Submitted to *INFORMS Journal on Computing*
manuscript (Please, provide the mansucript number!)

Computational Experiments with Cross and Crooked Cross Cuts

Sanjeeb Dash, Oktay Günlük
IBM Research, sanjeebd@us.ibm.com, gunluk@us.ibm.com

Juan Pablo Vielma
Massachusetts Institute of Technology, jvielma@mit.edu

In this paper, we study whether cuts obtained from two simplex tableau rows at a time can strengthen the bounds obtained by GMI cuts based on single tableau rows. We also study whether cross and crooked cross cuts, which generalize split cuts, can be separated in an effective manner for practical MIPs and can yield a non-trivial improvement over the bounds obtained by split cuts. We give positive answers to both these questions for MIPLIB 3.0 problems. Cross cuts are a special case of the t-branch split cuts studied by Li and Richard (2008). Split cuts are 1-branch split cuts, and cross cuts are 2-branch split cuts. Crooked cross cuts were introduced by Dash, Dey and Günlük (2010) and were shown to dominate cross cuts in Dash, Günlük and Molinaro (2012).

Key words: mixed integer programming; cutting planes; elementary closures
History:

1. Introduction

The Gomory mixed-integer (GMI) cut and its variants are currently the most important cutting planes for solving general mixed-integer programs (MIPs), and are incorporated in most commercial MIP solvers (Balas et al. 1996, Bixby et al. 2000). The GMI cut can be derived, via an algebraic approach, as a mixed-integer rounding (MIR) inequality from a single row relaxation of an MIP; in practice, it is usually derived from a simplex tableau row associated with an MIP. Alternatively, it can be derived, via a geometric approach, as a disjunctive cut and more specifically as a split cut. Recently, Balas and Saxena (2008) and Dash et al. (2010) approximately optimized over the split closure of practical MIP instances and obtained very strong bounds on the optimal values of such instances. There

has been much recent work on generalizing GMI cuts in different ways to obtain more effective cutting planes. An important goal of this recent work is to obtain cuts that significantly improve the bounds yielded by GMI cuts.

A well-studied generalization of the GMI cut involves the use of lattice-free sets to generate cutting planes (*lattice-free cuts*) from multiple rows of a simplex tableau associated with an MIP. Following the work of Andersen et al. (2007), who study lattice-free cuts from relaxations of an MIP with two rows, there have been many theoretical studies of extensions with two or more rows (Borozan and Cornuéjols 2009) and with additional structure (Dey and Wolsey 2010b, Basu et al. 2010, Dey and Wolsey 2010a, Fukasawa and Günlük 2011, Conforti et al. 2011b). See Dey and Tramontani (2009) and Conforti et al. (2011a) for recent surveys on this topic. Computational work on this topic is reported first by Espinoza (2010), and subsequently by Louveaux and Poirrier (2012), Dey et al. (2010), and Basu et al. (2011). However, these efforts have not resulted in computationally effective cutting planes for solving practical MIPs.

A different generalization of the GMI involves using disjunctions that combine two or more split disjunctions instead of a single split disjunction. One such generalization is the *t-branch split* cut (for integers $t \geq 2$) introduced by Li and Richard (2008). Yet another generalization is the *crooked cross* cut, introduced by Dash et al. (2012a, 2011). The crooked cross cut dominates the 2-branch split cut (also called the cross cut in Dash et al. (2012a)).

In earlier computational work on cuts from multiple tableau rows, Espinoza (2010) generated lattice-free cuts by considering up to 10 tableau rows simultaneously. However, his work does not measure the relative strength of GMI cuts versus multi-row cuts in the elementary closure sense as in later rounds of his algorithm, cuts are generated using earlier cuts. Louveaux and Poirrier (2012) give a fast algorithm to separate all lattice-free cuts from two-row relaxations arising from pairs of tableau rows. They consider a subset of all such pairs and generate lattice-free cuts, augment the LP relaxation with these cuts and derive a new tableau (up to five times), but they do not lift the cuts with respect to the non-basic integral variables. Recently, Dey et al. (2010) also experimented with cuts obtained from pairs of tableau rows. Their approach involves first relaxing the integrality of non-basic integral variables and then finding violated 2 dimensional lattice-free cuts and finally lifting the cut coefficients to re-introduce integrality information. These lifted lattice-free cuts are known to be crooked cross cuts (Dash et al. 2012a) but the converse is

not true (Dash et al. 2012b). Their computational experiments are performed on randomly generated problems, where they get a nontrivial improvement over one round of GMI cuts, but their procedure has limited success on practical MIPs. Basu et al. (2011) also separate triangle cuts of type 2 but conclude that their "family of two-row cuts is not competitive with GMI cuts in terms of gap closed" for MIPLIB problems. Their approach is essentially the same as that of Dey et al. (2010) and has some inherent difficulties. Firstly, it is not always clear what the best way to lift a given lattice-free cut is. Secondly, it is hard to decide which lattice-free cut for the two-row continuous group relaxation would yield a good cut after lifting. Therefore, although Louveaux and Poirrier (2012) can separate lattice free cuts from such a relaxation quickly, it is still not clear if this relaxation is the best model for obtaining cuts from pairs of tableau rows.

In this paper, we first investigate the strength of split, cross and crooked cross cuts that are obtained using two row relaxations of MIPs. To obtain the relaxations, we take pairs of rows of the optimal tableau associated with the LP relaxation of an MIP together with the integrality and bound information on the variables. This approach is different from the lattice-free cut based approaches in Basu et al. (2011) and Dey et al. (2010) as we use the bound and integrality information on the variables directly. As in Basu et al. (2011) and Dey et al. (2010), we find violated cuts heuristically. We obtain, for the first time, bounds for the MIPLIB 3.0 problems (Bixby et al. 1998) that nontrivially exceed the bounds obtained by only adding GMI cuts based on these tableau rows, with the biggest fraction of the improvement coming from split cuts. In our second set of computational experiments, we investigate the strength of split, cross and crooked cross cuts using the original formulation of an MIP instead of its two row relaxations. Our computational results show that bounds obtained in Balas and Saxena (2008) and Dash et al. (2010) using split cuts only can be exceeded by using cross and crooked cross cuts. For a number of MIPLIB 3.0 problems, we are able to obtain noticeably better bounds on the optimal objective function value by generating cross cuts.

The remainder of the paper is structured as follows. In Section 2, we define the different classes of cuts used in this paper, and in Section 3, we discuss separation models for these cut classes. In Section 4, we describe the main heuristics we use to separate cuts from these classes. In Section 5, we use the heuristics described in the previous sections to obtain cross cuts from pairs of tableau rows and show that these cuts yield significantly better bounds

than the GMI cuts derived from the same tableau rows. In Section 6, we explain how we find violated cross cuts and crooked cross cuts (from all constraints). We compare bounds on the integer optimum value obtained in this manner with bounds obtained earlier by approximately optimizing over the split closure.

2. Preliminaries

Consider the following mixed-integer set with m rows

$$P = \{(x, y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} : Ax + Gy = b,\ x \geq 0,\ y \geq 0\}$$

where A, G, b are rational matrices with m rows and $n_1, n_2, 1$ columns, respectively. In general, the set of feasible solutions for any mixed-integer linear program can be framed in this way. We denote the linear programming (LP) relaxation of P by P^{LP}. In the remainder of the paper π and a (along with subscripts or superscripts) represent row vectors with n_1 components, and c is a row vector with n_2 components. We next discuss disjunctive cuts for P (see Balas (1979)).

Let $D = \cup_{k \in K} D_k$ where $D_k = \{(x, y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} : A^k x + G^k y \leq b^k\}$ for $k \in K$. If $\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} \subseteq D$, then we call D a *disjunction* and we call each D_k an *atom* of the disjunction D. A linear inequality is called a *disjunctive cut* for P obtained from the disjunction D if it is valid for $P^{LP} \cap D_k$ for all $k \in K$. All disjunctive cuts for P are valid for P. Note that multiple disjunctive cuts can be derived from the same disjunction. In this paper we are interested in the following types of disjunctions.

2.1. Split disjunctions

For a fixed $\pi \in \mathbb{Z}^{n_1} \setminus \{\mathbf{0}\}$, and $\gamma \in \mathbb{Z}$, a *split disjunction* is denoted by $S(\pi, \gamma)$ and defined as

$$S(\pi, \gamma) = S_1(\pi, \gamma) \cup S_2(\pi, \gamma),$$

where $S_1(\pi, \gamma) = \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi x \leq \gamma\}$ and $S_2(\pi, \gamma) = \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi x \geq \gamma + 1\}$. A linear inequality is said to be a *split cut* (Cook et al. 1990) for P if it is valid for $P^{LP} \cap S_1(\pi, \gamma)$ and $P^{LP} \cap S_2(\pi, \gamma)$. We define $P_S(\pi, \gamma)$ as the convex hull of $P^{LP} \cap S(\pi, \gamma)$, and P_S to be the set of points in P^{LP} which satisfy all split cuts for P, i.e., the *split closure* of P. Therefore

$$P_S = \bigcap_{\pi \in \mathbb{Z}^{n_1}} \bigcap_{\gamma \in \mathbb{Z}} P_S(\pi, \gamma). \tag{1}$$

Assume that the following equation is satisfied by all points in P (for example, it could be obtained via a linear combination of the constraints $Ax + Gy = b$):

$$\sum_{i=1}^{n_1} a_i x_i + \sum_{i=1}^{n_2} c_i y_i = \beta. \tag{2}$$

Let $\hat{\beta} = \beta - \lfloor \beta \rfloor$ and assume that $\hat{\beta} \neq 0$. The *mixed-integer rounding* (MIR) cut derived from (2) is

$$\sum_{i=1}^{n_1} (\hat{\beta} \lfloor a_i \rfloor + \min(\hat{a}_i, \hat{\beta})) x_i + \sum_{i=1}^{n_2} \max(c_i, 0) y_i \geq \hat{\beta} \lceil \beta \rceil, \tag{3}$$

where $\hat{a}_i = a_i - \lfloor a_i \rfloor$. It is well known that the MIR cut (3) is a split cut for the following 1-row relaxation of P,

$$P_1 = \{(x, y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} : \sum_{i=1}^{n_1} a_i x_i + \sum_{i=1}^{n_2} c_i y_i = \beta,\ x \geq 0,\ y \geq 0\},$$

using the disjunction $S(\pi, \gamma)$ where

$$\gamma = \lfloor \beta \rfloor \text{ and } \pi_i = \begin{cases} \lfloor a_i \rfloor & \text{if } \hat{a}_i \leq \hat{\beta}, \\ \lceil a_i \rceil & \text{if } \hat{a}_i > \hat{\beta}. \end{cases} \tag{4}$$

Conversely, any split cut for P is an MIR cut derived from a single implied equation (2) for P along with nonnegativity constraints on the variables. When the MIR cut is derived from a row of the simplex tableau associated with the LP relaxation of the mixed-integer program, we call it the *Gomory mixed-integer* (GMI) cut.

2.2. Cross disjunctions and cross cuts

The *cross disjunction* (see Dash et al. (2012a)) associated to $\pi_1, \pi_2 \in \mathbb{Z}^{n_1} \setminus \{\mathbf{0}\}$, and $\gamma_1, \gamma_2 \in \mathbb{Z}$, is given by

$$C(\pi_1, \pi_2, \gamma_1, \gamma_2) = \bigcup_{k \in \{1,2,3,4\}} C_k(\pi_1, \pi_2, \gamma_1, \gamma_2)$$

where:

$$C_1(\pi_1, \pi_2, \gamma_1, \gamma_2) = \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi_1 x \leq \gamma_1,\ \pi_2 x \leq \gamma_2\},$$
$$C_2(\pi_1, \pi_2, \gamma_1, \gamma_2) = \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi_1 x \leq \gamma_1,\ \pi_2 x \geq \gamma_2 + 1\},$$
$$C_3(\pi_1, \pi_2, \gamma_1, \gamma_2) = \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi_1 x \geq \gamma_1 + 1,\ \pi_2 x \leq \gamma_2\}, \text{ and}$$
$$C_4(\pi_1, \pi_2, \gamma_1, \gamma_2) = \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi_1 x \geq \gamma_1 + 1,\ \pi_2 x \geq \gamma_2 + 1\}.$$

Cross disjunctions were first defined and studied by Li and Richard (2008) who call them 2-branch split disjunctions. A linear inequality valid for $P^{LP} \cap C_k(\pi_1, \pi_2, \gamma_1, \gamma_2)$ for $k = 1, \ldots, 4$, is called a *cross cut* for P obtained from the disjunction $C(\pi_1, \pi_2, \gamma_1, \gamma_2)$. As $P \subseteq \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} \subseteq C(\pi_1, \pi_2, \gamma_1, \gamma_2)$, cross cuts are valid for all points in P.

