Math. Prog. Comp. (2015) 7:113–148
DOI 10.1007/s12532-014-0076-9

FULL LENGTH PAPER

# The strength of multi-row models

**Quentin Louveaux · Laurent Poirrier · Domenico Salvagnin**

Received: 27 February 2013 / Accepted: 3 September 2014 / Published online: 25 September 2014


**Abstract** We develop a method for computing facet-defining valid inequalities for any mixed-integer set $P_J$. While our practical implementation does not return only facet-defining inequalities, it is able to find a separating cut whenever one exists. The separator is not comparable in speed with the specific cutting-plane generators used in branch-and-cut solvers, but it is general-purpose. We can thus use it to compute cuts derived from any reasonably small relaxation $P_J$ of a general mixed-integer problem, even when there exists no specific implementation for computing cuts with $P_J$. Exploiting this, we evaluate, from a computational perspective, the usefulness of cuts derived from several types of multi-row relaxations. In particular, we present

D. Salvagnin was supported by the University of Padova (Progetto di Ateneo 325 "Exploiting randomness in Mixed Integer Linear Programming"), and by MiUR, Italy (PRIN project "Mixed-Integer Nonlinear Optimization: Approaches and Applications").

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with its authors. Laurent Poirrier was supported by Progetto di Eccellenza 2008–2009 of the Fondazione Cassa Risparmio di Padova e Rovigo, Italy, by NSERC Discovery Council Grant Number RGPIN-37 1937–2009, and by Early Researcher Award number ER11-08-174.

Q. Louveaux
Montefiore Institute, University of Liège, Liège, Belgium
e-mail: q.louveaux@ulg.ac.be

L. Poirrier (✉)
Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Canada
e-mail: lpoirrier@uwaterloo.ca

D. Salvagnin
Department of Information Engineering, University of Padova, Padua, Italy
e-mail: domenico.salvagnin@unipd.it

results with four different strengthenings of the two-row intersection cut model, and multi-row models with up to fifteen rows. We conclude that only fully-strengthened two-row cuts seem to offer a significant advantage over two-row intersection cuts. Our results also indicate that the improvement obtained by going from models with very few rows to models with up to fifteen rows may not be worth the increased computing cost.

**Mathematics Subject Classification** 90C11 (Mixed integer programming)

## 1 Overview

In the last years, there has been a renewed interest in the MIP research community for finding new ways to compute general-purpose cutting-planes. Specifically, one of the subjects of attention was the generation of so-called multi-row cutting-planes, i.e. inequalities that are valid for relaxations with multiple rows of the problem to be solved.

In this paper, we aim at evaluating accurately the strength of the multi-row relaxations that are used to generate these cutting planes. To that end, we develop techniques to compute the cuts derived from any given relaxation. In other words, we consider the question of separating valid inequalities from arbitrary mixed-integer sets.

Our work is computational in that we take our models from libraries of mixed-integer problem instances that are widely used for benchmarking purposes, then numerically generate cuts for these models. It is also theoretical in that we do not compute cuts to solve problems faster overall, but rather to evaluate whether they could help solve problems faster *if* we had an efficient separator for the associated relaxation. In this perspective, we build a generic separator to identify those relaxations for which trying to develop a specific separator would be worthwhile.

Specifically, we study five types of relaxations of a mixed-integer problem, all of them function of a basis of a linear programming relaxation. While we will later define these relaxations precisely in a systematic way, we start by presenting why each of them is interesting to us.

The first relaxation is the $m$-row intersection cut model

$$P_I := \{(x, s) \in \mathbb{Z}^m \times \mathbb{R}^n_+ : x = f + Rs\}$$

introduced by Balas [7], and further characterized in the two-row case (i.e. for $m = 2$) by Andersen, Louveaux, Weismantel and Wolsey [3] and Cornuéjols and Margot [17]. It consists in dropping, from the original MIP, the integrality constraints on nonbasic variables and the bounds on basic variables. Furthermore, only one bound on the nonbasic variables is included in the description of $P_I$ (the nonnegativity $s \geq 0$), so for any variable having both a lower and an upper bound, one of them is also dropped (the one that is not binding in the basis, as can be seen by putting the problem in standard form). There is a strong relationship between the valid inequalities for $P_I$ and the lattice-free sets in $\mathbb{Z}^m$. We will see later that this relationship can be used to easily compute cuts from $P_I$. These cuts are called intersection cuts.

The other four relaxations consist in various strengthenings of the model $P_I$, which are built by reintroducing some of the constraints that we dropped earlier. In particular,

Dey and Wolsey [27], Basu et al. [11] and Fukasawa and Günlük [31] considered ways to exploit finite bounds on the basic variables $x$. Specifically, Dey and Wolsey [27] extend the geometric intuition that is characteristic of intersection cuts by introducing the concept of $S$-free set. An $S$-free set is a convex set that does not contain any element of $S$ in its interior. In a way similar to the intersection cut approach, one can generate valid inequalities for the strengthened model

$$P_{S\text{-free}} = \{(x, s) \in S \times \mathbb{R}^n_+ : x = f + Rs\},$$

by considering $S$-free sets instead of lattice-free (i.e. $\mathbb{Z}^m$-free) sets. They show that the resulting inequality is valid if $S$ is the set of integral points in some rational polyhedron. This condition is not restrictive since, in practice, we define $S$ to be the intersection of $\mathbb{Z}^m$ with the bound constraints on $x$. Furthermore, all valid inequalities for $P_{S\text{-free}}$ can be obtained through such intersection cuts [27], and there is a one-to-one correspondence between minimal inequalities for the infinite relaxation of $P_{S\text{-free}}$

$$\left\{(x, s) \in S \times \mathbb{R}^\infty_+ : x = f + \sum_{r \in \mathbb{R}^m} r s_r,\ s \text{ has a finite support}\right\}$$

and maximal $S$-free sets in $\mathbb{R}^m$ [11].

On the other hand, Dey and Wolsey [21,26] and Conforti, Cornuéjols and Zambelli [16] considered the problem of exploiting any integrality constraint on nonbasic variables. They propose a way to, given a valid intersection cut for $P_I$, strengthen the coefficients of the nonbasic integer variables by solving a so-called *lifting* problem. Assuming for simplicity that the original problem is a pure integer programming problem, the lifting yields valid inequalities for

$$\{(x, s) \in \mathbb{Z}^m \times \mathbb{Z}^n_+ : x = f + Rs\}. \tag{1}$$

Dey and Wolsey [21,26] show, in the two-row case, that these inequalities are *extreme* for the infinite model

$$\left\{(x, s) \in \mathbb{Z}^2 \times \mathbb{Z}^\infty_+ : x = f + \sum_{r \in \mathbb{R}^2} r s_r,\ s \text{ has a finite support}\right\} \tag{2}$$

if the initial inequality was extreme for the infinite relaxation of $P_I$. The lifting is unique if the initial intersection cut arises from specific types of lattice-free sets (namely Type-1 and Type-2 triangles [21,26]) and sequence-dependent otherwise. This has been extended with analogous results in the multi-row and $S$-free cases by Conforti, Cornuéjols and Zambelli [16]. Note that (1) is known as a *corner relaxation* of the original feasible set. The corner relaxation was studied in details by Gomory [33,34] and Gomory and Johnson [35].

One can also take advantage of the presence of upper bounds on nonbasic variables. Andersen, Louveaux and Weismantel [2] tackled that case, taking into account an upper bound on one nonbasic variable. This can be seen as considering at a pair of two-row

models arising from adjacent simplex bases. They describe geometrically the facet structure of that relaxation, showing that an additional class of facet-defining valid inequalities then arises, whose coefficients can be read from pentagons in $\mathbb{R}^2$, with a formula that they develop. Although the problem has not been thoroughly studied for more than one upper bound, we will consider here the fully-strengthened model

$$P_{IU} = \{(x, s) \in \mathbb{Z}^m \times \mathbb{R}^n_+ : x = f + Rs, s \leq U\}.$$

As a consequence, we are probably further away from a practical implementation of a separator for that particular model than for the other ones.

Lastly, we consider the model $P_{\text{full}}$ obtained by simultaneously exploiting all the strengthening of $P_I$ mentioned above. It simply consists in an $m$-row relaxation of the original problem, keeping all bounds and integrality constraints on the variables.

We now present some previous computational works that have been performed with multi-row cuts.

Espinoza [28] performed the first extensive computational tests with multi-row intersection cuts. He uses three types of fixed lattice-free sets in $\mathbb{R}^m$. The first is a maximal lattice-free simplex with integer vertices, the $m$-dimensional extension of Type-1 triangles (see e.g. [24] for a classification of the two-dimensional lattice-free sets). The second is a cross polytope, scaled and translated to contain the 0-1 hypercube, so as to also be a maximal lattice-free set. The third is the $m$-dimensional extension of a specific Type-2 triangle. He exploits the integrality of the nonbasic variable through a heuristic method similar to Dey and Wolsey's lifting [21], although without the guarantee of generating facet-defining inequalities for the corresponding strengthened model. He tested the three types of lattice-free sets separately for $m \in \{1, \ldots, 10\}$, on MIP instances from MIPLIB 3 [13] and MIPLIB 2003 [1]. The results of these experiments are promising in that they show that the addition of multi-row cuts can speed-up the overall branch-and-cut by up to 5 % in geometric mean over the testset, for specific types of cuts and values of $m$. However, they also show mixed results in general (i.e. without a priori knowledge of which types of cuts will be successful), with in most cases a slight increase in computing time over the solver's defaults.

