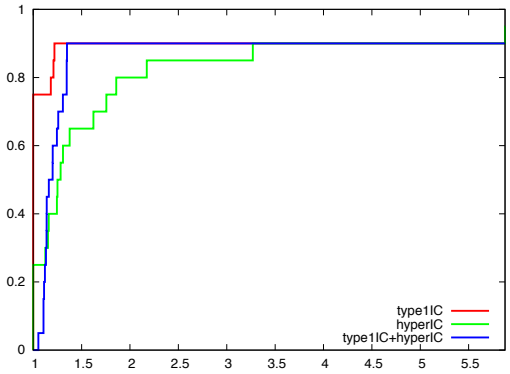
The performance profile shown in Figure 10a (for 22 accepted instances) compares the solution time of the described methods and one can observe `type1IC` and `hyperIC` perform better comparing with the other methods. Figure 10b shows the performance profile for these methods with the number of processed nodes as the performance measure. This figure shows the superiority of `watermelonIC` and `hyperIC` from the point of size of tree, which means that watermelon intersection cut is strong, but its generation is expensive. Figure 10c compares the solution time of turning on both type I and hypercube intersection cuts with using just each of these cuts. Based on this figure, `type1IC+hyperIC` is not effective in decreasing the solution time.



(a) Comparing the solution time



(b) Comparing the number of processed nodes



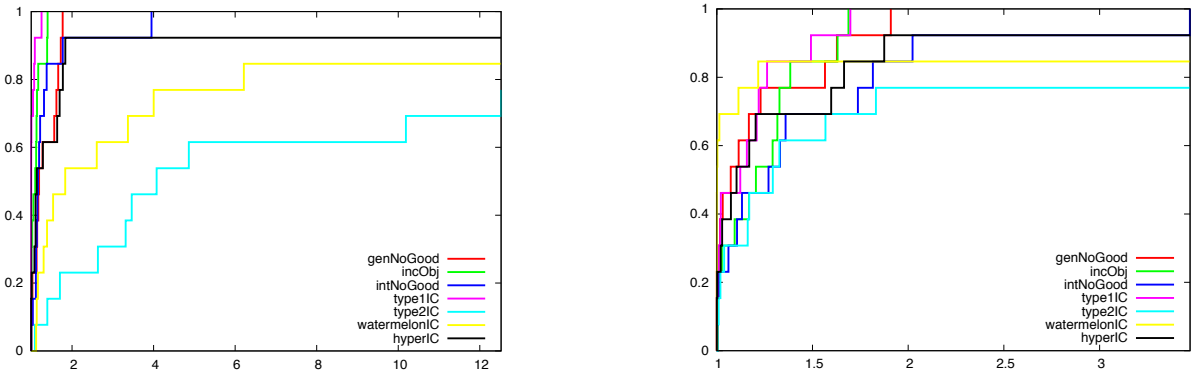
(c) Employing both type I and hypercube intersection cuts

Figure 10: Comparing the performance of different cuts on the `IBLP-DEN` set

6.1.3 `IBLP-ZHANG` Set

In order to investigate the performance of different cuts for the `IBLP-ZHANG` set, we conducted a set of experiments with seven different methods on the instances of this set. These methods are `genNoGood`, `incObj`, `type1IC`, `type2IC`, `watermelonIC` and `hyperIC` (as were described in Section 6.1.1).

All methods were tested with both `linking` and `fractional` branching strategies, but due to the same reasons stated for the `IBLP-DEN` set, we here only analyze the results of using the `fractional` branching strategy. Figure 11a shows the performance profile (for 13 accepted instances) for the described methods with the solution time as the performance measure. This figure shows the superiority of `type1IC` and `incObj` methods. Furthermore, the performance profile shown in Figure 11b compares the number of processed nodes for the above methods and in the same way as Figure 10b, one can observe that the watermelon intersection cut performs better with respect to the size of tree.