We define $P_C(\pi_1, \pi_2, \gamma_1, \gamma_2)$ as the convex hull of $P^{LP} \cap C(\pi_1, \pi_2, \gamma_1, \gamma_2)$. The *cross closure* of P, denoted by P_C, is the set of points in P^{LP} that satisfy all cross cuts obtained from all possible disjunctions for P. Clearly,

$$P_C = \bigcap_{\pi_1, \pi_2 \in \mathbb{Z}^{n_1}} \bigcap_{\gamma_1, \gamma_2 \in \mathbb{Z}} P_C(\pi_1, \pi_2, \gamma_1, \gamma_2).$$

2.3. Crooked cross disjunctions and crooked cross cuts

Similar to cross disjunctions, the *crooked cross disjunction* (see Dash et al. (2012a)) associated to $\pi_1, \pi_2 \in \mathbb{Z}^{n_1} \setminus \{\mathbf{0}\}$, and $\gamma_1, \gamma_2 \in \mathbb{Z}$ is given by

$$D(\pi_1, \pi_2, \gamma_1, \gamma_2) = \bigcup_{k \in \{1,2,3,4\}} D_k(\pi_1, \pi_2, \gamma_1, \gamma_2)$$

where:

$$\begin{aligned}
D_1(\pi_1, \pi_2, \gamma_1, \gamma_2) &= \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi_1 x \leq \gamma_1,\ (\pi_2 - \pi_1)x \leq \gamma_2 - \gamma_1\},\\
D_2(\pi_1, \pi_2, \gamma_1, \gamma_2) &= \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi_1 x \leq \gamma_1,\ (\pi_2 - \pi_1)x \geq \gamma_2 - \gamma_1 + 1\},\\
D_3(\pi_1, \pi_2, \gamma_1, \gamma_2) &= \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi_1 x \geq \gamma_1 + 1,\ \pi_2 x \leq \gamma_2\},\ \text{and}\\
D_4(\pi_1, \pi_2, \gamma_1, \gamma_2) &= \{(x, y) \in \mathbb{R}^{n_1+n_2} : \pi_1 x \geq \gamma_1 + 1,\ \pi_2 x \geq \gamma_2 + 1\}.
\end{aligned}$$

Notice that D_3 and D_4 above are same as C_3 and C_4 described in Section 2.2 whereas D_1 and D_2 are different from C_1 and C_2. A linear inequality valid for $P^{LP} \cap D_k(\pi_1, \pi_2, \gamma_1, \gamma_2)$ for $k = 1, \ldots, 4$, is called a *crooked cross (CC) cut* for P obtained from the disjunction $D(\pi_1, \pi_2, \gamma_1, \gamma_2)$. Clearly CC cuts are valid for all points in P.

We define $P_{CC}(\pi_1, \pi_2, \gamma_1, \gamma_2)$ to be the convex hull of $P^{LP} \cap D(\pi_1, \pi_2, \gamma_1, \gamma_2)$, and denote the *CC closure* of P by P_{CC} where

$$P_{CC} = \bigcap_{\pi_1, \pi_2 \in \mathbb{Z}^{n_1}} \bigcap_{\gamma_1, \gamma_2 \in \mathbb{Z}} P_{CC}(\pi_1, \pi_2, \gamma_1, \gamma_2).$$

It is easy to see that the CC closure and cross closure of P are contained in its split closure. Further, it is shown in Dash et al. (2011) that $P_{CC} \subseteq P_C$, but the containment is not strict (Dash et al. 2012b).

2.4. Breaking symmetry

We next discuss how to represent the disjunctions discussed above uniquely and how to identify "useless" disjunctions. First note that given $\pi_1, \pi_2 \in \mathbb{Z}^{n_1}$ and $\gamma_1, \gamma_2 \in \mathbb{Z}$, such that $S(\pi_1, \gamma_1) \subset S(\pi_2, \gamma_2)$, any split cut generated from $S(\pi_2, \gamma_2)$ can also be generated from $S(\pi_1, \gamma_1)$. This basic idea easily extends to more general disjunctions. We say that a split (or cross, or crooked cross) disjunction "dominates" a second split (or cross, or crooked cross) disjunction if the first one is strictly contained in the second.

A split disjunction $S(\pi, \gamma)$ is dominated by another split disjunction unless components of π are coprime. In other words, $S(\pi, \gamma)$ is dominated if there exists an integer $k > 1$ such that $\pi/k \in \mathbb{Z}^{n_1}$; in this case $S(\pi/k, \lfloor \gamma/k \rfloor)$ is strictly contained in $S(\pi, \gamma)$. Conversely, as $S(\pi, \gamma)$ is dominated by $S(\pi', \gamma')$ only when the hyperplanes defined by π and π' are parallel to each other, it is easy to show that $S(\pi, \gamma)$ is not contained in another split disjunction if the components of π are coprime. Consequently, it suffices to only consider π that have coprime elements in (1).

It follows that for $\pi \neq \pi'$, $S(\pi, \gamma) = S(\pi', \gamma')$ if and only $\pi' = -\pi$ and $\gamma' = -\gamma - 1$. Notice that exactly one of γ and $-\gamma - 1$ is nonnegative and the other one is strictly negative. Consequently, in the definition of the split closure in (1), one can assume $\gamma \in \mathbb{Z}_+$. We therefore have the following observation.

Observation 1. A split disjunction $S(\pi, \gamma)$ is uniquely defined by its parameters if $\gamma \geq 0$. Furthermore, it is not dominated by another split disjunction if and only if elements of π are coprime.

It is relatively straight forward to extend this observation to cross disjunctions as follows after noticing the fact that $C(\pi_1, \pi_2, \gamma_1, \gamma_2) = C(\pi_2, \pi_1, \gamma_2, \gamma_1)$.

Observation 2. A cross disjunction $C(\pi_1, \pi_2, \gamma_1, \gamma_2)$ is uniquely defined by its parameters if $\gamma_1, \gamma_2 \geq 0$ and $[\pi_1, \gamma_1]$ is greater than $[\pi_2, \gamma_2]$ in the lexicographical sense. Furthermore, it is not dominated by another cross disjunction if elements of $[\pi_1, \pi_2]$ are coprime.

When the elements of the vector $[\pi_1, \pi_2] \in \mathbb{Z}^{2n_1}$ are divisible by a common integer $k > 1$, then it is easy to show that $D(\pi_1, \pi_2, \gamma_1, \gamma_2)$ is dominated by $D((1/k)\pi_1, (1/k)\pi_2, \lfloor \gamma_1/k \rfloor, \lfloor \gamma_2/k \rfloor)$. However, due to the asymmetry in the description of crooked cross disjunctions, $D(\pi_1, \pi_2, \gamma_1, \gamma_2) = D(\pi_1', \pi_2', \gamma_1', \gamma_2')$ only when $\pi_1 = \pi_1', \pi_2 = \pi_2', \gamma_1 = \gamma_1'$ and $\gamma_2 = \gamma_2'$ and consequently these disjunctions are uniquely defined by the associated parameters. To see this, consider the crooked cross disjunction $D([1, 0], [0, 1], 0, 0)$

for $n_1 = 2$ as shown in Figure 1. Assume that there exists $\pi_1, \pi_2 \in \mathbb{Z}^2$ and $\gamma_1, \gamma_2 \in \mathbb{Z}$ such that $D(\pi_1, \pi_2, \gamma_1, \gamma_2) = D([1,0],[0,1],0,0)$. It is easy to see that either $\pi_1 = [1,0]$, $\gamma_1 = 0$, or, $\pi_1 = [-1,0]$, $\gamma_1 = -1$. Furthermore, if $\pi_1 = [1,0]$ clearly $\pi_2 = [0,1]$ and $(\pi_1, \pi_2, \gamma_1, \gamma_2) = ([1,0],[0,1],0,0)$. On the other hand, when $\pi_1 = [-1,0]$ and $\gamma_1 = -1$, it is possible to show that there does not exist π_2, γ_2 such that $D(\pi_1, \pi_2, \gamma_1, \gamma_2) = D([1,0],[0,1],0,0)$ holds. We therefore make the following observation.

Observation 3. A crooked cross disjunction $D(\pi_1, \pi_2, \gamma_1, \gamma_2)$ is uniquely defined by its parameters. Furthermore, it is not dominated by another crooked cross disjunction if the elements of the extended vector $[\pi_1, \pi_2]$ are coprime.

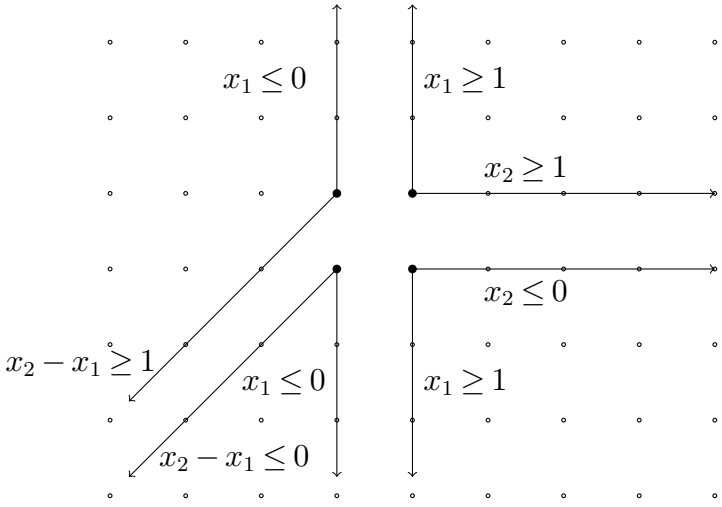


Figure 1 **A crooked cross disjunction defined by** $\pi_1 = [1,0]$, $\gamma_1 = 0$ **and** $\pi_2 = [0,1]$, $\gamma_2 = 0$

Given a pair of split disjunctions $S(\pi_1, \gamma_1)$ and $S(\pi_2, \gamma_2)$, we get only one "natural" cross disjunction by intersecting these two split disjunctions, which is $C(\pi_1, \pi_2, \gamma_1, \gamma_2)$. However, we can construct eight "natural" crooked cross disjunctions; we get different crooked cross disjunctions by switching (π_1, γ_1) and (π_2, γ_2), and by separately multiplying these vectors by ± 1.

3. Separating cross and crooked cross cuts

Given a mixed integer set $P = \{(x,y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} : Ax + Gy = b,\ x \geq 0,\ y \geq 0\}$, and a point $(\bar{x}, \bar{y}) \in P^{LP}$, it is easy to write a linear program to separate $(\bar{x}, \bar{y})$ from $P_C(\pi_1, \pi_2, \gamma_1, \gamma_2)$ if $\pi_1, \pi_2 \in \mathbb{Z}^{n_1}$, and $\gamma_1, \gamma_2 \in \mathbb{Z}$ are fixed. More precisely, a violated cross cut of the form $ax + cy \geq d$ (here a and c are row vectors and d is a number), if it exists, can be obtained

by solving the following linear program.

Cross Cut Separation LP:

$$\min \quad z = a\bar{x} + c\bar{y} - d \tag{5a}$$

subject to

$$a \geq \lambda_1 A - \alpha_1\pi_1 - \beta_1\pi_2, \qquad d \leq \lambda_1 b - \alpha_1\gamma_1 - \beta_1\gamma_2 \tag{5b}$$

$$a \geq \lambda_2 A - \alpha_2\pi_1 + \beta_2\pi_2, \qquad d \leq \lambda_2 b - \alpha_2\gamma_1 + \beta_2(\gamma_2+1) \tag{5c}$$

$$a \geq \lambda_3 A + \alpha_3\pi_1 - \beta_3\pi_2, \qquad d \leq \lambda_3 b + \alpha_3(\gamma_1+1) - \beta_3\gamma_2 \tag{5d}$$

$$a \geq \lambda_4 A + \alpha_4\pi_1 + \beta_4\pi_2, \qquad d \leq \lambda_4 b + \alpha_4(\gamma_1+1) + \beta_4(\gamma_2+1) \tag{5e}$$

$$c \geq \lambda_i G \qquad \forall i \in \{1,2,3,4\} \tag{5f}$$

$$\beta_i, \alpha_i \geq 0, \qquad \forall i \in \{1,2,3,4\} \tag{5g}$$

$$\lambda_i \text{ free}, \qquad \forall i \in \{1,2,3,4\} \tag{5h}$$

$$a, c, d \text{ free}. \tag{5i}$$

$$\sum_{i=1}^{4} (\|\lambda_i\|_1 + \alpha_i + \beta_i) \leq \quad 1. \tag{5j}$$

Here $\lambda_1, \ldots, \lambda_4$ are row vectors with m components, and α_i and β_i are real numbers.

As $\mathbf{0}$ is a feasible solution to the cross cut separation LP, the optimal value $z^* \leq 0$. Note that if there exists a solution to the separation LP with $z < 0$, then the cross cut $ax + cy \geq d$ associated with this solution is violated by $(\bar{x}, \bar{y})$ and conversely, if there exists a violated cross cut, then there is a corresponding solution to the LP with $z < 0$. This implies that the optimal value $z^* = 0$ if and only if $(\bar{x}, \bar{y}) \in P_C(\pi_1, \pi_2, \gamma_1, \gamma_2)$.

Observe that the constraints (5b)-(5i) define a cone, and thus the objective function value of the cross cut separation LP without the *normalization constraint* (5j) is unbounded (when a violated cut exists). The normalization constraint makes the problem bounded while preserving all valid cuts up to scalar multiplication (see Fischetti et al. (2011), Balas and Bonami (2009)). Many other normalization constraints can be found in the literature, see Balas and Bonami (2009). We experimented with the three normalizations (a), (b), (c) mentioned in (Balas and Bonami 2009, p. 185) as well as a variant where we use l_∞ norm instead of l_1 or l_2 as the weight of row multipliers. We ran our code with each of these

normalizations and chose (a), the *unweighted* version in Balas and Bonami (2009) which yielded the best average gap closed after one hour.

We now describe the separation algorithm we used in our computational experiments. The procedure receives as input a mixed integer set P, a list of cross disjunctions C, a point $(\bar{x}, \bar{y}) \in P^{LP}$ to be separated and the maximum number of cuts p to be returned. The procedure then attempts to separate $(\bar{x}, \bar{y}) \in P^{LP}$ for each cross disjunction in C.

Algorithm 1: Cross.LP$(C, P, (\bar{x}, \bar{y}), p)$.

Input: List of cross disjunctions C, mixed integer set P, point $(\bar{x}, \bar{y}) \in P^{LP}$ and list size p.

Result: List of cuts L.

1 Set $L = \emptyset$.
2 **for** $(\pi^1, \gamma^1, \pi^2, \gamma^2) \in C$ **do**
3 Solve the cross cut separation LP for $P_C(\pi^1, \pi^2, \gamma^1, \gamma^2)$.
4 **if** *a violated cut is found* **then** add it to the list L. **if** $|L| \geq p$ **then** exit the for loop.
5 **end**

3.1. Separating split cuts

Split cuts derived from a fixed split disjunction $S(\pi, \gamma)$ can be separated in a similar fashion by solving the *split cut separation LP* which we define by replacing the constraints (5b)-(5e) by

$$a \geq \lambda_1 A - \alpha_1 \pi, \qquad d \leq \lambda_1 b - \alpha_1 \gamma$$
$$a \geq \lambda_2 A + \alpha_2 \pi, \qquad d \leq \lambda_2 b + \alpha_2 (\gamma + 1)$$

and the number 4 by 2 in the constraints (5f)-(5h) and (5j).