Dey et al. [22,23] focused on two-row intersection cuts from Type-2 triangles. They devise a heuristic method for building a Type-2 triangle that is parametric in the model data. The aim is to exploit the problem structure, namely the values of $f$ and $R$ in $P_I = \{(x, s) \in \mathbb{Z}^2 \times \mathbb{R}^n_+ : x = f + Rs\}$, to generate strong cuts that are, in some way, "different" from GMI cuts. The cuts are then strengthened with a variant of the lifting method of Dey and Wolsey [21], to take advantage of the integrality of nonbasic variables. The method is tested on modified versions of the randomly generated multidimensional knapsack instances of Atamtürk [5] [22]. From these experiments, they conclude that some two-row cuts do provide an advantage over GMI cuts, although few of the many two-row cuts generated are eventually helpful. This result emphasizes the importance of the selection of the two-row cuts. Furthermore, their data show that in the presence of many nonbasic integer variables, the performance of both two-row and GMI cuts deteriorates, suggesting a need for different relaxations in that case. To the contrary, the presence in the original problem of the bounds that are dropped in $P_I$ did not seem to significantly impair the effectiveness

of the two-row cuts (compared to when no such bounds are present). In [23], they perform experiments on MIPLIB instances, and conclude that exploiting the integrality of nonbasic variables appears more effective on 2-row split cuts than on Type-2 triangle cuts, on instances with a sparse tableau.

Basu et al. [10] study the case of two-row models $P_I$ for which one of the components of $f$ is integral. This situation typically arises when the optimal basis of the LP relaxation is degenerate, which is frequent in MIP formulations. They present a heuristic algorithm for generating maximal lattice-free Type-1 and Type-2 triangles for such model. They show how to compute an intersection cut from these triangles, and develop a closed-form formula implementing Dey and Wolsey's lifting [21]. Note that the latter formula applies to all Type-1 and Type-2 triangles, even when both components of $f$ are fractional. An alternative closed-form formula is also presented, that exploits nonnegativity of one of the basic variables. Experiments are performed on instances from a slight modification of MIPLIB 3 [37], measuring the percentage of gap closed by the two-row cuts. They conclude that the family of cuts that they test is effective, but not competitive with GMI cuts.

Basu et al. [12] adopt a probabilistic approach to compare the triangle closure of $P_I$ to its split closure. Their work presents theoretical estimates of the relative strength of these two closures, for random $P_I$ models with data following uniform distributions. They use two distinct types of measure for evaluating strength, one showing the gap between the two closures for the "worst" choice of objective function, and the other showing the average gap over nonnegative cost vectors. According to the results, the literature examples illustrating that the triangle closure can be vastly stronger than the split closure are not necessarily pathological extreme cases. However, they also indicate that on average, split cuts are as useful as two-row cuts from triangles.

Louveaux and Poirrier [36] present an exact separator for the two-row intersection cut model $P_I$. They conclude that two-row intersection cuts close significantly more gap than one-row intersection cuts, but one can achieve almost as much when restricting to two-row cuts from split sets, which can be obtained by reading a one-row intersection cut from a linear combination of the two rows.

Dash et al. [20] propose some heuristics to computationally evaluate the strength of cross and crooked cross cuts. Cross cuts are cutting planes obtained by considering several split disjunctions simultaneously and it has been shown that many of the multi-row cutting planes considered in this paper can be obtained using cross (or crooked cross) disjunctions. They also specialize their experiments to finding cross cuts for two-row relaxations of problems of the MIPLIB. In this respect, their experiment is very close to the computational experiments proposed in this paper for two-row cuts as most of the cuts generated here can be obtained using their disjunctions. There are however two main differences. The first difference comes from the fact that our first aim is to evaluate specific models and their strengthenings, which cannot be done explicitly using the disjunctions. The second difference is that, in principle, our approach allows us to generate all separating facet-defining inequalities for $P_{full}$, whereas it is not known whether crooked cross cuts do (the crooked cross cut closure has not been proven yet to be polyhedral [18]).

The work of Fukasawa and Goycoolea [32] does not take place in the context of multi-row cuts. However, it is very related to our work in that they develop an exact

separator for the mixed-integer knapsack (i.e. one-row) model

$$P_1 = \{x \in \mathbb{R}^n : a^t x \leq b,\ l \leq x \leq u,\ x_j \in \mathbb{Z}, \forall j \in J\}.$$

Part of the techniques they develop to speed up separation translate directly to our problem. In particular, they also deploy row-generation to optimize over the polar set of $P_1$, and they lift valid inequalities from tight bounds (see Sect. 3). The objective of their computations is to compare a specific type of cuts alone to all so-called knapsack cuts, i.e. to all valid inequalities for $P_1$ together. The specific cuts they study are obtained through the mixed-integer rounding procedure (MIR [38]), which when applied to tableau rows simply yields GMIs. Their results show that in terms of strengthening of the LP bound, MIR cuts alone achieve almost as much as knapsack cuts.

Finally, we note that we are basically using a oracle to separate cutting planes. In this context, the earlier studies date back to the work of Boyd [14] on Fenchel cuts and of Applegate et al. [4] on cuts outside of the template paradigm for the TSP. The approach was later extended to general MIPs by Chvátal et al. [15]. Our separation model is quite similar to the one in [15] (it is basically its dual), yet there is a key difference in that we choose a normalization condition in the polar space that does not introduce artificial vertices, so that we are guaranteed to get only facet-defining inequalities, without the need for a tilting procedure. On the other hand, as will be described in Sect. 3, we split the separation into two phases for computational reasons, so we need a lifting procedure to obtain facet-defining inequalities in the original space.

A constant among these results is that they underline the impressive power of the classic GMI cuts and their variants. This raises the following question: How strong does a relaxation need to be for the cuts it yields to significantly outperform GMIs? This is one of the questions we will tackle here.

In Sect. 2, we cover the basic tools and notations that we build upon. In Sect. 3, we present the methods that we use to develop our generic separator. We describe our experimental setup in Sect. 4, and the remaining sections detail the results of our tests. Specifically, we show that only fully-strengthened two-row cuts seem to offer a significant advantage over two-row intersection cuts (as opposed to cuts that are partially strengthened through lifting alone, or through the use of S-free sets alone). Our results also indicate that the improvement obtained by going from four-row to fifteen-row cuts may not be worth the increased computing cost.

## 2 Polarity for general polyhedra

We now set aside the question of the relaxation, and focus on the problem of separating cuts for any given relaxation $P_J$. Let $P_J$ be the mixed-integer set

$$P_J = \left\{x \in \mathbb{R}^n : x \in P_{LP},\ x_j \in \mathbb{Z} \text{ for all } j \in J\right\} \tag{3}$$

where $P_{LP}$ is a polyhedron in $\mathbb{R}^n$ and $J$ is a subset of $N = \{1, \ldots, n\}$. Here, we always assume that $P_{LP}$ is a rational polyhedron. We are interested in the valid inequalities for $\text{conv}(P_J)$. For conciseness, we denote $P := \text{conv}(P_J)$. We thus want to characterize

the set of all $(\alpha, \alpha_0) \in \mathbb{R}^{n+1}$ such that $\alpha^T x \geq \alpha_0$ is a valid inequality for $P$:

$$Q := \{(\alpha, \alpha_0) \in \mathbb{R}^{n+1} : \alpha^T x \geq \alpha_0, \text{ for all } x \in P\}. \tag{4}$$

In order to simplify our statements, it will often be useful to assume that $P$ is a polyhedral cone. To show that this assumption can be made without loss of generality, we construct the following "conified" variant of $P$.

**Definition 1** Given a polyhedron $P$, by the Minkowski-Weyl theorem, there exist finite sets $\mathcal{V}$, $\mathcal{R}$, and $\mathcal{L}$ such that

$$P = \text{conv}(\mathcal{V}) + \text{cone}(\mathcal{R}) + \text{lin}(\mathcal{L}).$$

Let $\mathcal{V}^+ := \{(v, -1) \in \mathbb{R}^{n+1} : v \in \mathcal{V}\}$, $\mathcal{R}^+ := \{(r, 0) \in \mathbb{R}^{n+1} : r \in \mathcal{R}\}$, and $\mathcal{L}^+ := \{(l, 0) \in \mathbb{R}^{n+1} : l \in \mathcal{L}\}$. We define $C$ as

$$C := \text{cone}\{\mathcal{V}^+, \mathcal{R}^+\} + \text{lin}\{\mathcal{L}^+\}. \tag{5}$$

Note that if $P$ is pointed, the sets $\mathcal{V}$, $\mathcal{R}$, and $\mathcal{L}$ can be taken as follows: $\mathcal{V}$ is the set of the vertices of $P$, $\mathcal{R}$ is the set of the extreme rays of $P$, and $\mathcal{L}$ is empty. By construction, $C$ is a polyhedral cone generated by the rays $\mathcal{V}^+$, $\mathcal{R}^+$, $\mathcal{L}^+$ and $-\mathcal{L}^+$. The relationship between the descriptions of $P$ and $C$ is very strong, as shown by the following proposition.

**Proposition 1** (a) *$v$ is a vertex of $P$ if and only if $(v, -1)$ is an extreme ray of $C$.*
(b) *$r$ is an extreme ray of $P$ if and only if $(r, 0)$ is an extreme ray of $C$.*
(c) *$l$ is in the lineality space of $P$ if and only if $(l, 0)$ is in the lineality space of $C$.*
(d) *$(\alpha, \alpha_0) \in \mathbb{R}^{n+1}$ describes a valid inequality $\alpha^T x \geq \alpha_0$ for $P$ if and only if $\alpha^T x + \alpha_0 x_0 \geq 0$ is valid for $C$.*
(e) *$(\alpha, \alpha_0) \in \mathbb{R}^{n+1}$ describes a facet-defining inequality $\alpha^T x \geq \alpha_0$ for $P$ if and only if $\alpha^T x + \alpha_0 x_0 \geq 0$ is facet-defining for $C$.*
(f) $P = \{x \in \mathbb{R}^n : (x, -1) \in C\}$

*Proof* See e.g. [40]. □

We are now ready to state the properties of $Q$.