(a) Comparing the solution time

(b) Comparing the number of processed nodes

Figure 11: Comparing the performance of different cuts on the `IBLP-ZHANG` set

6.1.4 `IBLP-FIS` Set

For evaluation of the performance of different cuts on this set, all instances were first solved with the methods `genNoGood`, `intNoGood`, `type1IC`, `type2IC`, `watermelonIC` and `hyperIC` (as illustrated in Section 6.1.1). The branching strategy was set to `fractional`. Since the number of accepted instances was very small by using the selection criteria for the other test sets, we accepted all instances which call the cut generator at least one time during the solution process. The results of 28 accepted instances are shown in Figure 12. This plot is composed of two subplots with solution time and duality gap as the performance measures. In the left (right) plot, the point `(x,y)` for a specific method shows that the solution time (gap) of the fraction `y` of instances is not greater than `x` by employing that method. This figure shows that the watermelon intersection cut has better performance comparing with the other cuts for this test set.

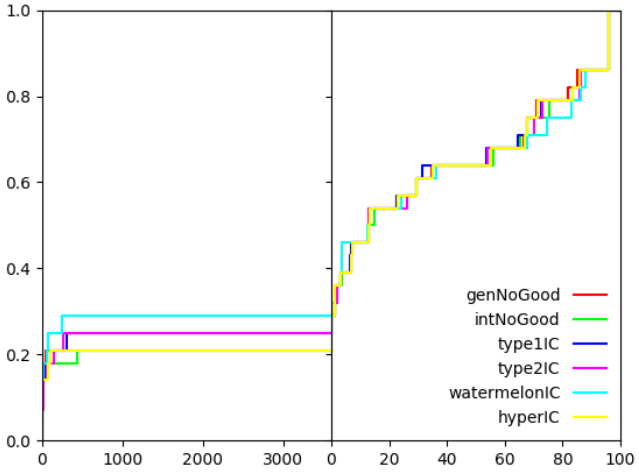


Figure 12: Comparing the performance of different cuts on the `IBLP-FIS` set

6.2 Strengthening Integer No-good Cut

We illustrated in Section 4 that how a cut can be strengthened by improving its right-hand side. In this section, we show the results of employing this idea for improving the performance of integer no-good cut for the `INTERD-DEN` and `IBLP-DEN` sets.

As we explained in Section 4, it is not required to solve problem (22) to optimality and the selected values for its termination parameters can affect the performance of generated cuts. The decision about the nodes in which the generated cuts are strengthened can also affect the solution process. Hence, for each data set, we tested multiple methods which are different in the termination criteria of problem (22) and the level of the search tree that we start strengthening the cuts. The results of a subset of these methods are shown in Figures 13 and 14. These figures show the *base best performance profiles* for the solution time and number of processed nodes. In a base best performance profile, for a specific method, the point `(x,y)` shows that for the fraction `y` of all instances, the value of the investigating criterion by employing this method is less than or equal to `x` times of the value of the criterion for the base method. A base best performance profile provides an appropriate tool for comparing each method with the base method.

The first set of experiments were conducted on the `INTERD-DEN` set. The branching strategy was set to `linking` for all methods and the the integer no-good cut (without strengthening) was turned on in the base method. The other methods are:

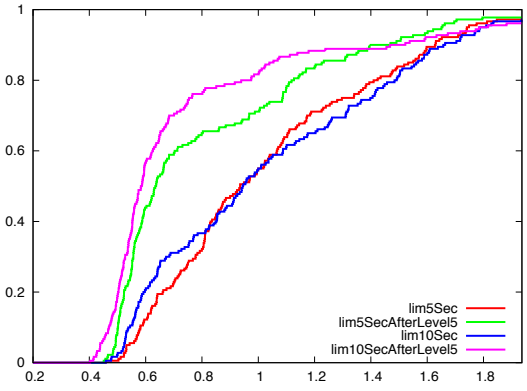
- `limTSec`: The integer no-good cut was on and the cuts were strengthenth by setting the time limit of problem (22) to `T` seconds for $\texttt{T} \in \{5, 10\}$.
- `limTSecAfterLevelL`: The same as `limTSec`, but only the cuts generated after the $\texttt{L}^{th}$ level of the search tree were strengthenth for $\texttt{T} \in \{5, 10\}$ and $\texttt{L} \in \{5\}$.