The separation algorithm we used in our computational experiments is called **Split.LP**$(S, P, (\bar{x}, \bar{y}), p)$ and it is very similar to that of cross cuts above. The only difference is that the input to the algorithm consists of a list of split disjunctions and the split cut separation LP is solved for each disjunction to find violated cuts.

3.2. Separating crooked cross cuts

For a given fixed crooked cross disjunction $D(\pi_1, \pi_2, \gamma_1, \gamma_2)$, the associated separation problem can again be formulated as a linear program. The resulting LP is identical to (5a)-(5i)

except constraints (5b) and (5c), associated with the first two disjunctions are replaced with

$$a \geq \lambda_1 A - \alpha_1 \pi_1 - \beta_1(\pi_2 - \pi_1), \qquad d \leq \lambda_1 b - \alpha_1 \gamma_1 - \beta_1(\gamma_2 - \gamma_1) \tag{6}$$

$$a \geq \lambda_2 A - \alpha_2 \pi_1 + \beta_2(\pi_2 - \pi_1), \qquad d \leq \lambda_2 b - \alpha_2 \gamma_1 + \beta_2(\gamma_2 - \gamma_1 + 1). \tag{7}$$

The resulting LP is called the *crooked cross cut separation LP* and it produces a violated cut provided that there is one.

We call the separation routine for crooked cross cuts $\mathbf{CCross.LP}(C, P, (\bar{x}, \bar{y}), p)$, where C stands for a list of cross disjunctions. The algorithm is essentially identical to that Cross.LP except that the crooked cross cut separation LP is solved to identify cuts; as discussed earlier there are eight distinct crooked cross disjunctions that can be defined from a cross disjunctions, and each one is tested for a violated cut in CCross.LP.

3.3. A bilinear integer program for separating from the (crooked) cross cut closure

If we let $\pi_1, \pi_2, \gamma_1, \gamma_2$ be variables in (5a)-(5i) we obtain a bilinear mixed integer separation problem for cross cuts given by (5a)- (5j), and

$$\pi_1, \pi_2 \in \mathbb{Z}^{n_1}, \ \gamma_1, \gamma_2 \in \mathbb{Z}. \tag{8}$$

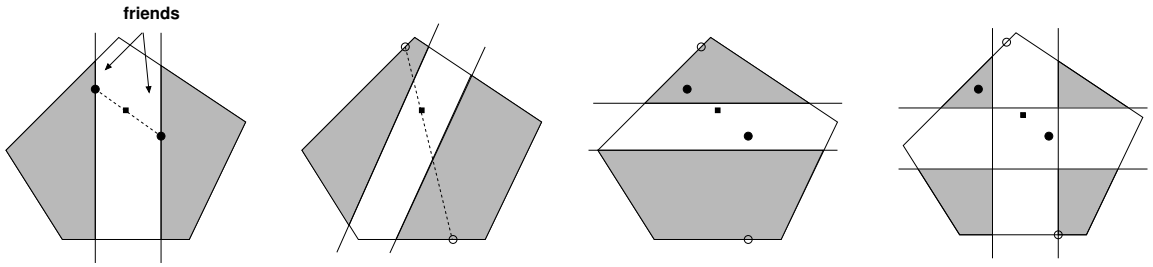
This separation problem will automatically select a cross disjunction from all possible cross disjunctions. A similar bilinear separation problem can be formulated for crooked cross cuts as well. Unfortunately, in both cases the bilinear problem is significantly harder to solve than the original separation problem as it contains integer variables as well as non-convex constraints. Still, the same can be said of the bilinear program for split cuts introduced in Dash et al. (2010), Balas and Saxena (2008) and these later formulations proved useful when (approximately) solved with specialized techniques. Developing similar techniques for the bilinear mixed integer separation problem for the cross closure or crooked cross closure is possible but it is beyond the scope of this paper.

3.4. When the separation LP fails

For a given disjunction, if the associated separation LP fails to produce a cut violated by the point $\bar{p} = (\bar{x}, \bar{y})$, useful information can be extracted from the optimal solution of the dual of the LP. For simplicity, consider a split disjunction $S(\pi, \gamma) = S_1(\pi, \gamma) \cup S_2(\pi, \gamma)$ such that $\bar{p} \notin S(\pi, \gamma)$ and assume that $\bar{p} \in P_S(\pi, \gamma)$. In this case, for some $1 > \mu > 0$

$$\bar{p} = \mu p^1 + (1 - \mu) p^2 \quad \text{ such that } \quad p^i \in P^{LP} \cap S_i(\pi, \gamma), \ \text{ for } \ i = 1, 2.$$

We refer to points p^1, p^2 as a pair of *friends* of $\bar{p}$. From a computational point of view, this information can be very useful when selecting a new disjunction from among a list of candidate disjunctions to find cuts separating the same point $\bar{p}$. More precisely consider a second disjunction $S(\pi', \gamma')$ not containing $\bar{p}$. If both p^1, p^2 obtained from the previous disjunction belong to $S(\pi', \gamma')$, then clearly $\bar{p} \in P_S(\pi', \gamma')$, and therefore no violated split cut can be derived from $S(\pi', \gamma')$. We note that a similar idea is developed and used independently by Fischetti and Salvagnin (2011). We discuss how we exploit this idea further in Section 6.



(a) (b) (c) (d)

Figure 2 Friends of a point.

The idea of friends is illustrated in Figure 2. For simplicity, this figure shows the projection onto a two dimensional space that contains the π's for all the split disjunctions considered. In Figure 2(a), we depict a pair of friends of $\bar{p}$ obtained from the displayed disjunction, say $S(\pi, \gamma)$. The polygon represents P^{LP}, the pair of parallel vertical lines represent the hyperplanes $\pi x = \gamma$ and $\pi x = \gamma + 1$ that define the disjunction and the shaded regions represent the atoms of the disjunction intersected with P^{LP}. The point $\bar{p}$ is represented by a filled square, and the friends are represented by filled circles. In Figure 2(b), we depict a pair of "good" friends of $\bar{p}$ (defined below) obtained from a different disjunction. In Figure 2(c), we depict a disjunction by a pair of horizontal lines, and show the two pairs of friends of $\bar{p}$ from Figures 2(a),(b). One pair does not belong to the disjunction, but the other pair does and consequently no violated split cut can be obtained from this disjunction. Figure 2(d) shows a cross disjunction that excludes one friend from each disjunction and therefore has the potential of producing violated cuts.

We next discuss how to generate a pair of friends from the separation LP when it does not produce a violated cut. Remember that the normalization constraint (5j) requires taking

the absolute value of the λ variables and therefore in the reformulation of the split cut separation LP below, we use two sets of nonnegative variables. Let $(\bar{x}, \bar{y}) \in P^{LP}$ denote the point to be separated.

$$\min \quad z = a\bar{x} + c\bar{y} - d \tag{9}$$

subject to

$$a \geq \lambda^{1,+}A - \lambda^{1,-}A - \alpha_1\pi, \qquad d \leq \lambda^{1,+}b - \lambda^{1,-}b - \alpha_1\gamma \tag{10}$$

$$a \geq \lambda^{2,+}A - \lambda^{2,-}A + \alpha_2\pi, \qquad d \leq \lambda^{2,+}b - \lambda^{2,-}b + \alpha_2(\gamma+1) \tag{11}$$

$$c \geq \lambda^{i,+}G - \lambda^{i,+}G \qquad \text{for } i = \{1,2\} \tag{12}$$

$$\lambda^{i,+}, \lambda^{i,-}, \beta_i, \alpha_i \geq 0, \qquad \text{for } i = \{1,2\} \tag{13}$$

$$a, c, d \text{ free.} \tag{14}$$

$$\sum_{i=1}^{2}(\alpha_i + \beta_i + \sum_{j=1}^{m}(\lambda_j^{i,+} + \lambda_j^{i,-})) \leq 1 + n_1 + n_2. \tag{15}$$

Note that when $(\bar{x}, \bar{y}) \in P_S(\pi, \gamma)$, the optimal solution to the separation LP has $z = 0$. Now consider the dual of this LP which also has an optimal value of 0.

$$\max \quad (1 + n_1 + n_2)\eta \tag{16}$$

subject to

$$Au^i + Gv^i \leq \omega_i b - \mathbf{1}\eta \qquad \text{for } i = 1,2 \tag{17}$$

$$-Au^i - Gv^i \leq -\omega_i b - \mathbf{1}\eta \qquad \text{for } i = 1,2 \tag{18}$$

$$-\pi u^1 \leq -\omega_1\gamma - \eta \tag{19}$$

$$\pi u^2 \leq \omega_2(\gamma+1) - \eta \tag{20}$$

$$-u^1 - u^2 = \bar{x} \tag{21}$$

$$-v^1 - v^2 = \bar{y} \tag{22}$$

$$\omega_1 + \omega_2 = -1 \tag{23}$$

$$u^i, v^i, \omega_i, \eta \leq 0 \qquad \text{for } i = 1,2 \tag{24}$$

Here, for $i = 1,2$, $u^i \in \mathbb{R}^{n_1}$, $v^i \in \mathbb{R}^{n_2}$, and $\eta, \omega_i \in \mathbb{R}$.

Let $(\hat{u}^1, \hat{u}^2, \hat{v}^1, \hat{v}^2, \hat{\omega}^1, \hat{\omega}^2, \hat{\eta})$ be an optimal dual solution. As the optimal objective value of the dual problem equals 0, we have $0 = (1 + n_1 + n_2)\hat{\eta}$ and therefore $\hat{\eta} = 0$. Therefore, constraints (17) and (18) imply that $A\hat{u}^i + G\hat{v}^i = \omega_i b$ for $i = 1, 2$. First assume that $\hat{\omega}_1, \hat{\omega}_2 < 0$, and let

$$\bar{u}^i = \hat{u}^i/\hat{\omega}_i \quad \text{and} \quad \bar{v}^i = \hat{v}^i/\hat{\omega}_i$$

Note that $(\bar{u}^i, \bar{v}^i) \geq 0$ and $A\bar{u}^i + G\bar{v}^i = b$ and therefore $(\bar{u}^i, \bar{v}^i) \in P^{LP}$ for $i = 1, 2$. Next, notice that $(\bar{x}, \bar{y})$ is a convex combination of $(\bar{u}^1, \bar{v}^1)$ and $(\bar{u}^2, \bar{v}^2)$ as (21)–(24) imply that

$$\bar{x} = (-\hat{\omega}_1)\bar{u}^1 + (-\hat{\omega}_2)\bar{u}^2 \quad \text{and} \quad \bar{y} = (-\hat{\omega}_1)\bar{v}^1 + (-\hat{\omega}_2)\bar{v}^2. \tag{25}$$

Finally, due to (19) and (20), we have $(\bar{u}^i, \bar{v}^i) \in S_i(\pi, \gamma)$ and consequently, the points $(\bar{u}^i, \bar{v}^i)$ for $i = 1, 2$ give a pair of friends for $(\bar{x}, \bar{y})$.

Next consider the case when one of $\hat{\omega}_1, \hat{\omega}_2$ is zero. Without loss of generality, let $\hat{\omega}_1 = -1$ and $\hat{\omega}_2 = 0$. Let $\bar{u} = -\hat{u}^1$, $\bar{v} = -\hat{v}^1$ and $d_u = -\hat{u}^2$, $d_v = -\hat{v}^2$ and notice that by (17), (18) and (24), $(\bar{u}, \bar{v}) \in P^{LP}$ and (d_u, d_v) belongs to the recession cone of P^{LP}. Also notice that $\pi\bar{u} \leq \gamma$ and $\pi d_u \geq 0$ by (19) and (20), respectively. Furthermore, as $\bar{x} = \bar{u} + d_u$ by (21), and as $\pi\bar{x} > \gamma$ we have $\pi d_u > 0$. Therefore, $\pi(\bar{u} + \alpha d_u) \geq \gamma + 1$ for some $\alpha > 0$ and points $(\bar{u}, \bar{v})$ and $(\bar{u}, \bar{v}) + \alpha(d_u, d_v)$ give a pair of friends for $(\bar{x}, \bar{y})$.

Therefore if the split cut separation LP fails to produce a violated cut, then it is possible to produce a pair of friends of $\bar{p}$, namely, $p^1, p^2 \in P^{LP}$ such that $\bar{p} = (1 - \mu)p^1 + \mu p^2$ for some μ with $0 < \mu < 1$. In other words,

$$p^2 = p^1 + \frac{1}{\mu}(\bar{p} - p^1).$$

Now consider a point $p' = p^1 + \theta(\bar{p} - p^1)$ such that $\theta > 1/\mu$. If $p' \in P^{LP}$, then clearly p^1 and p' also form a pair of friends for $\bar{p}$. Moreover, from a computational point of view, this new pair is more useful than p^1, p^2 for checking if disjunctions other than $S(\pi, \gamma)$ have the potential to yield violated cuts. Assume $\bar{p} \notin S(\pi', \gamma') \neq S(\pi, \gamma)$. If $p^1, p^2 \in S(\pi', \gamma')$ (implying that $\bar{p} \in P_S(\pi', \gamma')$), then $p' \in S(\pi', \gamma')$. Consequently, it is best to find the point $\bar{p}^2 = p^1 + \theta(\bar{p} - p^1) \in P^{LP}$ such that θ is maximized. Similarly, one can find the point $\bar{p}^1 = p^2 + \theta(\bar{p} - p^2) \in P^{LP}$ such that θ is maximized. We call the new pair of friends $\bar{p}^1$ and $\bar{p}^2$ *good* friends of $\bar{p}$; see Figure 2(b).