**Proposition 2** (a) *$Q$ is a polyhedral cone, and it is the polar of $C$*
(b) *$C$ is the polar of $Q$.*
(c) *$Q$ is described by*

$$Q = \left\{ (\alpha, \alpha_0) \in \mathbb{R}^{n+1} \; : \; \begin{array}{ll} \alpha^T x \geq \alpha_0 & \text{for all } x \in \mathcal{V} \\ \alpha^T r \geq 0 & \text{for all } r \in \mathcal{R} \\ \alpha^T l = 0 & \text{for all } l \in \mathcal{L} \end{array} \right\}. \tag{6}$$

(d) *Let $Q = \text{cone}(B) + \text{lin}(\Gamma)$, where $\text{lin}(\Gamma)$ is the lineality space of $Q$, and $B$ is a minimal generating set (i.e. $\text{cone}(B \backslash \{b\}) + \text{lin}(\Gamma) \neq Q$ for any $b \in B$). A valid inequality $\alpha^T x \geq \alpha_0$ for $P$ is facet-defining if and only if $\alpha \in \text{cone}(\beta) + \text{lin}(\Gamma)$ for some $\beta \in B$.*

(e) *$Q$ is pointed if and only if $P$ is full-dimensional. In that case, $\alpha^T x \geq \alpha_0$ is facet-defining for $P$ if and only if $(\alpha, \alpha_0)$ is an extreme ray of $Q$.*

*Proof* See e.g. [40]. We refer the interested reader to [42] for more details. □

Let $x^*$ be a point that we want to separate. We are looking for a facet-defining inequality $(\alpha, \alpha_0)$ of $P$ such that $\alpha^T x^* < \alpha_0$. Note that, by the separating hyperplane theorem, such a valid inequality exists if and only if $x^* \notin P$. One can find $(\alpha, \alpha_0)$ by maximizing, over the set of the valid inequalities, the violation at $x^*$. This corresponds to solving

$$\begin{array}{l} \min \boldsymbol{\alpha}^T x^* - \boldsymbol{\alpha}_0 \\ \text{s.t. } (\boldsymbol{\alpha}, \boldsymbol{\alpha}_0) \in Q. \end{array} \tag{7}$$

Throughout this paper, we take the convention that variables are emphasized in bold within optimization problems, so as to distinguish them from constant data. Since $Q$ is a polyhedron, (7) is a linear programming problem. But since it is a cone, (0, 0) will be an optimal solution when $x^* \in P$, and the problem will always be unbounded when a separating inequality exists. Therefore, in order to choose finite solutions of (7), we need to impose some normalization on the cut coefficients. Note that fundamentally, as any cut-generating linear program, (7) only lets us discriminate between separating and nonseparating valid inequalities for $P$. In particular, any measure of violation is dependent on the specific normalization that we adopt.

A common form of normalization is to fix the right-hand side $\alpha_0$ of the cuts. In some specific contexts, it is possible to fix $\alpha_0 = 1$ without loss of generality (see e.g. [36]). However, as we are here in the case of a general polyhedron $P$, we would have to consider a disjunctive constraint of the type $\alpha_0 = 1 \vee \alpha_0 = 0 \vee \alpha_0 = -1$. Indeed, any valid inequality $(\alpha, \alpha_0)$ can be made to fall into one of these three cases by dividing it by $|\alpha_0|$ if $\alpha_0 \neq 0$, but all three cases are necessary in order to cover all the possibilities. Unfortunately, incorporating such disjunctive constraint, the cut-generating problem could no longer be modeled as a single linear programming problem (it could be tackled by solving three LPs however). Note also that $Q \cap \{\alpha_0 = 0\}$ is still a cone, so we would require an additional normalization constraint in the case $\alpha_0 = 0$.

An alternative is to bound the magnitude of the cut coefficients with a constraint of the type

$$\sum_{i=1}^{n} |\alpha_i| \leq 1. \tag{8}$$

The issue with the normalization (8) is that it amounts to intersecting $Q$ with several half-spaces, as is made obvious by the equivalent linear formulation

$$\begin{cases} -\gamma_i \leq \alpha_i \leq \gamma_i, \quad \text{for all } i \\ \displaystyle\sum_{i=1}^{n} \gamma_i \leq 1. \end{cases}$$

When $Q$ is pointed, intersecting it with several half-spaces may yield vertices that do not correspond to an extreme ray of $Q$. Therefore, if we optimize over that truncated cone, we could obtain optimal solutions that do not give facet-defining inequalities.

To overcome these limitations, we adopt a normalization proposed by Balas and Perregaard [8]

$$\alpha^T(\tilde{x} - x^*) = 1 \tag{9}$$

where $\tilde{x}$ is an arbitrary point that belongs to the convex hull of $P$. That normalization is linear, and consists in intersecting $Q$ with a single hyperplane. As we now demonstrate, it makes the optimization problem bounded, while not cutting away any relevant solutions.

**Proposition 3** *The optimization problem*

$$\begin{aligned} \bar{s} = \min\ & \boldsymbol{\alpha}^T x^* - \boldsymbol{\alpha}_0 \\ \text{s.t. } & (\boldsymbol{\alpha}, \boldsymbol{\alpha}_0) \in Q \\ & \boldsymbol{\alpha}^T(\tilde{x} - x^*) = 1 \end{aligned} \tag{10}$$

*is always bounded. In particular, if* $(\bar{\alpha}, \bar{\alpha}_0)$ *is an optimal solution and* $\bar{s}$ *its objective function value, then* $\bar{s} \geq -1$*, which means that the violation of* $(\bar{\alpha}, \bar{\alpha}_0)$ *at* $x^*$ *is at most one.*

*Proof* As $\tilde{x} \in P$, $\alpha^T\tilde{x} \geq \alpha_0$ is a valid constraint for $Q$. The normalization constraint (9) gives $\alpha^T\tilde{x} = 1 - \alpha^T x^*$, which lets us replace $\alpha^T\tilde{x}$ in the previous inequality, and obtain $\alpha^T x^* - \alpha_0 \geq -1$. □

Remark that $\bar{s} = -1$ when the optimal solution corresponds to an inequality that is tight at $\tilde{x}$, because $\alpha^T x^* = \alpha_0 - 1$ if and only if $\alpha^T\tilde{x} = \alpha_0$.

We now show that the addition of the normalization constraint does not remove any interesting valid inequality from the feasible region of the optimization problem (10).

**Proposition 4** *Given any valid inequality* $(\alpha, \alpha_0) \in Q$ *separating* $x^* \notin P$*, there exists* $\lambda > 0$ *such that* $(\lambda\alpha, \lambda\alpha_0) \in Q$ *separates* $x^*$ *and satisfies* $(\lambda\alpha)^T(\tilde{x} - x^*) = 1$*, for any* $\tilde{x} \in P$*.*

*Proof* By hypothesis, $\alpha^T\tilde{x} \geq \alpha_0$ and $\alpha^T x^* < \alpha_0$. Thus $\alpha^T\tilde{x} > \alpha^T x^*$, and $\alpha^T(\tilde{x} - x^*) > 0$. The claim follows, by letting $\lambda = 1/(\alpha^T(\tilde{x} - x^*))$. □

**Corollary 1** *The optimization problem* (10) *may be infeasible only if* $x^* \in P$*.*

One motivation for our choice of normalization is that, when $P$ is full-dimensional (i.e. when $Q$ is pointed), we can obtain facet-defining inequalities for $P$ (i.e. extreme rays of $Q$) by using the simplex method. We now show that this result holds even if $P$ is not full-dimensional.

**Theorem 1** *Let* $P$ *be a polyhedron in* $\mathbb{R}^n$*. If* $x^* \in \text{aff}(P) \setminus \text{conv}(P)$ *and* $(\alpha, \alpha_0)$ *describes a valid inequality separating* $x^*$ *obtained by optimizing over the linear problem* (10) *with the simplex method, then* $\alpha^T x \geq \alpha_0$ *is facet-defining for* $P$*.*

To prove Theorem 1, we first need Lemmas 1, 2 and 3.

**Lemma 1** *Let $C$ be a polyhedral cone in $\mathbb{R}^m$ with $e_k \notin \text{aff}(C)$, and $C_{\{k\}} := C + \text{lin}(e_k)$. If the inequality $\alpha^T x \geq 0$ is facet-defining for $C_{\{k\}}$, then it is facet-defining for $C$.*

*Proof* First note that $\alpha_k = 0$. Indeed, since $\alpha^T x \geq 0$ defines a proper face of $C_{\{k\}}$, there exists $w \in C_{\{k\}}$ such that $\alpha^T w = 0$. Therefore, $\alpha^T(w + \mu e_k) \geq 0$, so $\alpha^T \mu e_k \geq 0$, for all $\mu \in \mathbb{R}$.

Now let $d = \dim(C_{\{k\}})$. Because $e_k \notin \text{aff}(C)$, $d = \dim(C) + 1$. Since $C \subseteq C_{\{k\}}$, $\alpha^T x \geq 0$ is valid for $C$. Since $\alpha^T x \geq 0$ is facet-defining for $C_{\{k\}}$, there exist $d$ affinely independent points $y^0, \ldots, y^{d-1}$ in $C_{\{k\}} \cap \{x \in \mathbb{R}^m : \alpha^T x = 0\}$. We assume without loss of generality that $y^0 = 0$. The $d-1$ points $y^1, \ldots, y^{d-1}$ are linearly independent, so the matrix

$$Y := \left[ y^1 | \cdots | y^{d-1} \right]$$

is full column rank (i.e. $\text{rank}(Y) = d - 1$). For each point $y^i$, one can construct $z^i = y^i + \mu_i e_k$ with $\mu_i \in \mathbb{R}$ such that $z^i \in C$. The matrix

$$Z := \left[ z^1 | \cdots | z^{d-1} \right]$$

differs from $Y$ only in the $k$th row, so its rank is at least $d - 2$. As $\alpha_k = 0$, $\alpha^T z^i = \alpha^T y^i = 0$ for all $i$. Hence, letting $z^0 = 0$, we know $d-1$ affinely independent $z^i$ points in $C \cap \{x \in \mathbb{R}^m : \alpha^T x = 0\}$, which is thus of dimension at least $d - 2 = \dim(C) - 1$.