Whenever it was needed to solve problem (22) in these methods, the branching strategy was set to `linking` and all cuts were turned off. All other parameters were set to their default values. In order to plot Figure 13 (and also Figure 14), we considered the instances (i)whose solution time is greater than 5 seconds and less than 3600 seconds by employing the base method and (ii)which call the cut generator at least one time during the solution process.

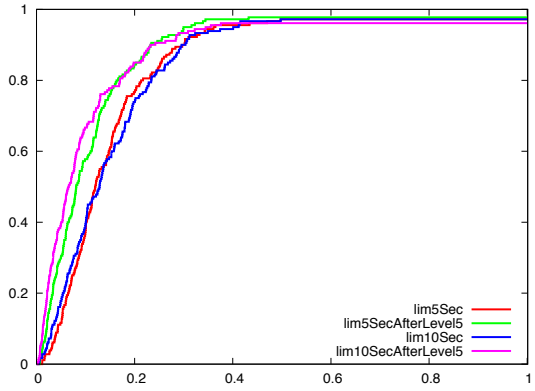
The performance profile shown in Figure 13a (for 180 accepted instances) compares the solution time of the described methods with the base method. It shows that strengthening the integer no-good cut can decrease the solution time of most of the problems and `lim10SecAfterLevel5` performs better than the other ones. Moreover, Figure 13b compares the number of the processed nodes and one can observe that strengthening the generated cuts can decrease the size of tree substantially.

In the second set of experiments, we considered the `IBLP-DEN` set. The branching strategy was set to `fractional` for all methods and the the integer no-good cut (without strengthening) was turned on in the base method. The other methods are `limTSecAfterLevelL` for $\texttt{T} \in \{2, 5\}$ and $\texttt{L} \in \{10, 15\}$. The setting for solving the problem (22) was the same as the one used for the `INTERD-DEN` set.

The Figures 14a and 14b compare the performance of the described methods with the base method with the solution time and the number of processed nodes as the performance measures, respectively



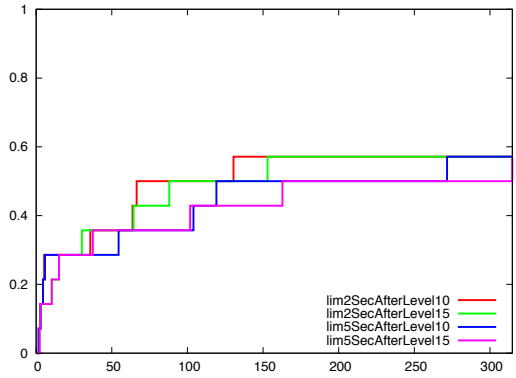
(a) Comparing the solution time



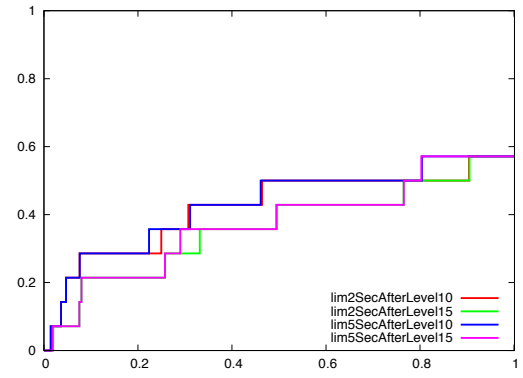
(b) Comparing the number of processed nodes

Figure 13: Impact of strengthening the integer no-good cut on the `INTERD-DEN` set

(for 14 accepted instances). As one can observe, none of the methods could not improve the solution time of the base method, however, they can decreases the number of nodes for some instances.



(a) Comparing the solution time



(b) Comparing the number of processed nodes

Figure 14: Impact of strengthening the integer no-good cut on the `IBLP-DEN` set

6.3 Impact of Bound Cut

In this section, we evaluate the impact of employing non-parametric and parametric bound cuts on the instances of the `INTERD-DEN` and `IBLP-DEN` sets.