It is possible to extend these ideas to separation LPs other than the split cut separation LP but the analysis is more tedious and as we discuss later in Section 6, we implemented this idea only for split cut separation.

4. Separating cross cuts using rank-2 split cuts

In this section we describe separation heuristics to find violated cross cuts. Solving the separation problem exactly for split cuts is hard for practical instances (Balas and Saxena 2008, Dash et al. 2010) and we expect that finding violated cross or crooked cross cuts will also be very hard. For computational tractability, we do not attempt to generate a cross disjunction from scratch but instead focus on extending existing split cut heuristics to the cross cut setting. To achieve this we use the following proposition, which shows that certain rank-2 split cuts are also cross cuts. We note that not all rank-2 split cuts are cross cuts and these two cut families do not dominate each other (Dash et al. 2012b).

PROPOSITION 1. *Let $\pi_1 \in \mathbb{Z}^{n_1}$ and $\gamma_1 \in \mathbb{Z}$. Any split cut for $P_S(\pi_1, \gamma_1)$ is a cross cut for P. Furthermore, if a split cut for $P_S(\pi_1, \gamma_1)$ is obtained from the split disjunction $S(\pi_2, \gamma_2)$, then the same cut is a cross cut for P obtained from the cross disjunction $C(\pi_1, \pi_2, \gamma_1, \gamma_2)$.*

Let $ax + cy \geq d$ be a split cut for $P_S(\pi_1, \gamma_1)$ obtained from the split disjunction $S(\pi_2, \gamma_2)$. We will call $ax + cy \geq d$ simply "the cut" in the rest of the proof. By definition, the cut is valid for both $P_S(\pi_1, \gamma_1) \cap S_1(\pi_2, \gamma_2)$ and $P_S(\pi_1, \gamma_1) \cap S_2(\pi_2, \gamma_2)$ where $S_1(\pi_2, \gamma_2)$ and $S_2(\pi_2, \gamma_2)$ are the two half-spaces that define the disjunction $S(\pi_2, \gamma_2)$ as defined in Section 2.1. In addition, note that

$$P^{LP} \cap S_1(\pi_1, \gamma_1),\ P^{LP} \cap S_2(\pi_1, \gamma_1) \subseteq P_S(\pi_1, \gamma_1)$$

as $P_S(\pi_1, \gamma_1)$ is the convex hull of the union of these two sets.

Clearly, as the cut is valid for $P_S(\pi_1, \gamma_1) \cap S_1(\pi_2, \gamma_2)$, it is valid for its subsets

$$P^{LP} \cap S_1(\pi_1, \gamma_1) \cap S_1(\pi_2, \gamma_2) \text{ and } P^{LP} \cap S_2(\pi_1, \gamma_1) \cap S_1(\pi_2, \gamma_2).$$

Similarly, the cut is also valid for

$$P^{LP} \cap S_1(\pi_1, \gamma_1) \cap S_2(\pi_2, \gamma_2) \text{ and } P^{LP} \cap S_2(\pi_1, \gamma_1) \cap S_2(\pi_2, \gamma_2).$$

To conclude the proof, it is sufficient to observe that

$$S_i(\pi_1, \gamma_1) \cap S_j(\pi_2, \gamma_2) = C_{2i+j-2}(\pi_1, \pi_2, \gamma_1, \gamma_2)$$

for $i, j \in \{1, 2\}$, which shows that the cut is valid for $P^{LP} \cap C_k(\pi_1, \pi_2, \gamma_1, \gamma_2)$ for $k \in \{1, 2, 3, 4\}$.

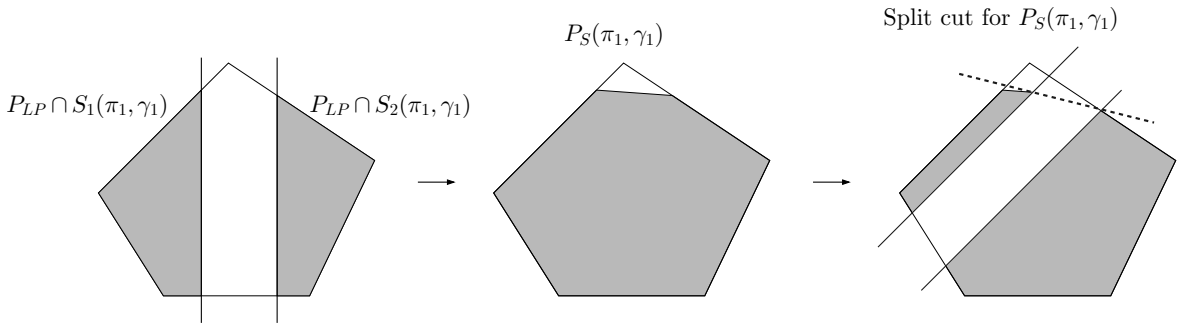


Figure 3 **Rank-2 split cuts which are also cross cuts.**

In Figure 3 we show an example demonstrating how Proposition 1 works. The first picture shows $P^{LP} \cap S(\pi_1, \gamma_1)$, and the second picture shows its convex hull $P_S(\pi_1, \gamma_1)$. The last picture shows a split cut for $P_S(\pi_1, \gamma_1)$ which is also a cross cut for P.

We next shows that some known rank-2 split cuts, called 2-step MIR cuts, are also cross cuts. Dash and Günlük (2006) study the following simple mixed-integer set

$$Q = \big\{ y \in \mathbb{R},\ x_1, x_2 \in \mathbb{Z} : y + \alpha x_1 + x_2 \ \geq\ \beta,\ \ y, x_1 \geq 0 \big\}.$$

where $\beta, \alpha \in \mathbb{R}$ and $1 > \beta > \alpha > 0$, and show that

$$\big(1/(\beta - \alpha \lfloor \beta/\alpha \rfloor)\big) y + x_1 + \lceil \beta/\alpha \rceil x_2 \geq \lceil \beta/\alpha \rceil \tag{26}$$

is valid (and facet defining) for Q provided that $1/\alpha \geq \lceil \beta/\alpha \rceil > \beta/\alpha$. The validity proof in Dash and Günlük (2006) essentially shows that the 2-step MIR inequality (26) is an MIR inequality (or, split cut) for the set $Q' \subseteq Q$ obtained by strengthening the original set with the simple MIR inequality

$$y + \alpha x_1 + \beta x_2 \ \geq\ \beta,$$

obtained using the disjunction $\{x : x_2 \leq 0\} \cup \{x : x_2 \geq 1\}$. As inequality (26) is a split cut for $Q' \supseteq Q_S([0,1], 0)$, by Proposition 1, it is a cross cut for the original set Q.

We next present two heuristics that separate cross cuts that are rank-2 split cuts. We begin with a MIP-based heuristic which yields effective cuts but it is time consuming. The second heuristic is based on a split cut heuristic by Dash and Goycoolea (2010) and as it is LP-based, it is significantly faster.

4.1. An MIP based heuristic when one of the disjunctions is fixed

As discussed in Section 2.1, given $(\bar{x}, \bar{y}) \in P^{LP}$, finding a violated split cut is equivalent to finding an implied equation of the form (2), such that the associated MIR cut (3) is violated by $(\bar{x}, \bar{y})$. This is a difficult computational task. Notice that when $\hat{\beta} \neq 0$, the MIR cut (3) can also be rewritten as

$$\sum_{i=1}^{n_1} (\lfloor a_i \rfloor + \min(\frac{\hat{a}_i}{\hat{\beta}}, 1)) x_i + \frac{1}{\hat{\beta}} \sum_{i=1}^{n_2} \max(c_i, 0) y_i \geq \lceil \beta \rceil, \tag{27}$$

for a given base inequality $ax + cy \geq \beta$ that is implied by P^{LP}. If this base inequality is instead implied by $P_S(\pi_1, \gamma_1)$ for some split disjunction $S(\pi_1, \gamma_1)$, the MIR inequality (27) is now a rank-2 split cut for P^{LP} that is also a cross cut. Instead of explicitly constructing $P_S(\pi_1, \gamma_1)$ – which can be quite difficult – we use the fact that an inequality is implied by $P_S(\pi_1, \gamma_1)$ if and only if it is implied both by $P^{LP} \cap S_1(\pi_1, \gamma_1)$ and $P^{LP} \cap S_2(\pi_1, \gamma_1)$. We can then easily characterize the required implied inequalities in a manner similar to the characterization of split cuts (as being valid for two polyhedra). However, separating an MIR inequality (27) based on an inequality that is valid for two polyhedra is computationally difficult. We therefore instead use the weakening of (27) given by

$$\sum_{i=1}^{n_1} \lceil a_i \rceil x_i + \frac{1}{\varepsilon} \sum_{i=1}^{n_2} \max(c_i, 0) y_i \geq \lceil \beta \rceil, \tag{28}$$

for any positive $\epsilon \leq \hat{\beta}$. This *approximate MIR* cut is also a split cut obtained from the disjunction (4). Also notice that inequality (28) becomes the so-called pro-CG cut (see Bonami et al. (2008)) when all c_i are non-positive and therefore the second summation in inequality (28) vanishes. Below we write an MIP model for separating this approximate MIR cut. Clearly, the existence of a violated MIR cut does not guarantee the existence of a violated approximate MIR cut. However, our MIP model always returns a base inequality from which an MIR cut can be derived. Additionally, the associated split disjunction can be used to obtain other split cuts using a separation LP (we note that this is the underlying idea used in Balas and Saxena (2008) to separate split cuts). Writing this MIP for a base inequality implied by $P^{LP} \cap S_1(\pi_1, \gamma_1)$ and $P^{LP} \cap S_2(\pi_1, \gamma_1)$ yield the following rank-2 split cut separation MIP where $\epsilon \in (0, 1)$ is a given constant and $\tilde{a}$ and $\tilde{c}$ are unknown row vectors with n_1 and n_2 components respectively, and $\tilde{b}$ is an unknown number.

$$\min \quad z = \tilde{a}\bar{x} + \frac{1}{\varepsilon}\tilde{c}\bar{y} - (\tilde{b} + 1) \tag{29a}$$

subject to:

$$\tilde{a} \geq \lambda_1 A + \alpha_1 \pi, \tag{29b}$$

$$\tilde{a} \geq \lambda_2 A - \alpha_2 \pi, \tag{29c}$$

$$\tilde{c} \geq \lambda_1 G \tag{29d}$$

$$\tilde{c} \geq \lambda_2 G \tag{29e}$$

$$\tilde{b} + \varepsilon \leq \lambda_1 b + \alpha_1 \gamma \tag{29f}$$

$$\tilde{b} + \varepsilon \leq \lambda_2 b - \alpha_2 (\gamma + 1) \tag{29g}$$

$$\tilde{c}, \alpha_1, \alpha_2 \geq 0, \qquad \tilde{a}, \tilde{b}, \lambda_1, \lambda_2 \text{ free} \tag{29h}$$

$$\tilde{a} \in \mathbb{Z}^{n_1}, \tilde{b} \in \mathbb{Z}. \tag{29i}$$

Consider a feasible solution for this model. Let $a \in \mathbb{R}^{n_1}$, $c \in \mathbb{R}^{n_2}$ be vectors defined by $c = \max\{\lambda_1 G, \lambda_2 G\}$ and $a = \max\{\lambda_1 A, \lambda_2 A\}$ with the maximum being taken componentwise. Let $\beta = \tilde{b} + \varepsilon \leq \min\{\lambda_1 b + \alpha_1 \gamma, \lambda_2 b - \alpha_2(\gamma + 1)\}$. Then $ax + cy \geq \beta$ is a valid inequality for $P^{LP} \cap S_1(\pi, \gamma)$ and $P^{LP} \cap S_2(\pi, \gamma)$. As $\lceil \beta \rceil = \tilde{b} + 1$ and $\beta - \lfloor \beta \rfloor \geq \varepsilon$,

$$\tilde{a}x + \frac{1}{\varepsilon}\tilde{c}y \geq \tilde{b} + 1 \tag{30}$$

is an approximate MIR cut derived from $ax + cy \geq \beta$. If the associated disjunction is $S(\pi', \gamma')$, then (30) is a cross cut for the cross disjunction $C(\pi, \pi', \gamma, \gamma')$. Furthermore, once the base inequality is $ax + cy \geq \beta$ is obtained, we can write the actual MIR inequality (27) using it. Additionally, even if the MIR inequality is not violated, we can still use the separation LP for $C(\pi, \pi', \gamma, \gamma')$ to obtain additional cross cuts.

We next present the actual heuristic we used in our computational study. As this MIP-based heuristic is computationally expensive, we do not actually run it to find rank-2 MIR cuts, instead, we use the split disjunction identified by it to separate cross cuts using the heuristic Cross.LP.

Algorithm 2: Cross.MIP$(S, P, (\bar{x}, \bar{y}), p)$.

Input: List of split disjunctions S, a mixed integer set P, point $(\bar{x}, \bar{y}) \in P^{LP}$ and list size p.

Result: List of cuts L and list of cross disjunctions D.

Set $L = \emptyset, D = \emptyset$.
for $(\pi, \gamma) \in S$ **do**
  Use the MIP heuristic to find the most violated approximate MIR cut for $P_S^{IP}(\pi, \gamma)$.
  Obtain the associated split disjunction (π', γ').
  Set $L = L \cup \text{Cross.LP}\left(\{(\pi, \gamma, \pi', \gamma')\}, P, (\bar{x}, \bar{y}), 1\right)$.
  Set $D = D \cup \{(\pi, \gamma, \pi', \gamma')\}$.
  if $|L| \geq p$ **then** exit the for loop.
end

4.2. A fast LP based heuristic

Our next procedure generates split cuts for an approximation of $P_S(\pi, \gamma)$ instead of $P_S(\pi, \gamma)$ itself. Remember that $P_S(\pi, \gamma)$ denotes the convex hull of $P^{LP} \cap S(\pi, \gamma)$ and as such its linear description can be obtained by adding all possible split cuts that can be generated by the disjunction $S(\pi, \gamma)$ to P^{LP}. If $\tilde{P}_S(\pi, \gamma)$ is a relaxation of $P_S(\pi, \gamma)$, any split cut for $\tilde{P}_S(\pi, \gamma)$ is a split cut for $P_S(\pi, \gamma)$ and therefore a cross cut for P.