We finally show that it is of dimension exactly $\dim(C) - 1$. Indeed, otherwise $\alpha^T x = 0$ for all $x \in C$, implying that $\alpha^T x = 0$ for all $x \in C_{\{k\}}$ (recall that $\alpha_k = 0$), which contradicts the hypothesis that $\alpha^T x \geq 0$ is facet-defining for $C_{\{k\}}$. □

**Lemma 2** *Let $C_K := C + \text{lin}(\{e_k : k \in K\})$ be a full-dimensional polyhedral cone in $\mathbb{R}^m$ such that $\dim(C + \text{lin}(\{e_k : k \in K,\ k \neq \bar{k}\})) < m$ for any $\bar{k} \in K$. If $\alpha^T x \geq 0$ is facet-defining for $C_K$, then it is facet-defining for $C$.*

*Proof* Because $\dim(C_{K\setminus\{\bar{k}\}}) < \dim(C_K)$, we know that $e_{\bar{k}} \notin \text{aff}(C_{K\setminus\{\bar{k}\}})$, for all $\bar{k} \in K$. Furthermore, $e_{\bar{k}} \notin \text{aff}(C_{K\setminus\bar{K}})$, for all $\bar{k} \in \bar{K} \subseteq K$. Let $\kappa := |K|$, $K = \{k^1, \ldots, k^\kappa\}$, and $K^p := \{k^1, \ldots, k^p\}$ where $0 \leq p \leq \kappa$. The proof works by backward induction on $p$. By hypothesis, $\alpha^T x \geq 0$ is facet-defining for $C_{K^\kappa}$. By Lemma 1, if it is facet-defining for $C_{K^p}$, then it is also facet-defining for $C_{K^{p-1}}$. Finally we obtain that $\alpha^T x \geq 0$ is facet-defining for $C_{K^0} = C$. □

**Lemma 3** *Let $Q$ be a polyhedral cone in $\mathbb{R}^m$ and $H$ the hyperplane $\{\alpha \in \mathbb{R}^m : \pi^T \alpha = 1\}$. If $Q \cap H$ is pointed and $\alpha^*$ is a vertex of $Q \cap H$, then*

(a) $\dim(\text{lin.space}(Q)) \in \{0, 1\}$,
(b) *if* $\dim(\text{lin.space}(Q)) = 1$, *then $\alpha^*$ is on the lineality direction of $Q$,*
(c) *if* $\dim(\text{lin.space}(Q)) = 0$, *then $\alpha^*$ is an extreme ray of $Q$.*

*Proof* (a) We contradict $\dim(\text{lin.space}(Q)) \geq 2$. Let $l^1$ and $l^2$ be two distinct vectors of a basis of the linear space of $Q$. Both $\pi^T l^1 \neq 0$ and $\pi^T l^2 \neq 0$, otherwise $Q \cap H$ would not be pointed. Let $\mu^1, \mu^2 \in \mathbb{R}$ be such that $\pi^T \mu^1 l^1 = 1$ and $\pi^T \mu^2 l^2 = 1$. Because $l^1$ and $l^2$ are distinct vectors of a basis, $\mu^1 l^1 \neq \mu^2 l^2$. We can construct $l' = \mu^1 l^1 - \mu^2 l^2$ in the lineality space of $Q$ with $\pi^T l' = 0$, which contradicts the fact that $Q \cap H$ is pointed. (b) Let $l$ be the lineality direction of $Q$. Again, $\pi^T l \neq 0$, otherwise $Q \cap H$ would not be pointed. Let $\mu \in \mathbb{R}$ be such that $\pi^T \mu l = 1$. As $\alpha^*$ is a ray of $Q$, we can construct $\alpha^1 := \frac{3}{2}\alpha^* - \frac{1}{2}\mu l$ and $\alpha^2 := \frac{1}{2}\alpha^* + \frac{1}{2}\mu l$. One can easily verify that $\alpha^1, \alpha^2 \in Q \cap H$, and that $\alpha^* = \frac{1}{2}\alpha^1 + \frac{1}{2}\alpha^2$, contradicting the fact that $\alpha^*$ is a vertex of $Q \cap H$. (c) Standard. □

*Proof of Theorem 1* To simplify the proof, we consider that we are looking for a valid inequality $\alpha^T r \geq 0$ for $C$, and rewrite (10) as

$$\begin{aligned} \min\ & \boldsymbol{\alpha}^T r^* \\ \text{s.t. } & \boldsymbol{\alpha} \in Q \\ & \boldsymbol{\alpha}^T (\tilde{r} - r^*) = 1 \end{aligned} \tag{11}$$

where $r^* = (x^*,\ 1)$ and $\tilde{r} = (\tilde{x},\ 1)$. By Proposition 3, we know that the linear programming problem (11) has a finite optimal objective function value. If the feasible region is not pointed, then in the solution $\alpha^*$ returned by the simplex method, some nonbasic free variables must be fixed at an arbitrary value (typically zero), with a zero reduced cost, and could not enter the basis at a finite value. Let us denote by $K$ the index set of such variables, i.e. $\alpha^*_k$ is nonbasic and fixed to zero for all $k \in K$. We claim that $\alpha^*$ is a vertex of the pointed polyhedron

$$Q_{*K} := \left\{ \alpha \in Q : \alpha^T (\tilde{r} - r^*) = 1 \quad \text{and} \quad \alpha_k = 0,\ \text{for all } k \in K \right\}$$

obtained by intersecting the feasible region with $\{\alpha_k = 0,\ \text{for all } k \in K\}$. Indeed, it corresponds to a basic feasible solution for that polyhedron (in the classical sense of a basic feasible solution, i.e. where all free variables are basic). Note that since none of the nonbasic free variables could enter the basis at a finite value, $Q_{*K\setminus\{k\}}$ is not pointed, for any $k \in K$. Removing the normalization constraint, we obtain

$$Q_K := \{\alpha \in Q : \alpha_k = 0,\ \text{for all } k \in K\}$$

which, by Lemma 3 has a lineality space of dimension zero or one. Lemma 3 also specifies that the point $\alpha^*$ is either an extreme ray of $Q_K$ or a lineality direction of $Q_K$. The latter would imply that $\alpha^{*T} x = 0$ is a valid equality for $C$ that separates $r^*$, contradicting the hypothesis that $r^* \in \text{aff}(C)$. Hence, $Q_K$ is pointed and $\alpha^*$ is an extreme ray of $Q_K$.

Now observe that $Q_K$ is the polar of

$$C_K = C + \text{lin}(\{e_k : k \in K\})$$

which is full-dimensional since $Q_K$ is pointed. As $\alpha^*$ is an extreme ray of $Q_K$, ${\alpha^*}^T x \geq 0$ is facet-defining for $C_K$. By Lemma 2, it is also facet-defining for $C$.

□

Note that we consider that to each vertex of the feasible region of a linear program, corresponds at least one basis. This is not true with textbook descriptions of the simplex method when there are redundant constraints. It is true however with most practical implementations, where the standard form typically includes one logical variable for each tableau row (that variable is fixed to zero in phase II).

Remark also that a specific implementation of the simplex method may not ensure that, as required by the above proof, none of the nonbasic free variables could enter the basis at a finite value. However, enforcing this condition can easily be implemented as a post-processing, whenever finding facet-defining inequalities is desirable.

## 3 Towards a generic MIP separator

We have assumed so far that we have an inner description of $P = \text{conv}(P_J)$. However, this assumption does not hold in practice since, in the context of mixed-integer programming, we are typically provided with a description of $P_J$ as $P_J := P_{LP} \cap \{x_J \in \mathbb{Z}\}$, where $P_{LP} := \{x \in \mathbb{R}^n_+ : Ax = b\}$ is a *linear relaxation* of $P_J$. To overcome this issue, we adopt a classic row-generation approach, where we optimize over a relaxation $Q(S, T)$ of $Q$ and iteratively strengthen this relaxation by adding to it constraints of $Q$ that are violated by its incumbent optimal solution.

Given finite sets $S \subseteq P_J$ and $T \subseteq \text{recc}(\text{conv}(P_J))$, we start with the partial description $Q(S, T)$ of $Q$ given by

$$Q(S,T) := \left\{ (\alpha, \alpha_0) \in \mathbb{R}^{n+1} \;\middle|\; \begin{array}{ll} \alpha^T x^i \geq \alpha_0 & \forall x^i \in S \\ \alpha^T r^j \geq 0 & \forall r^j \in T \end{array} \right\}, \tag{12}$$

which is easily seen to be a relaxation of (6). Note that a single vector $l$ in the lineality space of $P$ (i.e. $l \in \mathcal{L}$ in (6)) will be represented by two constraints, namely $-l, l \in T$ in (12). We can now find candidate (i.e. possibly invalid) inequalities separating $x^*$ by solving the optimization problem

$$\begin{array}{l} \min \boldsymbol{\alpha}^T x^* - \boldsymbol{\alpha}_0 \\ \text{s.t. } (\boldsymbol{\alpha}, \boldsymbol{\alpha}_0) \in Q(S,T) \\ \quad \boldsymbol{\alpha}^T (\tilde{x} - x^*) = 1 \end{array} \tag{13}$$

which we call *master* problem. Given a candidate inequality $(\bar{\alpha}, \bar{\alpha}_0)$, solving the *slave* mixed-integer problem

$$\begin{array}{l} \min \bar{\alpha}^T \boldsymbol{y} \\ \text{s.t. } \boldsymbol{y} \in P_J \end{array} \tag{14}$$

shows whether $(\bar{\alpha}, \bar{\alpha}_0)$ is valid. In particular, if (14) has an optimal solution $y^*$ such that $\bar{\alpha}^T y^* \geq \alpha_0$, then we know that $(\bar{\alpha}, \bar{\alpha}_0)$ is a valid inequality for $P_J$. If $\bar{\alpha}^T y^* < \alpha_0$, then $(\bar{\alpha}, \bar{\alpha}_0)$ is not a valid inequality for $P_J$. In that case, since $y^* \in P_J$, we can add

the constraint $\alpha^T y^* \geq \alpha$ to $Q(S,T)$, which is currently violated by $(\bar{\alpha}, \bar{\alpha}_0)$. This amounts to appending $\{y^*\}$ to the set $S$. Finally, if the slave is unbounded, there must exist a mixed-integer infinite direction $r^*$ of $P_J$ such that $\bar{\alpha}^T r^* < 0$. We thus add to $Q(S,T)$ the constraint $\alpha^T r^* \geq 0$, which is valid for $Q$ and cuts off $(\bar{\alpha}, \bar{\alpha}_0)$. In other words, we append $\{r^*\}$ to $T$.