For each data set (for both non-parametric and parametric bound cuts), we tested multiple methods which differ in (i)the termination parameters of problem (22) and (ii)the nodes in which bound cut is generated. In the same way as Section 6.2, figures shown in this section are base best performance profiles. The description of the investigated methods in this section are:

- `onlyRootTimeTNodeN`: The same as the base method, but the bound cut was generated at the root node with `T` seconds and `N` nodes as the time and node limits for the problem (22),

respectively.

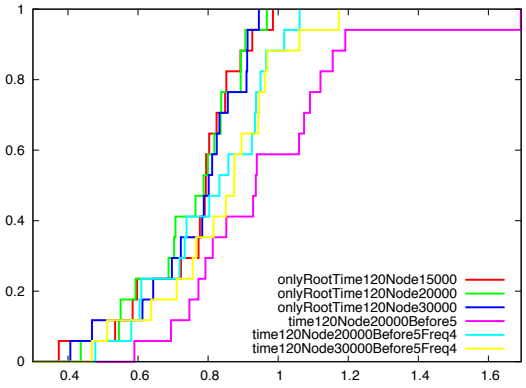
- `onlyRootNodeN`: The same as the method `onlyRootTimeTNodeN`, but there was no time limit for the problem (22).
- `timeTNodeNBeforeLFreqF`: The same as the base method, but the bound cut was generated with frequency `F`, whenever the cut generator was called for the nodes whose level in the tree was less than `L`. Moreover, the time and node limits for the problem (22) were set to `T` seconds and `N` nodes, respectively.
- `timeTNodeNBeforeL`: The same as the method `timeTNodeNBeforeLFreqF`, but the frequency was set to 1.
- `nodeNBeforeLFreqF`: The same as the method `timeTNodeNBeforeLFreqF`, but there was no time limit for the problem (22).
- `nodeNFreqF`: The same as the method `nodeNBeforeLFreqF`, but there was no limit on the level of nodes in which the bound cut was generated.

In the first set of experiments, we considered the `INTERD-DEN` set. In the experiments for both non-parametric and parametric bound cuts, we set the branching strategy to `linking` for all methods and only benders cut was turned on in the base method. Furthermore, in all methods, we only turned on the benders cut in solving problem (22) and all other parameters (except the termination criteria) were set to their default values. For this data set (and also the `IBLP-DEN` set), we selected the instances (i)whose solution time is greater than 50 seconds and less than 3600 seconds by employing the base method and (ii)which call the cut generator at least one time during the solution process. Figures 15 and 16 show the results for the parametric and non-parametric bound cuts for this data set (for 17 accepted instances), respectively.

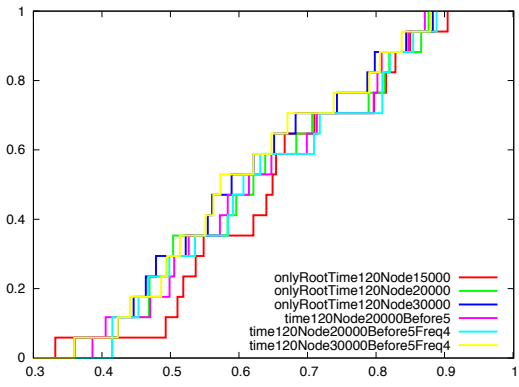
Figure 15a shows that employing the parametric bound cut can decrease the solution time and the best settings are the three methods in which we generated the bound cut only in the root node (i.e., the methods `onlyRootTime120NodeN` for `N` $\in \{15000, 20000, 30000\}$). As one can observe, the performance of these methods is degraded as we increased the number of generated bound cuts. Note that when we generate the bound cut only in the root node, the generated cut can be considered as either parametric or non-parametric bound cuts because the approach of generating bound cut in the root node is the same for both of these categories. Furthermore, Figure 15b shows that generating the parametric bound cut has a considerable effect on decreasing the number of processed nodes.

As one can observe in Figure 16a, employing the non-parametric bound cut can decrease the solution time and similar to the parametric case, the best methods are those in which we generated the bound cut only in the root node. Moreover, Figure 16b shows that generating non-parametric bound cut can be effective in decreasing the number of processed nodes.