In this heuristic, we obtain the set $\tilde{P}_S(\pi, \gamma)$ by using the split cuts that have been generated earlier in the computation. Every time a split cut is generated by one of our procedures, we keep track of the split disjunction used to generate the cut and associate cuts with disjunctions. An approximation of $P_S(\pi, \gamma)$ can then be easily constructed by adding to P^{LP} all cuts that are associated with $S(\pi, \gamma)$. We then generate split cuts for $\tilde{P}_S(\pi, \gamma)$ using the rank-1 GMI heuristic of Dash and Goycoolea (2010) (referred to henceforth as the DG heuristic), which is an LP-based heuristic that finds violated split cuts using simplex tableau rows associated with feasible and infeasible bases. Given a point to be separated, their heuristic constructs a basis and selects violated rank-1 GMI cuts from the corresponding tableau rows which, as discussed earlier, are split cuts. They show that this approach produces cuts that give a reasonably good approximation of the split closure for problems in MIPLIB 3.0. Subsequent papers by Fischetti and Salvagnin (2011) and Bonami (2012) confirm this observation and give faster heuristics to find such cuts.

We next present the separation algorithm that we used in our computational study. The input of the algorithm includes a list of triplets (π, γ, K) where (π, γ) is a disjunction and K is a list of split cuts associated with this disjunction. We obtain $\tilde{P}_S(\pi, \gamma)$ by adding all the cuts in K to P^{LP}. As this is a fast heuristic, we do not always run the Cross.LP separation unlike the Cross.MIP heuristic described earlier. However, we still keep track of pairs of split disjunctions (or cross disjunctions) that lead to violated inequalities for possible future use. Every time a rank-1 GMI cut is found for $\tilde{P}_S(\pi, \gamma)$, we identify the associated disjunction (π', γ') and save the cross disjunction $(\pi, \gamma, \tilde{\pi}, \tilde{\gamma})$ in a list of *good cross disjunctions* for later calls to Cross.LP. In our preliminary computational experiments, we realized that cut generation time increases significantly as more and more disjunctions are considered. Consequently, in the final version of this separation heuristic, we decided to limit the total number of disjunctions for which we add cuts instead of limiting the total number of cuts.

Algorithm 3: Cross.DG$(S^+, P, (\bar{x}, \bar{y}), R)$.

Input: List of split disjunctions with associated cuts S^+, a mixed integer set P, point to separate $(\bar{x}, \bar{y}) \in P^{LP}$ and rounds of cuts R.

Result: List of cuts L and cross disjunctions D.

Set $L = \emptyset$, $D = \emptyset$ and $r = 0$.
for $(\pi, \gamma, K) \in S^+$ **do**
  Set $\tilde{P}_S(\pi, \gamma) = P \cap K$. `/* Add cuts in K to P */`
  Use the DG heuristic to find violated rank-1 GMI cuts for $\tilde{P}_S(\pi, \gamma)$.
  Let $\{(c^j, \pi^j, \gamma^j)\}_{j \in J}$ be the cuts with associated split disjunctions.
  if $|J| > 0$ **then**
    Set $L = L \cup \{c^j\}_{j \in J}$.
    Set $D = D \cup \{(\pi, \gamma, \pi^j, \gamma^j)\}_{j \in J}$.
    Set $r = r + 1$.
  end
  if $r \geq R$ **then** exit the for loop.
end

5. Cross cuts from two tableau rows

In this section we report on our computational experiments with cross and crooked cuts derived from two-row relaxations that are obtained by taking pairs of optimal simplex tableau rows of the LP relaxation of the MIP. Our aim here is to establish that cuts obtained from such two row relaxations are indeed stronger than cuts that can be obtained from one row relaxations defined by single tableau rows.

Previous experiments (Fukasawa and Goycoolea 2011) show that GMI cuts obtained from the optimal tableau rows give the same objective bound as all cuts that can be obtained from one row relaxations that are defined by a tableau row plus the integrality and bound information on the variables. In our experiments we observe that cuts from two tableau rows increase the lower bound on the objective function value significantly on the same problem set.

Our approach is different from that of Dey et al. (2010) who also experiment with cuts obtained from pairs of tableau rows. The computational approach in Dey et al. (2010) can be summarized as follows:

(i) construct the two-row continuous group relaxation from a given pair of tableau rows by relaxing the integrality of non-basic integral variables,

(ii) find violated 2D lattice-free cuts, in particular triangle cuts of type 2, and

(iii) then lift the coefficients of the variables which were relaxed to be continuous, so as to re-introduce integrality information.

In contrast, we do not work with the continuous group relaxation but instead generate split, cross and crooked cross cuts from the two-row relaxation that is formed by a pair of tableau rows and the integrality and bound information on all variables. In other words, we work with tighter relaxations as we keep the integrality and bound information on all variables. We next formally define the relaxations we are interested in and discuss our computational approach and results. We note that our preliminary computational experiments suggest that the additional gap closed by crooked cross cuts over cross cuts is not significant.

5.1. One-row vs two-row relaxations

Given a mixed-integer program of the form

$$\min\{c^T x : Ax = b,\ l \le x \le u,\ x_k \in \mathbb{Z}\ \forall k \in T\},$$

where T is the set of indices of the integer variables, consider the reformulation obtained by using an optimal basis of the LP-relaxation:

$$\min\{c^T x : x_B + A_B^{-1} A_N x_N = A_B^{-1} b,\ l \le x \le u,\ x_k \in \mathbb{Z}\ \forall k \in T\},$$

where x_B and x_N denote the basic and non-basic variables and A_B and A_N denote the submatrix of A corresponding to the the basic and non-basic variables, respectively. For each row of this relaxation where the basic variable x_{B_i} is integral, i.e. $B_i \in T$, but the right-hand side is fractional, we define the following one-row relaxation of the feasible region:

$$P_i = \{x : x_{B_i} + \sum_{k \in N} \bar{a}_{ik} x_k = \bar{b}_i,\ l \le x \le u,\ x_k \in \mathbb{Z}\ \forall k \in T\},$$

where $\bar{a}_{ik}$ denotes the k'th entry of the associated row of $A_B^{-1} A_N$ and $\bar{b} = A_B^{-1} b$. It is well-known that the Gomory mixed-integer (GMI) cut derived from the ith row of the simplex tableau is a valid inequality for P_i. Following Balas et al. (1996), we refer to adding all violated GMI cuts simultaneously to the LP relaxation as a *round* of GMI cuts.

Subsequent to the work in Dash and Günlük (2008) and Fischetti and Saturni (2007) comparing the effectiveness of one round of GMI cuts to different classes of "one-row" cuts, Fukasawa and Goycoolea (2011) showed computationally that, for most MIPLIB 3.0 problems, adding all (knapsack) cuts based on the relaxations P_i does not yield improved bounds over that obtained by adding one round of GMI cuts. In other words, they showed that no additional cuts from individual rows of an optimal simplex tableau are useful over and above GMI cuts from these rows for most MIPLIB 3.0 problems.

Therefore, a natural question is whether one can obtain useful valid inequalities by using pairs of simplex tableau rows that give relaxations of the form:

$$P_{ij} = \{x : x_{B_i} + \sum_{k \in N} \bar{a}_{ik} x_k = \bar{b}_i,\ x_{B_j} + \sum_{k \in N} \bar{a}_{jk} x_k = \bar{b}_j,\ l \le x \le u,\ x_k \in \mathbb{Z}\ \forall k \in T\},$$

where $i \ne j \in \{1, \ldots, m\}$, $B_i, B_j \in T$, and both $\bar{b}_i$ and $\bar{b}_j$ are fractional. Clearly $P_{ij} = P_i \cap P_j$. In this section we do not attempt to generate all valid inequalities for P_{ij} but instead restrict our attention to those that can be generated as cross cuts.

5.2. Computational approach and separation

For a given MIP, we first solve its LP relaxation to obtain an optimal simplex tableau and the corresponding solution. We then identify the "numerically stable" rows of the tableau that lead to violated GMI cuts (i.e., the rows i such that $B_i \in T$ and $\bar{b}_i$ is fractional). As in Fischetti and Salvagnin (2011), a tableau row is considered numerically stable if the ratio of its largest coefficient to its smallest coefficient (in absolute value) is not too large; more precisely we only consider tableau rows with $\max\{|a_{ik}|/|a_{il}| : k, l \in N\} \leq 10^9$ as in Fischetti and Salvagnin (2011). Let the set of such tableau row indices be I_0. For each $i, j \in I_0$, we then construct the two-row relaxation P_{ij} provided that the defining equations for P_i and P_j have common non-basic variables with non-zero coefficients. We let $I \subset I_0 \times I_0$ be the set of all such pairs of row indices. The motivation for ignoring index pairs i, j where the associated tableau rows do not have any variables in common is that both split cuts (and rank-2 split cuts) from such P_{ij} are implied by split cuts (and rank-2 split cuts) from P_i and P_j (Dash 2010), and the experimental work by Fukasawa and Goycoolea (2011) suggests that cuts in addition to the GMI cuts from individual rows are unlikely to be very useful. In our experiments, we observe that $|I|$ is usually much less than $|I_0|(|I_0|-1)/2$ leading to much faster computing times without a noticeable decrease in final bounds obtained.

We use a few different procedures to generate cuts from the two row relaxations P_{ij}. Our first procedure separates split cuts from two-row relaxations P_{ij} via the DG heuristic (Dash and Goycoolea 2010). In this case we invoke the FEAS, SPARSE and RANDOM versions of the heuristic; see (Dash and Goycoolea 2010, Table 4).

Algorithm 4: Split2.heur$(I, P, Q, (\bar{x}, \bar{y}))$

Input: List of row index pairs I, original mixed integer set P, current LP relaxation $Q \subseteq P^{LP}$ and point $(\bar{x}, \bar{y}) \in Q$.

```
for (i, j) ∈ I do
    Use the DG heuristic to find rank-1 GMI cuts for P_{i,j} violated by (x̄, ȳ).
    if violated cuts are found then
        Add the violated cuts to Q.
        Solve Q and let (x̄, ȳ) be its solution.
        Restart the for loop from the beginning.
    end
end
```

As we may repeat the for loop in the above procedure multiple times, we may consider the same P_{ij} multiple times in order to generate cuts, which differs from the approach described in Dey et al. (2010) and Basu et al. (2011). Also note that in this procedure we do not restrict the number of cuts added. Using this procedure, we are able to show that split cuts from two row relaxations yield significantly better bounds than one round of GMI cuts.

Our second procedure uses the DG heuristic to generate cross cuts through the Cross.DG procedure (see Algorithm 3 in Section 4.2). The main input to this procedure is the list of GMI cuts and associated split disjunctions for the tableau rows indexed by I_0. More precisely the list is defined as $L^{GMI} = (\pi^i, \gamma^i, \{g_i\})_{i \in I_0}$ where g_i denotes the GMI cut derived from row i and (π^i, γ^i) is the associated split disjunction. The output of the procedure consists of good cross disjunctions together with the associated pairs of row indices which we later use in the LP cross cut separator.

Algorithm 5: Cross2.heur$(I, P, L^{GMI}, Q, (\bar{x}, \bar{y}))$

Input: List of row index pairs I, original mixed integer set P, list of GMI cuts with associated split disjunctions L^{GMI}, current LP relaxation $Q \subseteq P^{LP}$ and point $(\bar{x}, \bar{y}) \in Q$.

Result: List I^{GOOD} of cross disjunctions with associated row index pairs.

1 Set $I^{GOOD} = \emptyset$.
2 **for** $(i, j) \in I$ **do**
3 Set $(C, D) = \text{Cross.DG}\left(\{(\pi^i, \gamma^i, \{g_i\}), (\pi^j, \gamma^j, \{g_j\})\}, P_{i,j}, (\bar{x}, \bar{y}), \infty\right)$.
4 **if** $|C| > 0$ **then**
5 Set I^{GOOD} to the union of I^{GOOD} and $D \times \{(i, j)\}$.
6 Add the violated cuts C to Q.
7 Solve Q and let $(\bar{x}, \bar{y})$ be its solution.
8 Restart the for loop from the beginning.
9 **end**
10 **end**

We also define a routine **Cross2.MIP** – which generates cross cuts from pairs of tableau rows – by simply replacing the call to Cross.DG in line 3 above by the call

$$(C, D) = \text{Cross.MIP}\left(\left\{\left(\pi^i, \gamma^i\right), \left(\pi^j, \gamma^j\right)\right\}, P_{i,j}, (\bar{x}, \bar{y}), 2\right).$$

In other words, for every P_{ij} with $(i,j) \in I$, we find an approximate MIR cut for $conv\left(P_{ij}^{LP} \cap S(\pi^i, \gamma^i)\right)$ and $conv\left(P_{ij}^{LP} \cap S(\pi^j, \gamma^j)\right)$.

To separate cross cuts which are not necessarily rank-2 split cuts, we use the following LP separation routine which takes as input a list I^+ of row index pairs with associated split disjunctions.

Algorithm 6: Cross2.LP$(I^+, P, Q, (\bar{x}, \bar{y}))$

Input: List of row index pairs with associated split disjunctions I^+, original mixed integer set P, current relaxation Q and initial point to separate $(\bar{x}, \bar{y}) \in Q \subseteq P^{LP}$.

```
for (π¹, γ¹, π², γ², i, j) ∈ I⁺ do
    if Cross.LP({(π¹, γ¹, π², γ²)}, P_{i,j}, (x̄, ȳ), 1) yields a violated cut then
        Add cut to Q, resolve Q and let (x̄, ȳ) be its solution.
        Restart the for loop from the beginning.
    end
end
```

The first list we consider for the argument I^+ is $I^{GMI} := \{(\pi^i, \gamma^i, \pi^j, \gamma^j, i, j)\}_{(i,j)\in I}$ where (π^i, γ^i) is the disjunction associated with the GMI cut g_i derived from row i. The second set we consider is I^{GOOD} obtained from Cross2.heur.