For technical reasons, MIP solvers are not well-suited for finding mixed-integer infinite directions in problems. However, such direction can be found easily by optimizing over the linear programming relaxation $\min\{\bar{\alpha}^T \boldsymbol{y} : \boldsymbol{y} \in P_{LP}\}$ of (14). Indeed, the simplex method lets us find rays of $P_{LP}$, and Lemma 4 shows that $P_{LP}$ and $\text{conv}(P_J) \neq \emptyset$ have the same recession cone.

**Lemma 4** *Let $P_J$ be nonempty, $P_{LP}$ be a linear relaxation of $P_J$ (i.e. $P_J = P_{LP} \cap \{x_J \in \mathbb{Z}\}$) and $r$ be a rational direction, the following statements are equivalent:*

1. *There exists $\lambda > 0$ such that $\lambda r$ is a mixed-integer infinite direction of $P_J$.*
2. *$r$ is a ray of* $\text{conv}(P_J)$.
3. *$r$ is a ray of $P_{LP}$.*

*Proof* $(1 \Rightarrow 3)$ This follows from $P_J \subseteq P_{LP}$. $(3 \Rightarrow 1)$ Since $r$ is rational, there exists $\lambda > 0$ such that $\lambda r_j \in \mathbb{Z}$ for all $j \in J$. Thus for every $x \in P_J$ and $y = x + \lambda r$, $y \in P_{LP}$ and $y_j \in \mathbb{Z}$ for all $j \in J$. Therefore $y \in P_J$. $(1 \Rightarrow 2)$ This follows from $P_J \subseteq \text{conv}(P_J)$. $(2 \Rightarrow 3)$ This follows from $\text{conv}(P_J) \subseteq P_{LP}$. □

Note that as long as $\tilde{x} \in S$, the master problem (13) is always bounded, even when $S = \{\tilde{x}\}$ and $T = \emptyset$. Indeed, the proof of Proposition 3 equally applies to the problem

$$\begin{aligned} \bar{s} = \min\ & \boldsymbol{\alpha}^T x^* - \boldsymbol{\alpha}_0 \\ \text{s.t. } & \boldsymbol{\alpha}^T \tilde{x} \geq \boldsymbol{\alpha}_0. \\ & \boldsymbol{\alpha}^T (\tilde{x} - x^*) = 1 \end{aligned} \tag{15}$$

The same row-generation method is used by Perregaard and Balas [39], except that they limit to linear programming slaves, in order to obtain fast separation. As the objective here is to have an exact separator, we do not consider such a relaxation.

An overview of the method is presented in Algorithm 1. The separation procedure computes a point of $P_J$ or a ray of $P_{LP}$ at each iteration. That involves solving a linear programming problem, then a mixed-integer problem whenever the former is bounded, so the resulting procedure is naturally slow in practice, but can be implemented. Although we do not aim at having a separator that is fast enough to be useful for speeding up the resolution of MIPs, preliminary computational results on a small subset of tiny instances clearly showed that computing times involved can be huge, making the cost of the computation prohibitive on any reasonable set of medium-sized benchmark instances. It is therefore necessary to make our separator faster.

A first possible way is to take advantage of the fact that the point $x^*$ typically has many components that are at one of their bounds in the formulation of $P_{LP}$. In such a case, it is possible to first focus on the face $\tilde{F} = P \cap \{x_j = x_j^*,$ for all $j$ such that $x_j^*$ is at a bound$\}$ of $P$, and find a valid inequality for $\tilde{F}$ separating $x^*$. One can then *lift* that inequality in order for it to be valid for the whole set $P$. These ideas were exploited by Applegate et al. [4] and Perregaard and Balas [39].

| | |
|---|---|
| **Step 0** | Input: a set $P_J$ and a point $x^*$ to separate. |
| | Solve a feasibility MIP over $P_J$. |
| | Let $\tilde{x} \in P_J$ be the solution found. |
| | Initialize $S := \{\tilde{x}\}$ and $T := \emptyset$. |
| **Step 1** | Solve the master problem (13). |
| | Let $(\bar{\alpha}, \bar{\alpha}_0)$ be an optimal solution (if such solution exists). |
| | If the master problem is infeasible or if $\bar{\alpha}^T x^* \geq \bar{\alpha}_0$: |
| | There is no valid inequality for $P_J$ separating $x^*$. Stop |
| **Step 2** | Solve the slave problem (14). |
| | If the slave problem is infeasible: |
| | Any inequality is valid for $P_J$. Consider $0 \geq 1$ and stop. |
| | If the slave problem is unbouned: |
| | Let $r$ be an extreme ray of its LP relaxation $P_{LP}$ that shows unboundedness (i.e. $\bar{\alpha}^T r < 0$). |
| | $T := T \cup \{r\}$. Go to Step 1. |
| | If the slave problem has an optimal solution $y^*$: |
| | If $\bar{\alpha}^T y^* < \bar{\alpha}_0$: |
| | $S := S \cup \{y^*\}$. Go to Step 1. |
| | If $\bar{\alpha}^T y^* \geq \bar{\alpha}_0$: |
| | $\bar{\alpha}^T x \geq \bar{\alpha}_0$ is a valid inequality for $P_J$ separating $x^*$. Stop. |

Algorithm 1: Row-generation algorithm for optimizing over $Q$

The next proposition shows, in the case of one component at a bound, that the lifting is always feasible and yields a valid inequality for $P$ that also separates $x^*$. The conclusion applies to any number of bounds by using Proposition 5 repeatedly, forming a feasible sequential lifting. Note that the bound constraint is presented in general form $f^T x \geq f_0$ to handle lower or upper bounds equivalently.

**Proposition 5** *Given $P = \text{conv}(\{x^i\}) + \text{cone}(\{r^j\}) \neq \emptyset$, consider an inequality $f^T x \geq f_0$ that is valid for $P$ and let $F := P \cap \{x : f^T x = f_0\}$. If $F \neq \emptyset$ and $\alpha^T x \geq \alpha_0$ is a valid inequality for $F$, then there always exists a finite coefficient $\mu$ such that $\alpha x + \mu(f^T x - f_0) \geq \alpha_0$ is valid for $P$.*

*Proof* See [6]. □

Note that if $\alpha^T x \geq \alpha_0$ is facet-defining for $F$, then $\alpha x + \mu(f^T x - f_0) \geq \alpha_0$ is facet-defining for $P$ (see [40]).

In case $F$ is empty, i.e. if it is not a proper face of $P$, then the next proposition provides a similar result, showing again how to lift to obtain a valid inequality for $P$.

**Proposition 6** *If $F = \emptyset$, there always exists a finite coefficient $\mu$ such that $\mu(f^T x - f_0) \geq 1$ is valid for $P$, i.e. we can always lift the inequality $0^T x \geq 1$.*

*Proof* Since $P = \text{conv}(\{x^i\}) + \text{cone}(\{r^j\})$, an inequality $\gamma^T x \geq \gamma_0$ is valid for $P$ if and only if $\gamma^T x^i \geq \gamma_0$ for all $i$ and $\gamma^T r^j \geq 0$ for all $j$. Let us consider the vertices first. Since $F = \emptyset$, it cannot be that $f^T x^i = f_0$, otherwise $x^i$ would be a feasible point in $F$. So we must have $f^T x^i > f_0$, and hence the condition

$$\mu \geq \frac{1}{f^T x^i - f_0}$$

whose right-hand side is finite. Let us next consider the rays. We must find $\mu$ such that $\mu f^T r^j \geq 0$. Since $f^T r^j \geq 0$, we get the condition $\mu \geq 0$ (the same per all rays). Taking the largest right-hand side of the constraints on $\mu$ proves the claim. □

In general, if $f^T x \geq f_0, \forall x \in P$, Proposition 5 shows that an inequality $(\alpha, \alpha_0)$ valid only for the nonempty set $P \cap \{f^T x = f_0\}$ can always be lifted so as to be valid for $P$. Proposition 6 indicates that this is also true when $P \cap \{f^T x = f_0\}$ is empty provided that $\alpha = 0$ and $\alpha_0 = 1$. While any inequality is valid for the empty set, we proved that for the special choice of the infeasible constraint $0 \geq 1$, the lifting problem is always feasible. Note however that when we lift $0 \geq 1$, the lifted inequality is not facet-defining in general. If one needs facet-defining inequalities, then lifting-based procedures must be disabled whenever $F = \emptyset$.

In all cases, if $f^T x^* = f_0$, then the slack of the lifted inequality at $x^*$,

$$\alpha^T x^* + \mu(f^T x^* - f_0) - \alpha_0$$

is equal to the slack of the initial inequality at $x^*$,

$$\alpha^T x^* - \alpha_0.$$

In other words, lifting a valid inequality that is violated at $x^*$ from a face that is tight at $x^*$ yields a new valid inequality that is violated by the same amount at $x^*$.

We can now apply these concepts to design a two-phase process. Let

$$P := \{x \in \mathbb{R}^n : Ax = b,\ l \leq x \leq u,\ x_J \in \mathbb{Z}\}$$

and $N$ be the index set of components $x^*$ that are at a bound, i.e. either $x^*_j = l_j$ or $x^*_j = u_j$, for all $j \in N$. We denote by $B$ the complement set, $B := \{1, \ldots, n\} \backslash N$. Note that since $x^*$ is an arbitrary point, $B$ does not necessarily index a basis.

In a first phase, we find a valid inequality $(\bar{\alpha}_B, \bar{\alpha}_0)$ for $F := P \cap \{x_N = x^*_N\}$ using Algorithm 1. If no such inequality exists, then we can already stop because $x^*_B \in F \subseteq P$, which means that there is no valid inequality for $P$ that separates $x^*$. In the special case where $F$ is empty, we consider $(\bar{\alpha}_B, \bar{\alpha}_0) = (0, 1)$, as Proposition 6 suggests.