The second set of experiments were conducted on the `IBLP-DEN` set. The branching strategy was set to `fractional` for all methods (both non-parametric and parametric cases) and only the hypercube intersection cut was turned on in the base method. In all methods for parametric bound cut, we only turned on the intersection cut I in solving problem (22) and all other parameters (except the termination criteria) were set to their default values. This problem was solved with the same setting

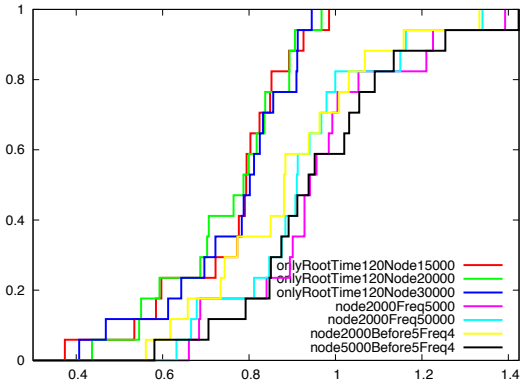


(a) Comparing the solution time

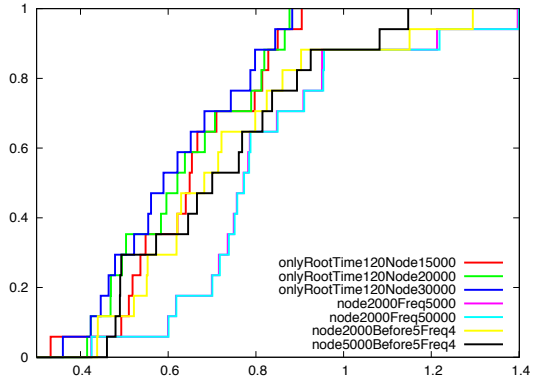


(b) Comparing the number of processed nodes

Figure 15: Impact of generating parametric bound cut on the `INTERD-DEN` set



(a) Comparing the solution time



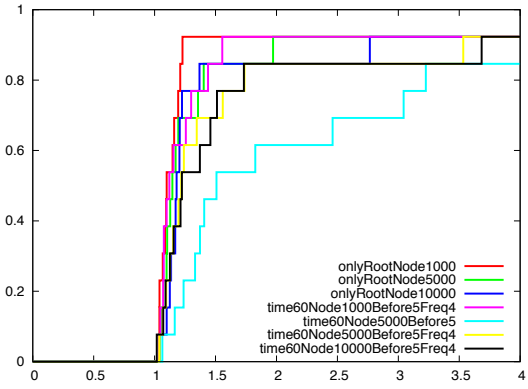
(b) Comparing the number of processed nodes

Figure 16: Impact of generating non-parametric bound cut on the `INTERD-DEN` set

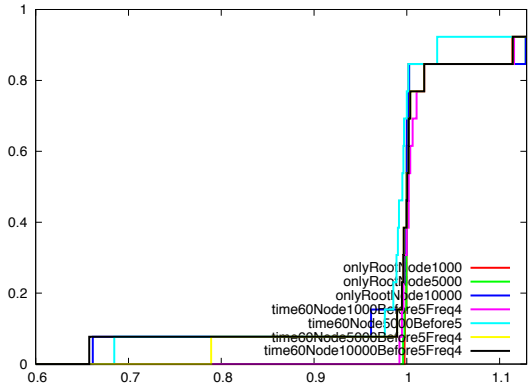
in the methods for the non-parametric bound cut, except that only hypercube intersection cut was enabled. The results of employing the parametric and non-parametric bound cuts are shown in Figures 17 and 18 (for 13 accepted instances), respectively. As one can observe, neither parametric nor parametric bound cuts are effective in decreasing the solution time and number of processed nodes for this set.

6.4 Evaluation of Strength of Different Cuts

We selected randomly 10 instances of the `INTERD-DEN` and `IBLP-DEN` sets and solved each instance with different cuts as shown in Tables 1 and 2. For each instance, we considered the first generated cut and compared its right-hand side with the best possible right-hand side. These values are shown in the `Orig RHS` and `Best RHS` columns, respectively. The `Obj before cut`, `Obj after orig cut` and `Obj after best cut` columns show, respectively, the objective values of the relaxation problem before adding the first cut, after adding this cut and after adding this cut with the best right-hand side.