5.3. Computational Experiments

Our computational results are obtained on a 2.93 GHz Intel Xeon machine running the Linux operating system. We solve linear programs and auxiliary integer programs (in Cross.MIP) with IBM ILOG CPLEX 12.2. In general, our computing times for the algorithms Cross2.LP and Cross2.MIP are nontrivially larger than the times for Split2.heur and Cross2.heur; our implementation of the first two routines is not competitive with the codes for the latter two routines.

We next discuss the integrality gap closed with cuts obtained using the routines in Section 5.2 for 54 out of 65 instances from MIPLIB 3.0. For an MIP where the objective function is to be minimized as in the MIPLIB instances, the *integrality gap closed* (abbreviated to as "gap closed") by a set $\mathcal{C}$ of cuts is

$$\frac{\text{(Obj. value after cuts } \mathcal{C} \text{ are added - LP relaxation value)}}{\text{(MIP optimum value - LP relaxation value)}} \times 100.$$

The 11 instances that we discard either have no integrality gap left, or no integrality gap closed after extensive generation of split cuts as reported in Balas and Saxena (2008), Dash et al. (2010); these instances are *dsbmip, enigma, noswot, air03, 10teams, mod010, markshare1, markshare2, pk1, stein27*, and *stein45*. Further, we replace free variables in any remaining instance with the difference of two nonnegative variables.

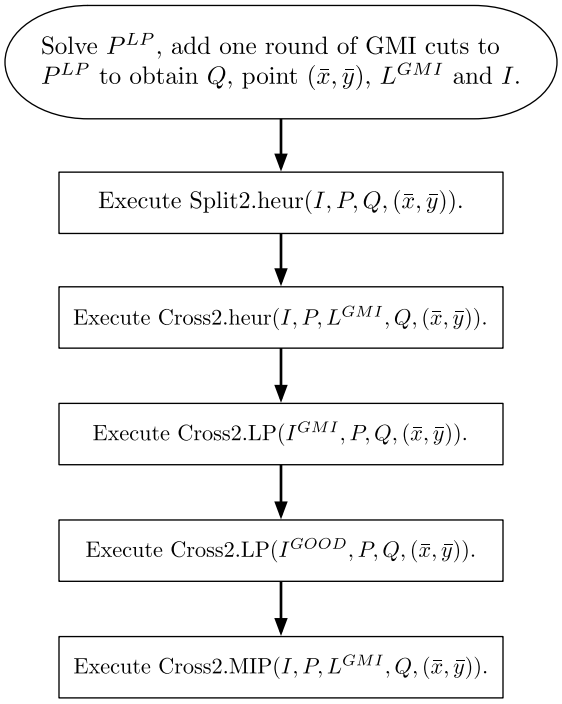


Figure 4 Flow chart for experiments with 2-row cuts

We execute the algorithms in Section 5.2 in the order shown in the flowchart in Figure 4. At the end of each algorithm, we terminate if the total elapsed time exceeds one hour. We thus run the separation algorithms sequentially in increasing order of computational difficulty. Consequently, the total gap closed after each algorithm is non-decreasing; also we never delete cuts, and therefore the number of cuts generated at the end of each algorithm also non-decreasing.

We present our computational results in Table 1 where the problem name is given in the column titled "problem". The next three columns give, respectively, the gap closed by one round of GMI cuts, the number of violated GMI cuts, and time taken to obtain these cuts (i.e., to generate them, and to reoptimize the LP after adding them). We then

give similar information for cuts generated by Split2.heur, Cross2.heur, and finally for cuts generated by all algorithms (under the heading ALL, which is how we will refer to the combination of all algorithms henceforth). Note that the computing time for some instances exceeds 3600 seconds as we check against the time limit only at the end of each algorithm in the flowchart in Figure 4. We terminate the overall procedure at one hour for *swath* as Cross2.LP($I^{GMI}, P, Q, (\bar{x}, \bar{y})$) starts before an elapsed time of an hour and takes almost 9 hours. However, the number of cuts and time reported for each algorithm do not include the number of GMI cuts and time taken to obtain them, though they include the number of cuts and time for any other preceding algorithm. If we reach the time limit before completing all algorithms, say while executing Split2.heur, we give a '-' in subsequent columns. Finally, in the last line, we give the arithmetic means of gap closed and geometric means of running time and number of cuts added. If for some problem, the time limit is reached before beginning the execution of Cross2.heur (say), then we use the gap closed, number of cuts and computing time from the preceding executed algorithm (say Split2.heur) in the means listed in the last row (and similarly for ALL).

Note that for many problems we obtain a significant increase over the gap closed by one round of GMI cuts by adding a small number (about the same order of magnitude as the number of GMI cuts) of cuts generated from two tableau rows. The numbers for the `gesa` problems, for example, are especially striking, as the gap closed increases by more than 30% with the addition of a small number of cuts. The additional gap closed as we move from Split.heur (35.76%) to Cross.heur (37.10%) to ALL (38.39%) is not as striking, as can be seen in the averages in Table 1. This is not shocking as the relaxations P_{ij} have only two rows, and we would expect a few cuts to give a good approximation to the integer hull.

We observe that if we execute Cross2.LP with I^+ set to I^{GMI} after adding one round of GMI cuts the gap closed is 32.70% on the average as opposed to 37.10% with Cross2.heur or 35.76% with Split2.heur. This means that the disjunctions found by Cross2.heur or Split2.heur are useful, and the disjunctions associated with the first round of GMI cuts alone are not enough to improve the bounds a lot (when used to obtain cuts for P_{ij}).

We note that the average gap closed for 53 out of these 54 problems considered in Louveaux and Poirrier (2012) (they have numerical difficulties with *dano3mip*) is 32.38%.

Table 1 Gap closed with two row cuts on MIPLIB 3.0 problems

	GMI			Split2.heur			Cross2.heur			ALL		
problem	% gap	cuts	time	% gap	cuts	time	% gap	cuts	time	% gap	cuts	time
air04	6.71	292	4.545	12.71	197	3713.11	-	-	-	-	-	-
air05	4.64	224	1.141	8.80	154	3247.60	8.80	154	4149.93	-	-	-
arki001	29.26	56	0.088	42.41	65	12.60	46.77	99	37.58	55.13	580	3618.72
bell3a	60.43	32	0.001	67.08	2	0.06	67.15	3	0.18	67.15	3	63.00
bell5	14.53	25	0.001	17.41	3	0.07	17.41	4	0.17	17.78	12	71.42
blend2	16.36	6	0.002	17.38	11	0.03	18.07	23	0.08	21.62	58	27.91
cap6000	41.65	2	0.069	54.96	11	0.41	62.73	28	1.33	63.76	144	2049.81
dano3mip	0.02	97	48.803	0.03	15	3738.08	-	-	-	-	-	-
danoint	1.74	52	0.141	1.74	7	17.96	1.74	12	42.22	1.74	97	1131.55
dcmulti	43.08	49	0.007	49.94	29	2.09	52.21	43	5.44	54.62	83	567.88
egout	55.93	40	0.001	58.06	4	0.09	58.45	7	0.25	58.85	9	19.92
fast0507	1.68	306	914.395	1.89	4	3591.79	-	-	-	-	-	-
fiber	65.02	41	0.008	76.04	25	0.68	76.49	27	1.35	78.87	226	3662.96
fixnet6	10.87	60	0.007	11.33	25	1.09	11.51	32	2.00	11.71	57	96.56
flugpl	11.74	10	0.000	11.74	0	0.01	13.70	9	0.07	14.10	13	20.18
gen	61.62	43	0.006	79.10	22	3.69	79.10	22	4.54	79.10	22	199.03
gesa2	27.19	58	0.013	51.40	37	0.84	64.32	42	1.26	65.00	68	81.18
gesa2_o	30.21	73	0.013	62.42	26	0.68	63.10	42	2.45	63.20	69	155.65
gesa3	45.87	85	0.033	82.37	29	3.72	82.69	36	9.03	83.01	44	446.39
gesa3_o	50.57	100	0.036	85.07	25	4.43	85.25	30	10.77	85.32	35	600.57
gt2	67.72	11	0.001	69.15	36	0.11	69.90	47	0.19	77.09	3314	2319.70
harp2	8.69	10	0.007	13.08	55	0.46	13.70	68	1.03	14.78	628	3613.73
khb05250	74.91	19	0.002	80.75	4	0.22	88.06	18	1.70	88.56	20	68.38
l152lav	1.55	51	0.036	26.08	121	67.39	26.08	122	85.52	26.09	124	4020.90
lseu	48.42	12	0.001	56.97	26	0.09	57.23	34	0.28	59.81	321	498.63
mas74	6.67	12	0.001	9.46	101	0.95	9.46	101	1.07	9.82	156	82.92
mas76	6.42	11	0.001	9.41	57	0.27	10.26	68	0.76	11.83	476	290.42
misc03	7.24	18	0.004	10.34	8	0.13	10.34	12	0.35	11.55	17	218.33
misc06	29.40	13	0.010	30.77	2	0.07	30.77	2	0.12	31.31	11	13.57
misc07	0.72	26	0.008	0.72	11	0.72	0.72	12	1.99	0.72	12	157.45
mitre	80.76	796	0.722	88.64	116	3716.44	-	-	-	-	-	-
mkc	1.21	70	0.240	23.72	238	203.11	23.72	241	241.75	27.11	822	3647.00
mod008	20.89	5	0.001	61.59	54	0.13	61.61	58	0.22	65.56	106	89.66
mod011	17.11	16	0.093	17.14	3	0.95	17.20	6	3.72	18.73	73	2757.33
modglob	15.10	30	0.002	33.69	67	1.66	35.33	82	2.69	35.44	105	242.57
nw04	62.27	6	0.367	68.44	6	4.82	68.44	6	9.78	68.92	7	4531.45
p0033	54.60	6	0.000	64.91	19	0.02	64.91	19	0.02	66.80	49	31.83
p0201	18.24	20	0.004	18.24	2	0.04	24.42	7	0.16	24.42	14	67.87
p0282	3.70	26	0.002	33.54	213	6.00	33.54	213	6.23	40.68	533	2090.18
p0548	39.46	47	0.004	64.15	336	6.06	68.70	461	14.19	73.49	3971	3631.71
p2756	0.46	36	0.009	0.54	13	0.19	0.54	13	0.34	2.63	81	76.80
pp08a	52.88	51	0.005	64.59	21	0.21	68.84	44	0.62	70.28	57	31.36
pp08aCUTS	30.07	41	0.009	41.12	41	2.19	41.75	48	3.89	42.47	64	139.13
qiu	1.99	36	0.204	1.99	0	0.67	1.99	1	2.48	2.60	46	570.98
qnet1	12.73	49	0.084	28.83	125	125.52	28.92	142	343.12	29.40	321	3687.26
qnet1_o	30.71	11	0.011	33.47	41	0.29	34.43	61	0.92	36.49	284	574.32
rentacar	29.05	16	0.234	29.05	0	0.24	29.05	0	0.67	29.15	2	207.64
rgn	4.49	16	0.001	20.28	33	0.29	24.67	45	0.89	27.92	80	89.49
rout	0.32	29	0.029	3.15	98	13.47	4.01	169	219.41	4.03	504	3731.11
set1ch	38.11	138	0.011	56.66	137	3.97	60.86	218	19.39	60.86	218	92.80
seymour	8.39	598	37.560	10.11	12	5180.11	-	-	-	-	-	-
swath	17.66	45	0.099	33.38	10	7.76	33.38	10	21.51	33.38	10	3600*
vpm1	9.45	15	0.002	11.15	6	0.02	11.15	6	0.04	11.15	6	3.89
vpm2	12.58	30	0.003	24.26	33	0.63	26.50	51	1.25	26.63	64	29.76
Average	25.80	32	0.02	35.76	21	1.90	37.10	28	4.26	38.39	67	388.22

For these problems, the respective values for Split2.heur and Cross2.heur are 36.44% and 37.80%. However, the experiments are not comparable, as Louveaux and Poirrier do not fix the tableau but update it up to five times, and also consider only a subset of all tableau row pairs.

In Table 2, we give more information on the relative performance of the algorithms in Table 1. In columns two to four, we repeat the averages for gaps closed, computing time and number of cuts from the last line of Table 1. In column six, we give the number of problems for which the gap closed minus the gap closed by one round of GMI cuts is at least 5%, and in column eight, we give the number of problems for which at least 5% of the gap remaining after one round of GMI cuts is closed by a given algorithm. In columns five and seven we give information similar to columns six and eight, respectively, except that we compare with the gap closed by the previously listed algorithm. First notice that Split2.heur takes about 100 times the computing time of a round of GMI cuts, and it improves the gap closed by at least 5% for 29 out 54 problems. For only about 1.34% extra gap closed, Cross2.heur takes more than twice the time as Split2.heur; furthermore it improves the gap closed by 5% or more for only 4 problems; in other words, it makes a big difference only for a small number of problems. However, changing the gap closed from say 90% to 95% is much harder than from say 30% to 35%; thus an absolute change in gap closed is not always the best measure of the effectiveness of an algorithm. This is the reason for considering relative changes in gap closed in columns seven and eight. Even by this measure, our implementation of cross cut separation makes not as big a difference as split cut separation from pairs of tableau rows.