In a second phase, we find a valid inequality $(\alpha_N, \alpha_B, \alpha_0)$ for $P$, again with Algorithm 1. But now, we can fix $\alpha_B = \bar{\alpha}_B$ and $\alpha_0 = \bar{\alpha}_0 + \alpha^T_N x^*_N$. The above reasoning guarantees that a feasible solution exists for the coefficients $\alpha_N$ and that the violation at $x^*$ of the initial cut obtained is conserved through the second phase. Remark also that in the second phase, the normalization constraint (9) is always satisfied, hence redundant.

Our proposed approach is summarized in Algorithm 2, where the inequalities $(\alpha, \alpha_0)$ in Phases 1 and 2 are found using Algorithm 1.

In our preliminary tests, the two-phase process yielded a speedup of about 6.0 over Algorithm 1, on a benchmark testset of 12 small MIPLIB instances. This can be explained by two factors. Firstly, whenever $x^* \in P$, we can interrupt the computation at the end of Phase 1, which should be fast as it operates with smaller slaves. Secondly,

| | |
|---|---|
| **Init** | Let $N := \{j \in \{1, \ldots, n\} : x_j^* = l_j \text{ or } x_j^* = u_j\}$ and $B = \{1, \ldots, n\} \backslash N$ |
| **Phase 1** | Let $P_1 := P \cap \{x_N = x_N^*\}$. |
| | If $P_1 = \emptyset$, |
| | $(\bar{\alpha}_B, \bar{\alpha}_0) := (0, 1)$. Go to Phase 2. |
| | Find a valid inequality $(\bar{\alpha}_B, \bar{\alpha}_0)$ for $P_1$. |
| | If no such inequality exist, |
| | No valid inequality for $P$ separates $x^*$. Stop. |
| **Phase 2** | Find a valid inequality $(\alpha_N, \bar{\alpha}_B, \alpha_0)$ for $P$. |

Algorithm 2: Two-phase process

the structure of the Phase 2 master may be simpler, as it has fewer variables. The bottleneck of the computation is solving the slave MIPs, but due to the simpler structure, Algorithm 1 may require fewer iterations than would be necessary without Phase 1. The cost of the two-phase process is that we lose some degrees of freedom in choosing the separating inequality, as the coefficients $\alpha_B$ and $\alpha_N$ are chosen independently, akin to a sequential lifting as opposed to a simultaneous lifting. However, we did not exploit these degrees of freedom previously, as we focus solely on maximizing cut violation with respect to the chosen normalization (9).

We only proved above that under certain condition, it is possible to lift a valid inequality. We did not provide a way to compute the lifting coefficient explicitly ($\mu$ in Proposition 5 and Proposition 6) as we simply rely on Algorithm 1 to find it. Nevertheless, it may be interesting to examine the lifting problem itself. That problem has been thoroughly studied [6,25,41], and we only present here a few properties that are useful in our context. For simplicity, we assume for the time being that we want to lift the cut coefficient of only one variable $x_k$, i.e. $N = \{k\}$. Let us consider a valid inequality $\bar{\alpha}_B \geq \bar{\alpha}_0$ for $F = P \cap \{x_k = x_k^*\}$. We want to find $\alpha_k$ such that $\alpha_k x_k + \bar{\alpha}_B x_B \geq \bar{\alpha}_0 + \alpha_k x_k^*$ for all $(x_k, x_B)$ in $P$. The condition is trivially satisfied when $x_k = x_k^*$. For $x_k \neq x_k^*$, we need $\alpha_k(x_k^* - x_k) \leq \bar{\alpha}_B x_B - \bar{\alpha}_0$ i.e.

$$\begin{cases} \alpha_k \leq \dfrac{\bar{\alpha}_B x_B - \bar{\alpha}_0}{x_k^* - x_k} & \forall x \in P : x_k < x_k^* \\ \alpha_k \geq \dfrac{\bar{\alpha}_B x_B - \bar{\alpha}_0}{x_k^* - x_k} & \forall x \in P : x_k > x_k^*. \end{cases} \tag{16}$$

For a given $k$, only one of the two types of conditions in (16) can occur, since $x_k^*$ is the value of a bound on $x_k$. Therefore, we can find $\alpha_k$ by solving an optimization problem whose feasible region is either $P \cap \{x_k < x_k^*\}$ or $P \cap \{x_k < x_k^*\}$. Unfortunately, we can not solve these problems in practice, be it only because their feasible regions may be open sets. However, if $x_k$ is a binary variable, then the problems reduce to

$$\alpha_k = \min \{\bar{\alpha}_B \boldsymbol{x}_B - \bar{\alpha}_0 : \boldsymbol{x} \in P \quad \text{and} \quad \boldsymbol{x}_k = 0\} \tag{17}$$

and

$$\alpha_k = -\min \{\bar{\alpha}_B \boldsymbol{x}_B - \bar{\alpha}_0 : \boldsymbol{x} \in P \quad \text{and} \quad \boldsymbol{x}_k = 1\}, \tag{18}$$

which are mixed-integer programming problems.

Therefore, for a binary variable, it is possible to compute a lifted cut coefficient by solving a single MIP instead of resorting to the row-generation Algorithm 1, which may need to solve many MIPs of the same size. In general for $|N| \geq 1$, we can compute sequentially a valid lifted coefficient $\alpha_k$ for all the binary variables, by solving for each $k$ the mixed-integer problem

$$\alpha_k = (1 - 2x_k^*) \min_{\boldsymbol{x} \in P} \left\{ \bar{\alpha}_B^T \boldsymbol{x}_B - \bar{\alpha}_0 \ : \ \boldsymbol{x}_{N\setminus\{k\}} = x^*_{N\setminus\{k\}}, \ \boldsymbol{x}_k = 1 - x_k^* \right\} \tag{19}$$

then setting $N := N\setminus\{k\}$ and $B := B \cup \{k\}$. Note that such a lifting is not unique. In particular, it is sequence-dependent, i.e. it depends on the order according to which it is applied to the variables.

For general integer and continuous variables, before Phase 2, we try to find lifting coefficients one by one. This is still done through row-generation, but in essence performing a sequential lifting instead of the simultaneous lifting that is Phase 2. By doing so, we again lose a degree of freedom that we were not exploiting previously. We do not have a formal argument for claiming that this sequential lifting would speed up computations. The intuition is that we will apply row-generation many times, but to solve simpler problems, and we can only verify it through experimentation. In practice, these procedures based on lifting further reduced the running time (in geometric mean) on our benchmark testset of 12 instances, by about 30 %.

Finally, we use a number of computational tricks in addition to the improvements mentioned above. One of them is of particular interest. We use the callback facilities of the branch-and-bound solver to interrupt the resolution of the slave MIP (14) prior to optimality, when given the opportunity. Indeed, before the termination of the branch-and-bound method, the global lower bound may already indicate that the candidate inequality is valid. Conversely, an incumbent integer solution, even nonoptimal, may be violated by the candidate inequality, effectively proving that it is not valid and providing an additional constraint for $Q(S, T)$. In practice, we stop the MIP solver after 1000 consecutive branch-and-bound nodes without progress in the MIP bound, when the incumbent provides a violated constraint of $Q(S, T)$. These computational tricks further halved the computing time (in geometric mean) on our 12-instances testset.

Overall, our tests show that the final algorithm is, on average, approximately 15 times faster than Algorithm 1, and more numerically stable, thus providing a better choice for tackling medium-sized benchmark instances.

## 4 Experimental setup

We aim at obtaining a reasonable evaluation of the percentage of gap closed by cuts from various relaxations. We work on the MIPLIB 3 problem instances [13], which are first preprocessed using the CPLEX 12.6 MIP solver (we performed only basic preprocessing, with default settings; there is no root node processing or cut generation). Note that the cuts we generate are rank-1 only with respect to these preprocessed instances, since for example, strengthened bounds could be considered as cutting planes. Therefore, for consistency, all the objective function gaps are also measured

with respect to the preprocessed instances. All runs are obtained on an Intel Xeon X5650 2.67 GHz, with the memory usage of CPLEX limited to 8 Gb per instance. The general layout of our experiment is summarized in Algorithm 3. For each instance, we start by optimizing over the LP relaxation and we compute an optimal LP tableau. From that tableau, we build the multi-row relaxations that we will use throughout the computation. We also compute a round of GMI cuts from that tableau, and add them to the formulation (we use our own GMI cut generator for that purpose). Note that these new constraints are not considered when forming the multi-row relaxations: all the multi-row models considered in our computations come directly from an LP tableau of the initial preprocessed formulation. We then enter the main loop. At each iteration we try to separate the current LP optimal point with each multi-row model, add the separating cuts to the formulation, and compute a new optimal point for the resulting LP.

We now describe how we construct the multi-row models. From a theoretical perspective, closures are a useful tool to evaluate the strength of a family of cutting planes. For a given family of cuts, the first closure of a MIP is the polyhedron obtained by adding, to the LP relaxation, all the cuts of the family that arise from models constructed from that LP relaxation. In practice, it means that each row of the underlying models can be a linear combination of any set of rows from any tableau of the LP relaxation. The $i$th closure is obtained by adding cuts from models constructed upon the $(i-1)$th closure. A cut is said to be rank-$i$ with respect to its family if it is valid for the $i$th closure, but not the $(i-1)$th.

```
1   Input: a mixed-integer problem P, its linear relaxation P_LP
2
3   Optimize over P_LP.
4   Generate a round of GMIs from the optimal tableau.
5   Build multi-row models from the optimal tableau.
6   Add the GMIs to P_LP
7
8   do
9           Optimize over P_LP. Let x* be the optimal solution.
10
11          for each multi-row model,
12                  generate a cut, trying to separate x*.
13          end for
14          Add the separating cuts to P_LP.
15  while at least one cut was added.
```

Algorithm 3: Main cut-generation loop

Because our separator is still too slow for that purpose, we can not claim to be even approaching the computation of a first closure for the various types of cuts we test. We thus have to determine a way to select the relaxations from which to generate cuts. Our first attempt is to focus on one optimal tableau, and build models from all combinations of rows that can be directly read from that tableau. This is a natural choice since in practice, most cutting planes are generated from simplex tableaux, and intersection cuts from an *optimal* tableau are guaranteed to separate the corresponding optimal vertex. This yields $\binom{m}{k}$ $k$-row models for a MIP with $m$ rows.