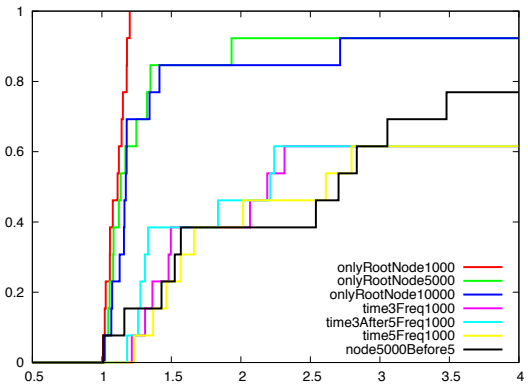


(a) Comparing the solution time

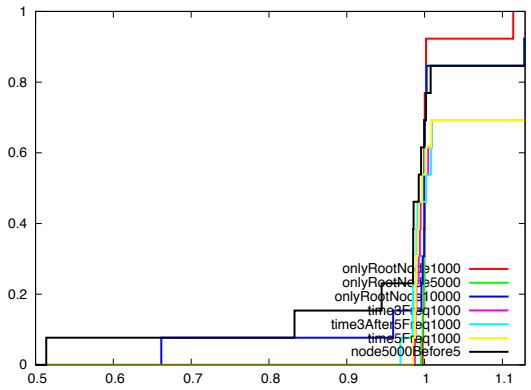


(b) Comparing the number of processed nodes

Figure 17: Impact of generating parametric bound cut for the `IBLP-DEN` set



(a) Comparing the solution time



(b) Comparing the number of processed nodes

Figure 18: Impact of generating non-parametric bound cut for the `IBLP-DEN` set

As one can observe in Table 1, the right-hand sides of the generated benders cuts are equal to the best possible right-hand sides for all five instances and it is one of the reasons of the strength of this cut. Furthermore, the right-hand sides of the generated watermelon intersection cuts are closer to their best possible values comparing with the integer no-good cuts and it verifies the results shown in Section 6.1.1.

Note that in Table 2, for the instances 1, 2 and 3, we observed that the bilevel feasible region of the node in which the first cut is generated is empty and it results that the best value of the right-hand side is ∞ (because the best right-hand side is obtained by minimizing the left-hand side over the bilevel feasible region) and the node should be pruned. This table also shows that the right-hand sides of the watermelon and hypercube intersection cuts are more closer to their best values in comparison with the integer no-good cuts and it verifies the superiority of these cuts over the integer no-good cut in Section 6.1.2. Moreover, since the hypercube intersection cut and strengthened no-good cut may remove a part of the non-improving bilevel feasible solutions, the right-hand sides of these cuts may be greater than their best possible values and it can be observed for the instances 3 and 4.

Table 1: Analysis of the right-hand sides of different cuts for the `INTERD-DEN` set

Instance	Obj	Benders cut				Watermelon intersection cut				Integer no-good cut			
	before cut	Orig RHS	Best RHS	Obj after orig cut	Obj after best cut	Orig RHS	Best RHS	Obj after orig cut	Obj after best cut	Orig RHS	Best RHS	Obj after orig cut	Obj after best cut
1	0	4520	4520	424.64	424.64	1	1.64	0	514.43	1	7	0	0
2	0	6252	6252	1270.08	1270.08	1	1.11	0	407.73	1	12	0	659.91
3	0	5662	5662	1285.7	1285.7	1	1.07	0	61.21	1	14	0	481.27
4	0	7113	7113	1274.73	1274.73	1	1.15	0	390.19	1	16	0	701.53
5	0	9685	9685	1720.64	1720.64	1	1.11	0	406.65	1	18	0	1070.59

Table 2: Analysis of the right-hand sides of different cuts for the `IBLP-DEN` set