Table 2 Comparison of different methods

	Average			Absolute change		Relative change	
method	gap closed	time	cuts	prev.+5%	GMI+5%	5% prev. gap	5% GMI gap
GMI	25.80	0.02	32	-	-	-	-
Split2.heur	35.76	1.90	21	29	29	29	29
Cross2.heur	37.10	4.26	28	4	31	9	34
ALL	38.39	388.22	67	3	35	9	36

It is natural to ask whether we can obtain the gap closed by Cross2.heur (or Split2.heur) by choosing only a subset of GMI cuts, and generating cuts from associated pairs of rows, or by carefully choosing pairs of rows. After all, in Cross2.heur, often many pairs of rows have to be examined before cuts are found. We attempt to answer this question by the computational experiments summarized in Table 3. In all algorithms considered in this experiment (except in Cross2.heur.nosort), each time we start the for loop in Section 5.2, we first sort the GMI cuts in decreasing order of dual values assigned to these cuts in the

previous (strengthened) LP relaxation. Our purpose here is to distinguish between more and less important GMI cuts. For example, Cross2.heur x 30% means that we generate cross cuts from pairs of rows associated with the top 30% of GMI cuts only. The next two rows have a similar meaning. In addition, in the for loop, the sets P_{ij} are considered in decreasing order of sums of dual values for the associated GMI cuts. For example, the first set P_{ij} considered for cross cut separation would be the set associated with the two rows that give the two "most important" GMI cuts (provided that they have common non-basic variables). The time taken by Cross2.heur x 30% is only 10 times the time to compute GMI cuts, but it does not close too much more gap. As more pairs of rows are considered for cross cut separation, more of the gap is closed but at a cost of higher computing times. Cross2.heur.nosort means we do not sort the GMI cuts by dual values, and simply consider them in the order they appear in the tableau and a pair $(i_1, j_1) < (i_2, j_2)$ if the first pair is lexicographically less then the second one. It does seem to take noticeably more time than Cross2.heur to close essentially the same gap (this relationship also holds between the sorted and non-sorted variants of Cross2.heur x 30% etc.). Thus our heuristic to order pairs seems useful, but our heuristic to drop pairs altogether seems less so. Finally, we can get the same gap as Cross2.heur, but in less time if we only consider pairs of tableau rows which have common variables that are basic in the current relaxation; this is given in Cross2.heur.basici.

Table 3 Comparison of different variants of Cross2.heur

Cut Separation Method	gap closed	time
Cross2.heur	37.10	4.26
Cross2.heur x 30%	29.47	0.11
Cross2.heur x 50%	32.31	0.47
Cross2.heur x 70%	34.56	1.57
Cross2.heur.nosort	36.81	6.08
Cross2.heur.basici	36.94	3.65

6. Bounds on cross and crooked cross cut closures

In this section we report on our computational experiments with cross and crooked cross cuts that are obtained using the full formulation of a MIP as opposed to using only two-row relaxations. We aim to establish that cross and crooked cross cuts give better lower bounds than split cuts on practical problem instances, namely the MIPLIB 3.0 problem

instances. In earlier experiments with split cuts, Balas and Saxena (2008) and Dash et al. (2010) present separation models that can, in principle, optimize over the split closure to any degree of precision. We do not propose any such mechanism for the cross or crooked cross cut closure; instead, we use heuristics to obtain effective cuts.

We present two computational experiments below. In the first one, we start with a short, fixed list of split disjunctions (the ones in L^{GMI}, described in Section 5.2) and compare the effect of split cuts with that of cross and crooked cross cuts that can be generated using the disjunctions in this list only. In the second experiment, we use heuristics to generate more split and cross disjunctions, motivated by the belief that not all "good" cross disjunctions correspond to pairs of split disjunctions from a list of "good" split disjunctions. With our heuristics, we are able to obtain better bounds than the best known split closure bounds for a number of problem instances.

6.1. Experiments with GMI disjunctions

In our first experiment, we first solve the LP relaxation of a given problem instance and identify numerically stable rows of the optimal tableau that yield violated GMI cuts, and the associated set of indices I_0 as in Section 5.2.Let $S^{GMI} = (\pi^i, \gamma^i)_{i \in I_0}$, i.e., S^{GMI} is the list of split disjunctions in L^{GMI} and we call it the list of GMI disjunctions.

We first add the GMI cuts in L^{GMI} (corresponding to the disjunctions S^{GMI}) to P^{LP} and call the resulting bound the GMI bound. We then separate and add split cuts iteratively until all the split cuts derivable from the disjunctions in S^{GMI} are satisfied. We record the lower bound obtained at the end of this step as the Split bound. To separate split cuts we use the algorithm Split.LP described in Section 3.1. We generate up to 10 cuts at a time to speed up the procedure by setting the parameter $p = 10$ in Split.LP.We note that this does not affect the final bound obtained by split cuts. In addition, we use the ideas discussed in Section 3.4 to avoid running the separation LP for disjunctions that cannot separate the current point at hand.

We next generate cross cuts iteratively from all pairs of disjunctions in S^{GMI}using Cross.LP. To speed up the computation, we always check for violated split cuts using algorithm Split.LP after violated cross cuts are added. In addition, we generate 5 cuts at a time by setting the parameter $p = 5$ in Cross.LP.Here we use a smaller number for p in Cross.LP than in Split.LP as generating cuts using Cross.LP is more expensive. We chose

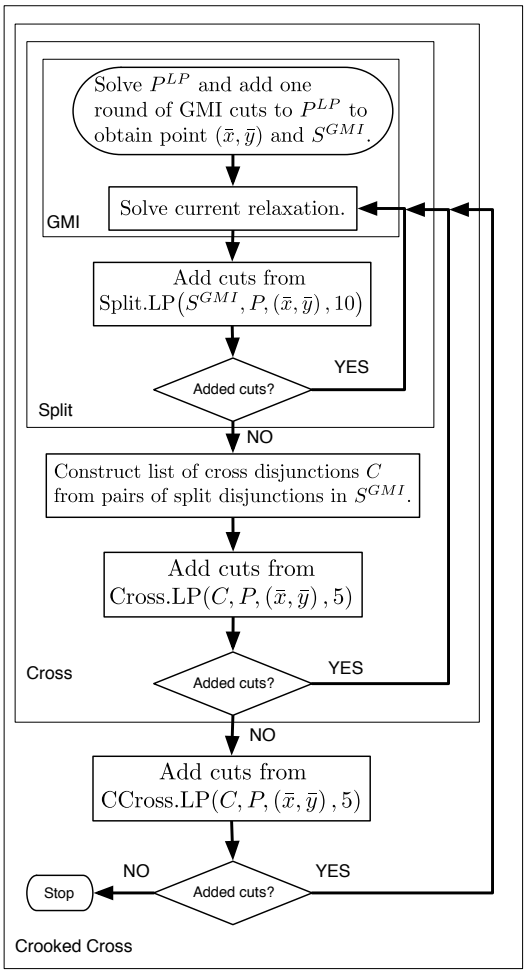


Figure 5 Flow chart for experiments with GMI disjunctions

the values of p after some preliminary testing.We also use the ideas discussed in Section 3.4 for efficiency. When no further cuts can be generated, we record the lower bound at hand as the Cross bound.

The last step is to generate crooked cross cuts from all pairs of disjunctions in S^{GMI} using CCross.LP with $p = 5$. To speed up the computation, we look for violated split and cross cuts after adding violated crooked cross cuts. As before, we skip solving separation LPs for some disjunctions using ideas from Section 3.4. We call the final bound obtained at the end of this step the Crooked Cross bound. We summarize the overall procedure in Figure 5.

In Table 4 we report on the average integrality gap closed by different classes of cuts for 53 MIPLIB 3.0 problems. We consider the same 54 instances discussed in Section 5.3 except *fast0507*; We exclude it from all experiments in this Section as we have numerical difficulties with it. The rows of Table 4, respectively, give the gap closed by the first round

of GMI cuts, split cuts from the GMI disjunctions, cross cuts from the GMI disjunctions and crooked cross cuts from the GMI disjunctions. For comparison, the last line gives the average of the best known split closure bounds combining results in Balas and Saxena (2008), Dash et al. (2010), Dash and Goycoolea (2010).

Table 4 Experiments with GMI disjunctions on MIPLIB problems

Cut family	% gap closed
GMI cuts	26.25
Split cuts from GMI disjunctions	41.42
Cross cuts from GMI disjunctions	43.96
Crooked cross cuts from GMI disjunctions	45.15
Best split closure bound	79.89

We make a few observations based on Table 4. The first round of GMI cuts does not yield a good approximation of the split closure on the average. However, separating additional split cuts from the GMI disjunctions significantly improves the lower bound. Adding Cross cuts based on pairs of these disjunctions also yields a non-trivial and yet not very substantial improvement. The average improvement due to crooked cross cuts is non-negligible but this is a bit misleading as almost all of this improvement is due to two problem instances, namely *gesa2* (about 29%) and *modglob* (about 16%). Without these two instances, the extra gap closed is very small on the average, and due to this observation we do not separate crooked cross cuts in subsequent experiments. We also note that the bound obtained using only GMI disjunctions is quite far from the best split closure bound.

6.2. Experiments with cross cuts from heuristic disjunctions

In this section, we attempt to exceed the best known elementary split closure bounds using cross cuts. The first step in our algorithm is to generate a list of good split disjunctions using the rank-1 GMI cut heuristic in Dash and Goycoolea (2010). We modify their code to store the split disjunctions whenever a violated GMI cut is found and keep track of the GMI cut generated by each disjunction. We call the list of split disjunctions obtained from the heuristic S^{DG}. To control the overall computing time, we terminate the heuristic when the number of generated cuts (and therefore the size of S^{DG}) exceeds 1000.

Our next step is to separate all split cuts obtainable from S^{DG} using Split.LP iteratively, as described in Section 6.1. While generating split cuts, we again keep track of which cuts are generated using which disjunction. At the end of this step, each split disjunction

in S^{DG} has at least one GMI cut and possibly many other split cuts associated with it. We call this combined list of disjunctions together with the cuts associated with each disjunction the list SC^{DG}. Using these split cuts, we can improve over the best known split closure bounds for 6 instances. In the next table, for each problem, we give its name, the percentage gap closed with split cuts in SC^{DG}, the best bounds for the split closure from Balas and Saxena (2008), Dash et al. (2010), Dash and Goycoolea (2010), and the percentage 100*(SC^{DG} splits - Best Split)/(100-Best Split).

problem	SC^{DG}	Best Split	Improvement	problem	SC^{DG}	Best Split	Improvement
gesa3o	95.3	95.2	2.1	modglob	94.0	92.2	23.1
mkc	52.4	49.3	6.1	qiu	78.1	77.5	2.6
mas74	14.3	14.0	0.3	rentacar	57.1	45.0	40.2

Unlike our experiments with the GMI disjunctions discussed earlier, we do not proceed to cross cut separation via LPs right away with all pairs of disjunctions in S^{DG}. Instead, we proceed with the separation heuristics described in Section 4. First we search for violated rank-2 split cuts that are cross cuts using the heuristic Cross.DG with the list SC^{DG}. When separating cross cuts with this heuristic, we also keep track of pairs of split disjunctions that lead to violated cuts and save them in a list called C for later use. Next, we generate cross cuts using the MIP heuristic Cross.MIP and then with the LP heuristic Cross.LP using the pairs of disjunctions in C. Finally we separate cross cuts from C^{DG} which consists of all pairs of disjunctions in S^{DG} through Cross.LP. At the end of this step, if we do not find any violated cross cuts, we terminate, else we fall back to separating split cuts. The overall algorithm is summarized in Figure 6, where we again note what parts of the procedure generate a given bound. Finally, while it is not explicitly noted in the figure, we use the ideas discussed in Section 3.4 to avoid solving LPs that cannot yield any cuts.

In our computational experiments we focus on 31 out of the 53 MIPLIB 3.0 instances discussed in 6.1 where split cuts close at least 1% and at most 99% of the integrality gap. For these instances the average gap closed by split cuts is 68.89% (Balas and Saxena 2008, Dash et al. 2010, Dash and Goycoolea 2010) whereas the gap closed by rank-1 GMI cuts generated by the DG heuristic DEF in Dash and Goycoolea (2010) is 47.93%. We run our

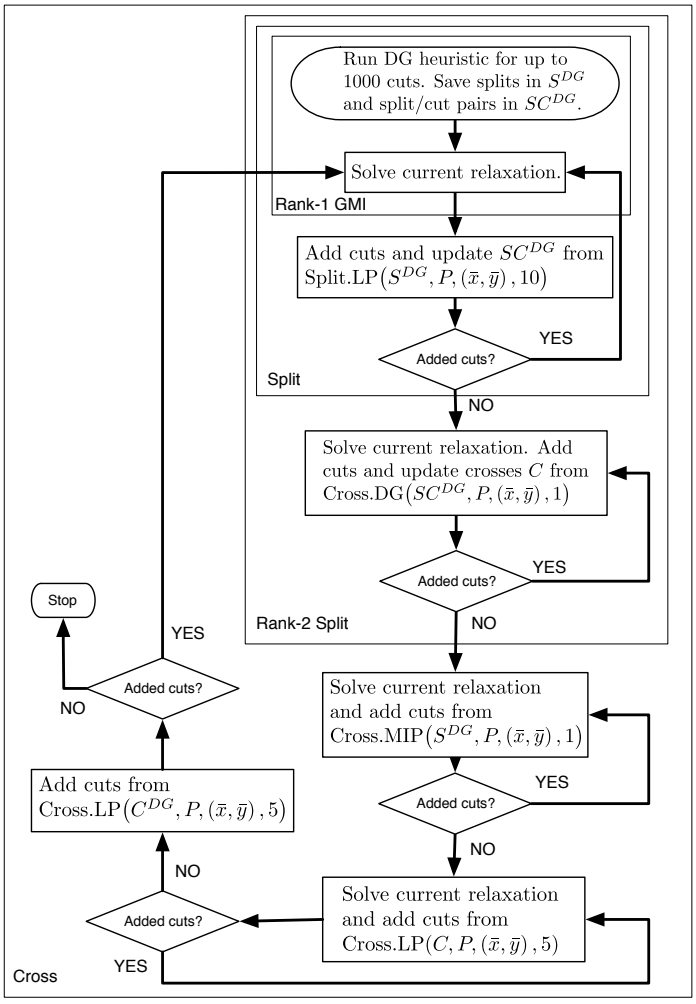


Figure 6 Flow chart for experiments with heuristic disjunctions

code with a time limit of 3 hours and collect the results in a table in the online supplement (which has the same format as Table 5 below). Examining this table, we observe that for 16 instances the final gap closed is greater than the best split cut bounds from Balas and Saxena (2008), Dash et al. (2010), Dash and Goycoolea (2010). We choose these 16 instances (and another three marked by a †) and rerun with a time limit of 48 hours and more aggressive rank-1 GMI cut generation using a variant of the default DG heuristic in Dash and Goycoolea (2010) which, in the notation in (Dash and Goycoolea 2010, Table 4), can be expressed as DEF + BG5000. That is, we invoke their "branch-and-gather" algorithm for 5000 nodes instead of only 5 nodes, and keep other settings in their default heuristic DEF the same to get a better approximation of the split closure.