Remark that all the relaxations that we consider are strengthenings of the intersection cut model $P_I$. We assume that in a practical context, the corresponding cuts would be generated with variants of the intersection cut method. For that reason, we will consider only models for which all basic variables are integer-constrained and at least one component of the right-hand side $f$ is fractional. Recall that we build all the models directly from data of the initial (preprocessed) problem, hence we generate only rank-1 cuts.

Computational issues can arise when computing cuts with our separator. Many of the steps in Algorithm 1 can fail for various reasons. We impose iteration limits on every MIP computation. Numerical issues happen too. For example, the solver can claim that the slave MIP (14) is unbounded while its LP relaxation is not (indeed those two problems are preprocessed separately to avoid other issues), or encounter variables whose value is too large to be represented internally. The solver sometimes claims the master LP (10) to be unbounded, which is not possible by construction, or infeasible while lifting, which should not happen either. Algorithm 1 is itself subject to an iteration limit, and we enforce a global time limit of 4 h per instance. Memory is also available in limited amounts, and the code may run out of space.

Finally, we discard valid inequalities that have bad numerical properties or do not separate the point $x^*$: if the violation at $x^*$ is less than $10^{-4}$, if the dynamism of the constraint (i.e. the quotient of the largest and the smallest nonzero cut coefficient) is larger than $10^{10}$, or if its efficacy (i.e. the violation divided by the norm) is less than $10^{-4}$. Both cuts with small violation and cuts with small efficacy are more likely to cause degeneracy-related numerical issues around the current LP optimum. Specifically, cuts with small efficacy are known to typically cut off a tiny volume of the LP relaxation, in which case the corresponding exact-arithmetic valid inequality may not be violated at all.

In the above failure cases, the computation stops for the concerned model but can continue with other models. In practice, issues arise for a number of models in most of the instances. But as long as, at each iteration of Algorithm 3, one or more models generate separating inequalities, the process can continue. For example, over the experiment run to generate Table 1, we successfully generated a total of 40,358 cuts and encountered 1,852 failures over 93 instances. For all of these, we still have a result, and for 14 of them, we even have a proof that, at the last iteration, no more gap could be closed with cuts from the selected models. This is made possible by the fact that Phase-1 is enough to show the latter, while actually computing a cut requires completing Phase-2, which is much more difficult computationally.

Note that the separation procedure outlined in Algorithm 1, along with its two-phase counterpart, is, in principle, exact, meaning that it will find a violated inequality if and only if one exists. However, implementations of Algorithm 1 may not be exact because of finite precision arithmetic, tolerances, and resource limits. In particular, our implementation is *not* exact because we are relying on a finite precision solver such as CPLEX and because of the numerical safeguards just described, although it can be considered *reasonably* accurate for well behaved instances. Since the indicator that we use to measure the strength of the models is the gap closed by the cut they yield, we do not need to guarantee that our cuts are facet-defining in the computations that generate the tables. As we do not constrain our separator to generate only facet-

**Table 1** All full two-row models and heuristically selected subset

| | GMIs only | | All pairs of rows | | | | Heuristic selection | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| name | cuts | %gc | time | cuts | %gc | ex. | time | cuts | %gc | ex. |
| 10teams | 101 | 57.1 | [4h] | 4 | 57.1 | | [4h] | 4 | 57.1 | |
| air03 | 3 | 100 | 0.5 | 31 | 100 | ✓ | 0.6 | 31 | 100 | ✓ |
| air04 | 100 | 7.9 | [4h] | 13 | 7.9 | | [4h] | 13 | 7.9 | |
| air05 | 101 | 4.6 | [4h] | 8 | 4.6 | | [4h] | 8 | 4.6 | |
| arki001 | 7 | 0.0 | [4h] | 7 | 0.0 | | [4h] | 7 | 0.0 | |
| bell3a | 14 | 39.0 | 118.6 | 121 | 59.0 | ✓ | 11.0 | 29 | 55.7 | ✓ |
| bell5 | 17 | 14.5 | 567.6 | 45 | 91.2 | | 264.1 | 26 | 19.3 | ✓ |
| blend2 | 5 | 16.0 | 369.6 | 40 | 28.2 | ✓ | 19.6 | 11 | 20.0 | ✓ |
| cap6000 | 2 | 39.9 | [4h] | 2 | 39.9 | | [4h] | 2 | 39.9 | |
| danoint | 31 | 0.3 | [4h] | 319 | 1.7 | | [4h] | 319 | 1.7 | |
| dcmulti | 36 | 47.8 | [4h] | 214 | 73.3 | | 3520.5 | 175 | 59.8 | ✓ |
| egout | 8 | 31.9 | 2.0 | 31 | 100.0 | ✓ | 0.8 | 21 | 98.4 | ✓ |
| fast0507 | 100 | 1.7 | [4h] | 5 | 1.7 | | [4h] | 5 | 1.7 | |
| fiber | 22 | 69.2 | 7623.8 | 581 | 87.0 | ✓ | 703.8 | 281 | 81.3 | ✓ |
| fixnet6 | 11 | 22.3 | [4h] | 155 | 51.0 | | 2315.3 | 69 | 36.5 | ✓ |
| flugpl | 7 | 10.8 | 2.7 | 6 | 51.2 | ✓ | 1.5 | 6 | 50.8 | ✓ |
| gen | 6 | 1.3 | [4h] | 7 | 39.2 | | [4h] | 7 | 39.2 | |
| gesa2 | 40 | 27.6 | [4h] | 220 | 73.8 | | 1336.9 | 196 | 70.8 | |
| gesa2_o | 70 | 30.7 | [4h] | 465 | 81.0 | | 2368.3 | 272 | 48.0 | |
| gesa3 | 37 | 20.5 | [4h] | 431 | 60.5 | | 964.9 | 278 | 56.1 | |
| gesa3_o | 64 | 50.5 | [4h] | 394 | 80.6 | | 13182.1 | 351 | 77.8 | |
| gt2 | 11 | 47.2 | [4h] | 66 | 58.6 | | 13837.6 | 78 | 58.6 | |
| harp2 | 22 | 22.8 | [4h] | 20 | 22.8 | | [4h] | 20 | 22.8 | |
| khb05250 | 19 | 73.2 | 9437.8 | 455 | 99.2 | | 2013.9 | 218 | 91.4 | ✓ |
| l152lav | 7 | 2.0 | [4h] | 4 | 3.1 | | [4h] | 4 | 3.1 | |
| lseu | 5 | 20.5 | 9175.0 | 57 | 70.2 | ✓ | 8747.3 | 57 | 70.2 | ✓ |
| markshare1 | 6 | 0.0 | 5968.7 | 1 | 0.0 | | 1212.1 | 1 | 0.0 | |
| markshare2 | 7 | 0.0 | 11193.7 | 1 | 0.0 | | 2506.7 | 1 | 0.0 | |
| mas74 | 11 | 6.7 | [4h] | 10 | 6.9 | | [4h] | 10 | 6.9 | |
| mas76 | 10 | 6.4 | [4h] | 6 | 6.4 | | [4h] | 6 | 6.4 | |
| misc03 | 4 | 8.6 | 5469.9 | 78 | 9.8 | | 38.0 | 9 | 8.6 | ✓ |
| misc06 | 16 | 28.5 | 3439.3 | 175 | 75.4 | | 54.6 | 27 | 48.8 | ✓ |
| misc07 | 5 | 0.7 | 4027.3 | 62 | 0.9 | ✓ | 2.6 | 13 | 0.7 | ✓ |
| mitre | 101 | 50.7 | [4h] | 828 | 67.3 | | [4h] | 828 | 67.3 | |
| mkc | 31 | 1.4 | [4h] | 44 | 1.4 | | [4h] | 44 | 1.4 | |
| mod008 | 5 | 21.6 | [4h] | 4 | 21.6 | | [4h] | 4 | 21.6 | |
| mod010 | 5 | 100 | 0.3 | 15 | 100 | ✓ | 0.2 | 15 | 100 | ✓ |
| mod011 | 22 | 31.3 | [4h] | 19 | 31.4 | | [4h] | 19 | 31.4 | |
| modglob | 28 | 17.3 | 6035.8 | 97 | 58.0 | | 110.9 | 39 | 31.9 | |
| noswot | 14 | 0.0 | 151.3 | 42 | 0.0 | ✓ | 4.3 | 12 | 0.0 | ✓ |
| nw04 | 2 | 29.8 | [4h] | 4 | 29.9 | | [4h] | 4 | 29.9 | |
| p0033 | 4 | 34.4 | 6.9 | 59 | 100.0 | ✓ | 6.5 | 59 | 100.0 | ✓ |
| p0201 | 14 | 0.4 | [4h] | 75 | 17.8 | | 8114.0 | 33 | 15.8 | ✓ |
| p0282 | 23 | 3.2 | 4781.9 | 174 | 42.3 | | 2087.6 | 143 | 41.9 | ✓ |
| p0548 | 31 | 61.7 | 33.3 | 112 | 99.9 | ✓ | 16.7 | 112 | 99.9 | ✓ |
| p2756 | 81 | 51.7 | [4h] | 528 | 97.2 | | 4330.7 | 404 | 77.5 | ✓ |
| pk1 | 15 | 0.0 | [4h] | 1 | 0.0 | | 9706.5 | 1 | 0.0 | |
| pp08a | 53 | 51.4 | 171.0 | 97 | 80.4 | ✓ | 13.3 | 85 | 76.2 | ✓ |
| pp08acuts | 41 | 31.5 | [4h] | 57 | 47.9 | | 3234.4 | 46 | 46.9 | |
| qiu | 23 | 1.7 | [4h] | 18 | 1.8 | | [4h] | 18 | 1.8 | |
| qnet1 | 21 | 11.8 | [4h] | 7 | 12.0 | | [4h] | 7 | 12.0 | |
| qnet1_o | 10 | 23.3 | [4h] | 39 | 25.1 | | [4h] | 41 | 25.1 | |
| rentacar | 13 | 5.0 | [4h] | 22 | 5.7 | | [4h] | 24 | 5.7 | |
| rgn | 12 | 5.0 | 226.7 | 33 | 42.0 | | 26.7 | 32 | 25.1 | ✓ |
| rout | 29 | 3.6 | [4h] | 13 | 3.8 | | [4h] | 13 | 3.8 | |
| set1ch | 121 | 28.2 | 5432.9 | 166 | 62.8 | ✓ | 37.4 | 153 | 61.8 | ✓ |
| seymour | 50 | 6.2 | [4h] | 24 | 10.9 | | [4h] | 24 | 11.0 | |
| stein27 | 21 | 0.0 | 624.8 | 7 | 0.0 | ✓ | 10.6 | 4 | 0.0 | ✓ |
| stein45 | 16 | 0.0 | [4h] | 7 | 0.0 | | 93.8 | 3 | 0.0 | ✓ |
| swath | 13 | 6.4 | [4h] | 25 | 31.4 | | [4h] | 25 | 31.4 | |
| vpm1 | 15 | 6.2 | 179.2 | 41 | 17.1 | ✓ | 4.0 | 25 | 15.9 | ✓ |
| vpm2 | 21 | 7.3 | 515.5 | 63 | 42.1 | | 58.5 | 47 | 33.6 | ✓ |
| average | 28.1 | **22.6** | 9580.6 | 107.3 | **40.2** | $\frac{16}{62}$ | 6880.4 | 77.7 | **35.7** | $\frac{28}{62}$ |
| geometric | 17.0 | 0.0 | 2994.7 | 33.2 | 0.0 | - | 835.2 | 24.5 | 0.0 | - |