Instance	Obj	Watermelon intersection cut				Hypercube intersection cut				Integer no-good cut			
	before cut	Orig RHS	Best RHS	Obj after orig cut	Obj after best cut	Orig RHS	Best RHS	Obj after orig cut	Obj after best cut	Orig RHS	Best RHS	Obj after orig cut	Obj after best cut
1	-669	0	∞	-623.47	prune	5	∞	-645	prune	-789	∞	-668.5	prune
2	-688	0	∞	-544	prune	1	∞	-632.46	prune	-29	∞	-676.36	prune
3	-766	0	0	-286	-286	3	2	-761.08	-766	-17	-4	-761.08	-508.05
4	-724	0	0	-578.63	-578.63	19	18	-709.75	-724	-407	-402	-721	-706
5	-659	0	∞	-656.67	prune	15	∞	prune	prune	-603	∞	-656.83	prune

7 Conclusions

Although the generation of valid inequalities has proven to be an invaluable tool in solving MIBLPs, there remains a large scope for developing the fundamental theory underlying the branch-and-cut algorithms that have become a standard solution method. We have taken an initial step towards filling in some of the gaps, as well as providing a framework within which to view the known methods, clarifying the relationship to and distinction from similar methods already studied in the MILP case. We also introduced new general methodology for generating valid inequalities that applies not only to MIBLPs, but also potentially to other classes of optimization problem (such as MILPs themselves). Finally, we provided a comprehensive comparison of known methods for the generation of valid inequalities. There remains a wide range of possibilities for innovation in leveraging the decades of research solving MILPs to derive improved methodology for MIBLPs. It is our hope that this work motivates future work along these lines, leading to further discoveries in this important field of research.

Acknowledgements

This research was made possible with support from National Science Foundation Grants CMMI-1435453, CMMI-0728011, and ACI-0102687, as well as Office of Naval Research Grant N000141912330.

References

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

E. Balas. Disjunctive programming. In *Annals of Discrete Mathematics 5: Discrete Optimization*, pages 3–51. North Holland, 1979.

E. Balas. Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. *SIAM Journal on Algebraic Discrete Methods*, 6(3):466–486, 1985.

E. Balas. Disjunctive programming: Properties of the convex hull of feasible points. *Discrete Applied Mathematics*, 89:3–44, 1998.

E. Balas, S. Ceria, and G. Corneujols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Mathematical Programming*, 58:295–324, 1993a.

E. Balas, S. Ceria, and G. Cornuéjols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Mathematical Programming*, 58:295324, 1993b.

J. Bard and J. Moore. A branch and bound algorithm for the bilevel programming problem. *SIAM Journal on Scientific and Statistical Computing*, 11(2):281–292, 1990.

A. Basu, C. T. Ryan, and S. Sankaranarayanan. Mixed-integer bilevel representability. *arXiv preprint arXiv:1808.03865*, 2018.

D. Bienstock, C. Chen, and G. Munoz. Outer-product-free sets for polynomial optimization and oracle-based cuts. *arXiv preprint arXiv:1610.04604*, 2016.

R. E. Bixby, S. Ceria, C. M. McZeal, and M. W. Savelsbergh. An updated mixed integer programming library: Miplib 3.0. Technical report, 1998.

S. Bolusani and T. Ralphs. A Framework for Generalized Benders' Decomposition and Its Application to Multilevel Optimization. Technical report, COR@L Laboratory Technical Report 20T-004, Lehigh University, 2020. URL http://coral.ie.lehigh.edu/~ted/files/papers/MultilevelBenders20.pdf.

S. Bolusani, S. Coniglio, T. Ralphs, and S. Tahernejad. A Unified Framework for Multistage Mixed Integer Linear Optimization. Technical report, COR@L Laboratory Technical Report 20T-005, Lehigh University, 2020. URL http://coral.ie.lehigh.edu/~ted/files/papers/MultistageFramework20.pdf.

A. Bulut and T. Ralphs. On the Complexity of Inverse Mixed Integer Linear Optimization. Technical report, COR@L Laboratory Technical Report 15T-001-R3, Lehigh University, 2015. URL http://coral.ie.lehigh.edu/~ted/files/papers/InverseMILP15.pdf.