In Table 5, we report on 16 out of the 32 instances where the bound we obtain with cross cuts exceeds the best split closure bound (from Balas and Saxena (2008), Dash et al. (2010),

Dash and Goycoolea (2010)) by an absolute value of at least 0.5%. For another instance, namely *rentacar*, we close 57.1% of the integrality gap with split cuts alone as opposed to the previously best known bound of 44.9% gap closed using split cuts; however as our split cut separation code does not terminate within the time limit, we do not generate any cross cuts and thus omit this instance from Table 5. In the first column, we give the problem name. In the second, third and forth columns, we give the bounds obtained by GMI cuts from the initial tableau, rank-1 GMI cuts from the modified DG heuristic, and split cuts using the disjunctions S^{DG}. The next two columns give, respectively, the bound obtained with cross cuts and the best bound obtained with split cuts in Balas and Saxena (2008), Dash et al. (2010), Dash and Goycoolea (2010). The following column gives the percentage of the remaining gap (after split cuts) closed by our procedure, i.e. $100*(Cross-Best\ Split)/(100-Best\ Split)$. The last column gives the overall computing time and the symbol "+" means that the code reached the time limit of 48 hours. This table was obtained with the same code and machine type as used in the previous experiments, except that the clock speed was slightly lower, at 2.33 GHz. In the last row, we give the arithmetic mean of the gaps closed across the 18 instances.

Table 5 Gap closed with cross cuts on MIPLIB problems

problem	GMI	DG	Split	Cross	Best Split	Improvement	time (sec)
bell5	14.5	25.9	86.2	99.8	93.0	97.1	2,120
cap6000	41.7	64.6	65.2	67.2	65.2	5.7	62,097
gesa3	45.9	93.3	95.1	97.4	95.8	38.1	+
gesa3o	50.6	93.1	95.3	99.0	95.2	79.2	124,339
gt2	67.7	96.6	96.7	99.2	98.4	50.0	1,470
mas74	6.7	11.4	14.3	15.7	14.0	2.0	+
mas76	6.4	16.1	25.1	34.5	26.5	10.9	+
mkc	1.2	4.1	52.4	55.3	49.3	11.8	+
modglob	15.1	90.8	94.0	99.0	92.2	87.2	18,273
p0033	54.6	84.1	86.2	100.0	87.4	100.0	301
p0201	18.2	71.5	74.0	98.4	74.9	93.6	+
pp08a	52.9	96.0	96.6	98.4	97.0	46.7	165,687
pp08aCUTS	30.1	93.3	94.7	96.5	95.8	16.7	+
qiu	2.0	21.8	78.1	78.4	77.5	4.0	+
set1ch	38.1	88.0	88.7	98.6	89.7	86.4	+
vpm2	12.6	72.0	76.5	81.7	81.0	3.7	+
Average	25.5	61.3	76.1	82.2	77.3	45.8	+

The averages of our split cut bounds for these instances and the best split cut bounds are very close, namely 76.1% and 77.3%, respectively. The average gap closed with additional cross cuts is 82.2%. In particular, note that for the problem *p0033*, 86.2% of the integrality gap is closed by split cuts (and this number is close to the best split cut bound) and cross cuts close all the remaining integrality gap within a few hundred seconds. On the average, cross cuts close almost half of the remaining gap after split cuts. Also note that for most of the instances, the algorithm terminated due to the time limit of 48 hours.

7. Concluding Remarks

Based on our our computational experiments we conclude that cross cuts yield an improvement over split cuts for some practical MIP problem instances. The heuristics we use to separate these cuts, however, are very computationally intensive and therefore not practical. An important remaining question is whether or not a fast and effective separation procedure can be developed to use these cuts to solve MIPs in practice. Most of the cutting planes used in current MIP solvers are split cuts and cross cuts have the potential to have a nontrivial impact. Unlike cross cuts, however, we are not able to make a case for crooked cross cuts as the additional benefit gained by separating these cuts does not seem significant.

We also studied the effect of two-row cuts which have been demonstrated to be significant in certain theoretical contexts (for example to obtain the convex hull of the two-row continuous group relaxation) but have not been shown to be computationally effective for practical MIPs in earlier studies. Our conclusion based on our experiments is that cuts from two optimal simplex tableau rows are indeed useful and separating cross cuts from such rows seems to be more practical then separating 2D lattice-free cuts and then lifting them.

Acknowledgments

A portion of this research was supported by grant # N000141110724 from the Office of Naval Research. We would like to thank the two referees for useful comments.

References

Andersen, K., Q. Louveaux, R. Weismantel, L. A. Wolsey. 2007. Inequalities from two rows of a simplex tableau. M. Fischetti, D. P. Williamson, eds., *IPCO*, *Lecture Notes in Computer Science*, vol. 4513. Springer, 1–15.

Balas, E. 1979. Disjunctive programming. *Annals of Discrete Mathematics* **5** 3–51.

Balas, E., P. Bonami. 2009. Generating lift-and-project cuts from the lp simplex tableau: open source implementation and testing of new variants. *Mathematical Programming Computation* **1** 165–199.

Balas, E., S. Ceria, G. Cornuéjols, N. Natraj. 1996. Gomory cuts revisited. *Operations Research Letters* **19** 1–9.

Balas, E., A. Saxena. 2008. Optimizing over the split closure. *Mathematical Programming* **113** 219–240.

Basu, A., P. Bonami, G. Cornuéjols, F. Margot. 2011. Experiments with two-row cuts from degenerate tableaux. *INFORMS J. Comput.* **23** 578–590.

Basu, A., M. Conforti, G. Cornuéjols, G. Zambelli. 2010. Minimal inequalities for an infinite relaxation of integer programs. *SIAM Journal on Discrete Mathematics* **24** 158–168.

Bixby, E. R., M. Fenelon, Z. Gu, E. Rothberg, R. Wunderling. 2000. MIP: theory and practice closing the gap. M.J.D. Powell, S. Scholtes, eds., *System Modelling and Optimization*, *IFIP The International Federation for Information Processing*, vol. 46. Springer US, 19–49.

Bixby, R.E., S. Ceria, C. M. McZeal, M.W.P. Savelsbergh. 1998. An updated mixed integer programming library: Miplib 3.0. *Optima* **58** 12–15.

Bonami, P. 2012. On optimizing over lift-and-project closures. *Mathematical Programming Computation* **4** 151–179.

Bonami, P., G. Cornuéjols, S. Dash, M. Fischetti, A. Lodi. 2008. Projected chvátal–gomory cuts for mixed integer linear programs. *Mathematical Programming* **113** 241–257.

Borozan, V., G. Cornuéjols. 2009. Minimal valid inequalities for integer constraints. *Math. Oper. Res.* **34** 538–546.

Conforti, M., G. Cornuéjols, G. Zambelli. 2011a. Corner polyhedron and intersection cuts. *Surveys in operations research and management science* **16** 105–120.

Conforti, M., G. Cornuéjols, G. Zambelli. 2011b. A geometric perspective on lifting. *Oper. Res.* **59** 569–577.

Cook, W., R. Kannan, A. Schrijver. 1990. Chvátal closures for mixed integer programming problems. *Mathematical Programming* **47** 155–174.

Dash, S. 2010. On the complexity of cutting-plane proofs using split cuts. *Operations Research Letters* **38** 109–114.

Dash, S., S. S. Dey, O. Günlük. 2011. On mixed-integer sets with two integer variables. *Operations Research Letters* **39** 305–309.

Dash, S., S. S. Dey, O. Günlük. 2012a. Two dimensional lattice-free cuts and asymmetric disjunctions for mixed-integer polyhedra. *Mathematical programming* **135** 221–254.

Dash, S., M. Goycoolea. 2010. A heuristic to generate rank-1 gmi cuts. *Mathematical Programming Computation* **2** 231–257.

Dash, S., O. Günlük. 2006. Valid inequalities based on simple mixed-integer sets. *Mathematical Programming* **105** 29–53.

Dash, S., O. Günlük. 2008. On the strength of gomory mixed-integer cuts as group cuts. *Mathematical Programming* **115** 387–407.

Dash, S., O. Günlük, A. Lodi. 2010. Mir closures of polyhedral sets. *Mathematical Programming* **121** 33–60.

Dash, S., O. Günlük, M. Molinaro. 2012b. On the relative strength of different generalizations of split cuts. IBM Technical Report RC25326, IBM, Yorktown Heights, NY.

Dey, S. S., A. Lodi, A. Tramontani, L. A. Wolsey. 2010. Experiments with two row tableau cuts. F. Eisenbrand, F. B. Shepherd, eds., *IPCO*, *Lecture Notes in Computer Science*, vol. 6080. Springer, 424–437.

Dey, S. S., A. Tramontani. 2009. Recent developments in multi-row cuts. *Optima* **80** 2–8.

Dey, S. S., L. A. Wolsey. 2010a. Constrained infinite group relaxations of MIPs. *SIAM Journal on Optimization* **20** 2890–2912.

Dey, S. S., L. A. Wolsey. 2010b. Two row mixed-integer cuts via lifting. *Mathematical Programming* **124** 143–174.

Espinoza, D. G. 2010. Computing with multi-row gomory cuts. *Operations Research Letters* **38** 115–120.

Fischetti, M., A. Lodi, A. Tramontani. 2011. On the separation of disjunctive cuts. *Mathematical Programming* **128** 205–230.

Fischetti, M., D. Salvagnin. 2011. A relax-and-cut framework for gomory mixed-integer cuts. *Mathematical Programming Computation* **3** 79–102.

Fischetti, M., C. Saturni. 2007. Mixed-integer cuts from cyclic groups. *Mathematical Programming* **109** 27–53.

Fukasawa, R., M. Goycoolea. 2011. On the exact separation of mixed integer knapsack cuts. *Mathematical programming* **128** 19–41.

Fukasawa, R., O. Günlük. 2011. Strengthening lattice-free cuts using non-negativity. *Discrete Optimization* **8** 229–245.

Li, Y., J-P. P. Richard. 2008. Cook, Kannan and Schrijvers's example revisited. *Discrete Optimization* **5** 724–734.

Louveaux, Q., L. Poirrier. 2012. An algorithm for the separation of two-row cuts. *Mathematical Programming* 1–36.

Online Supplement

Table 6 Gap closed with cross cuts on MIPLIB problems

problem	GMI	DG	Split	Cross	Best Split	Improvement	time (sec)
air04	7.42	9.54	25.00	25.00	91.23	-	10,812
air05	4.64	8.49	20.43	20.43	61.98	-	10,827
arki001	29.15	29.97	32.53	39.88	83.05	-	500
bell5	14.53	25.90	86.19	99.80	92.95	97.16	301
blend2	16.36	30.09	35.02	45.95	46.52	-	6,219
cap6000	41.65	62.43	63.55	66.14	65.17	2.78	10,904
danoint	1.74	1.74	4.61	4.72	8.20	-	10,870
gesa3	45.87	90.88	94.93	96.59	95.81	18.62	11,018
gesa3o	50.57	91.28	95.15	97.38	95.20	45.42	11,266
gt2	67.72	96.61	96.73	99.15	98.38	47.53	206
harp2	8.69	52.03	62.71	62.71	67.31	-	10,809
l152lav	1.55	30.44	77.32	82.37	95.20	-	12,176
lseu	48.42	76.43	84.08	93.88	93.75	2.08	749
mas74†	6.67	8.55	10.31	12.43	14.02	-	10,856
mas76	6.42	10.70	18.84	29.08	26.52	3.48	9,532
misc03	7.24	21.44	28.02	36.82	51.70	-	11,188
misc07	0.72	1.61	4.45	9.45	20.11	-	10,815
mkc	1.21	4.14	51.48	51.48	49.30	4.30	10,805
mod011	17.11	27.52	47.38	47.38	72.44	-	10,801
modglob	15.10	81.26	88.25	99.61	92.18	95.01	10,987
p0033	54.60	84.09	86.19	10.00	87.42	10.00	39
p0201	18.24	69.53	73.94	96.99	74.93	87.99	10,843
pp08a	52.88	94.07	95.12	97.83	97.03	26.94	10,802
pp08aCUTS†	30.07	83.44	92.38	95.48	95.81	-	10,863
qiu	4.21	21.76	78.09	78.09	77.51	2.58	13,014
rentacar	29.05	39.87	46.91	46.91	44.95	3.56	10,847
rout	0.32	23.70	54.90	58.61	70.70	-	11,042
set1ch	38.11	86.69	87.57	96.89	89.74	69.69	10,806
seymour	7.15	11.93	31.05	31.05	61.52	-	10,863
swath	17.66	33.96	34.01	34.06	33.96	0.15	10,889
vpm2†	12.58	68.34	74.86	79.51	81.05	-	10,810
Average	21.21	44.47	57.48	62.44	68.89	19.59	5,979