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

Cgl. Cut generator library, 2017. URL https://projects.coin-or.org/Cgl.

M. Conforti, G. Cornuéjols, and G. Zambelli. *Integer programming*, volume 271. Springer, 2014.

W. Cook, , R. Kannan, and A. Schrijver. Chva´tal Closures For Mixed Integer Programming Problems. *Mathematical Programming*, 47(2):155–174, 1990.

G. Cornuéjols. Valid inequalities for mixed integer linear programs. *Mathematical Programming*, 112(1):3–44, 2008.

A. Del Pia and R. Weismantel. On convergence in mixed integer programming. *Mathematical programming*, 135(1-2):397–412, 2012.

S. Dempe. *Foundations of Bilevel Programming*. Kluwer Academic Publishers, 2002.

S. DeNegre. *Interdiction and Discrete Bilevel Linear Programming*. PhD, Lehigh University, 2011. URL http://coral.ie.lehigh.edu/~ted/files/papers/ScottDeNegreDissertation11.pdf.

S. DeNegre and T. Ralphs. A Branch-and-Cut Algorithm for Bilevel Integer Programming. In *Proceedings of the Eleventh INFORMS Computing Society Meeting*, pages 65–78, 2009. doi: 10.1007/978-0-387-88843-9_4. URL http://coral.ie.lehigh.edu/~ted/files/papers/BILEVEL08.pdf.

S. DeNegre, T. Ralphs, and S. Tahernejad. MibS version 1.1.1 2019. doi: 10.5281/zenodo.1439102. URL https://github.com/coin-or/MibS.

J. Figueira. MCDM Numerical Instances Library, 2000. URL http://www.univ-valenciennes.fr/ROAD/MCDM/ListMOKP.html.

M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl. A new general-purpose algorithm for mixed-integer bilevel linear programs. *Operations Research*, 65(6):1615–1637, 2017.

M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl. On the use of intersection cuts for bilevel optimization. *Mathematical Programming*, 172:77–103, 2018.

D. Gade and S. Küçükyavuz. Pure cutting-plane algorithms and their convergence. *Wiley Encyclopedia of Operations Research and Management Science*, pages 1–11, 2011.

R. E. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Monthly*, 64:275–278, 1958.

R. E. Gomory. A algorithm for the mixed integer problem. Technical report, The RAND Cooperation, 1960.

M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, New York, 1993.

A. Hassanzadeh and T. Ralphs. On the Value Function of a Mixed Integer Linear Optimization Problem and an Algorithm for Its Construction. Technical report, COR@L Laboratory Technical Report 14T-004, Lehigh University, 2014. URL http://coral.ie.lehigh.edu/~ted/files/papers/MILPValueFunction14.pdf.

P. Loridan and J. Morgan. Weak via strong stackelberg problem: New results. *Journal of Global Optimization*, 8(3):263–287, 1996.

H. Marchand, A. Martin, R. Weismantel, and L. Wolsey. Cutting planes in integer and mixed integer programming. *Discrete Applied Mathematics*, 123(1):397–446, 2002.

R. Meyer. On the existence of optimal solutions to integer and mixed integer programming problems. *Mathematical Programming*, 7:223–235, 1974.

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

J. Munkres. *Topology*. Pearson Education, 2014.

G. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley and Sons, New York, 1988.

L. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3:1–22, 1976.

S. Tahernejad, T. Ralphs, and S. DeNegre. A Branch-and-Cut Algorithm for Mixed Integer Bilevel Linear Optimization Problems and Its Implementation. *Mathematical Programming Computation*, 12:529–568, 2020. doi: 10.1007/s12532-020-00183-6. URL http://coral.ie.lehigh.edu/~ted/files/papers/MIBLP16.pdf.

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

K. Wolter. Implementation of Cutting Plane Separators for Mixed Integer Programs. Master's thesis, Technische Universität Berlin, 2006.

J. Zhang and O. Y. Ozaltın. A branch-and-cut algorithm for discrete bilevel linear programs. *Optim. Online*, 2017.