defining inequalities, we can disable the LP post-processing necessary to cope with nonbasic free variables, and enable the lifting of $0 \geq 1$ inequalities when the Phase-1 slave MIP is infeasible.

The section "All pairs of rows" of Table 1 presents the results obtained by running Algorithm 3 with the selection of 2-row models described above. The column *cuts* indicates the number of cuts that are tight at the last iteration of Algorithm 3. The column *%gc* indicates the percentage of gap closed by all the cuts generated. A "✓" in the column *ex.* indicates that the separation is theoretically exact with respect to the selection of models (i.e. we can prove that no more gap could be closed with cuts from that selection of models). Again, this is only *theoretically* exact, because it does not take into account the numerical inaccuracies incurred by the floating-point representation of numbers. The section "GMIs only" is provided for comparison, and shows the number of GMIs that we computed from the initial optimal tableau and the percentage of gap that they close. We run our test on 62 of the 65 MIPLIB 3 instances: `dsbmip` and `enigma` have no integrality gap and no optimal solution is known for `dano3mip`.

We observe that, on average, the amount of gap closed by the two-row cuts (40.2 %) is significantly higher than the gap closed by the GMIs (22.6 %). This is not surprising, since we consider full two-row models here, i.e. we do not drop any integrality or bound constraint on the variables. On the other hand, our main observation in Table 1 is that the running time is not satisfactory. We hit the global time limit (4 h) in most instances, and we have exact separation for only 16 of them.

These excessive computing times can be blamed in part on the sheer number of two-row models we select when taking all suitable pairs of rows from an optimal tableau. Moreover, this problem is bound to get worse when considering relaxations with more than two rows. In order to alleviate the computational burden caused by this policy, we now look for a more restricted selection of models that could nevertheless provide similar results.

We try to construct models whose rows have similar supports. This is motivated by the extreme case where the rows have disjoint supports. In that case indeed, valid inequalities for the model are just conic combinations of valid inequalities for the individual rows. In order to have rows with similar supports, we design the heuristic row selector summarized in Algorithm 4. It first builds, for each row of the tableau, a cluster of other rows that have a similar support. Then, it loops through the clusters constructing one multi-row model with the rows in each one. Our intent with this heuristic is to favor models from rows with similar supports, while covering all rows of the tableau. Note that the constraints mentioned earlier for each row to have an integer-constrained basic variable, and for each model to have a fractional-valued basic variable, are incorporated into the heuristic.

At line 4 of Algorithm 4, we need to define a measure for the similarity of the supports of two rows. Let $S_c$ be the number of columns in which both rows have nonzero coefficients, and $S_d$ the number of columns in which exactly one row has a nonzero coefficients. We choose to maximize a score given by $S_c - S_d$, i.e. we score positively columns where both rows have nonzero coefficients and penalize columns where one row has a zero and the other not. We also tried a score given by $S_c/(S_c+S_d)$, with slightly less appealing results [40].

```
1   Input: A m-row simplex tableau and k, the target number of rows per model
2
3   for each row i of the tableau,
4           build a cluster of 5k − 1 rows having a support most similar to row i
5   end for
6
7   i := 1.
8   while models < MODELS_MAX,
9           Find a subset of k − 1 rows in the cluster i such that the k-row model
10          built from that subset and i does not already form a selected model.
11
12          if such a subset exist,
13                  build a k-row model from these rows and row i
14          end if
15
16          i := (i + 1)  mod m.
17  end
```

Algorithm 4: Heuristic selection of multi-row models

The section "Heuristic selection" of Table 1 shows results obtained with two-row models constructed according to Algorithm 4. Compared to when reading all two-row models from an optimal tableau, the average percentage of gap closed drops from 40.2 to 35.7 %, but the running times are divided by a factor 3 in geometric mean, and the number of instances with exact separation almost doubles. Moreover, the percentage of gap closed in the second column is within 1 % of result in the first column for 39 of the 62 instances (63 %), and within 2 % for 45 instances (73 %). In Table 1 and in all later tables, MODELS_MAX is set to the number $m$ of rows in the preprocessed formulation of the problem. Note that each row corresponds to an instance for which every test presented succeeded. As a consequence, the testsets may vary from table to table. We will detail in the later tables which instances are missing if any.

## 5 Gomory mixed-integer, one-row and two-row cuts

Armed with a reasonable heuristic model selector, we can now compare the strength of various relaxations. A first question that can be tackled regards the strength of one-row models. We know that, given a one-row model, the GMI formula provides a facet-defining inequality for the integer hull. But how much tighter would a formulation become when also considering the other facet-defining inequalities of that one-row model? This question is studied in depth by Fukasawa and Goycoolea, who develop an exact separator for mixed-integer knapsack cuts [32]. While more generic and hence slower, our separator can be exploited for similar purposes.

Table 2 shows our results for that experiment. As before, our code starts by generating a round of Gomory mixed-integer cuts from an optimal simplex tableau. The "GMIs" columns show the number of such cuts generated, and the resulting amount of gap they close. The "One-row ($P_{\text{full}}$)" columns shows the effect of separating over the corresponding one-row models, possibly generating cuts that are not GMIs. Anticipating on the following sections, the last group of columns shows the amount of gap closed by a comparable number of (fully-strengthened) two-row models. Despite the

**Table 2** GMIs, one-row and two-row cuts

| Name | GMIs | | One-row ($P_{\text{full}}$) | | | Two-row ($P_{\text{full}}$) | | |
|---|---|---|---|---|---|---|---|---|
| | Cuts | %gc | Cuts | %gc | ex. | Cuts | %gc | ex. |
| 10teams | 101 | 57.1 | 4 | 57.1 | | 4 | 57.1 | |
| air03 | 3 | 100 | 31 | 100 | ✓ | 31 | 100 | ✓ |
| air04 | 100 | 7.9 | 13 | 7.9 | | 13 | 7.9 | |
| air05 | 101 | 4.6 | 8 | 4.6 | | 8 | 4.6 | |
| arki001 | 7 | 0.0 | 13 | 0.0 | | 7 | 0.0 | |
| bell3a | 14 | 39.0 | 8 | 39.4 | ✓ | 29 | 55.7 | ✓ |
| bell5 | 17 | 14.5 | 24 | 18.7 | | 26 | 19.3 | ✓ |
| blend2 | 5 | 16.0 | 3 | 16.7 | ✓ | 11 | 20.0 | ✓ |
| cap6000 | 2 | 39.9 | 2 | 39.9 | | 2 | 39.9 | |
| danoint | 31 | 0.3 | 5 | 0.3 | | 319 | 1.7 | |
| dcmulti | 36 | 47.8 | 43 | 50.9 | ✓ | 175 | 59.8 | ✓ |
| egout | 8 | 31.9 | 9 | 36.4 | ✓ | 21 | 98.4 | ✓ |
| fast0507 | 100 | 1.7 | 5 | 1.7 | | 5 | 1.7 | |
| fiber | 22 | 69.2 | 72 | 73.2 | ✓ | 281 | 81.3 | ✓ |
| fixnet6 | 11 | 22.3 | 27 | 27.0 | ✓ | 69 | 36.5 | ✓ |
| flugpl | 7 | 10.8 | 4 | 10.8 | ✓ | 6 | 50.8 | ✓ |
| gen | 6 | 1.3 | 6 | 1.3 | ✓ | 7 | 39.2 | |
| gesa2 | 40 | 27.6 | 53 | 29.1 | | 196 | 70.8 | |
| gesa2_o | 70 | 30.7 | 95 | 33.1 | ✓ | 272 | 48.0 | |
| gesa3 | 37 | 20.5 | 115 | 21.9 | ✓ | 278 | 56.1 | |
| gesa3_o | 64 | 50.5 | 124 | 50.7 | | 351 | 77.8 | |
| gt2 | 11 | 47.2 | 27 | 56.8 | ✓ | 78 | 58.6 | |
| harp2 | 22 | 22.8 | 20 | 22.8 | | 20 | 22.8 | |
| khb05250 | 19 | 73.2 | 150 | 91.4 | | 218 | 91.4 | ✓ |
| l152lav | 7 | 2.0 | 12 | 6.1 | | 4 | 3.1 | |
| lseu | 5 | 20.5 | 47 | 23.4 | ✓ | 57 | 70.2 | ✓ |
| markshare1 | 6 | 0.0 | 1 | 0.0 | | 1 | 0.0 | |
| markshare2 | 7 | 0.0 | 1 | 0.0 | | 1 | 0.0 | |
| mas74 | 11 | 6.7 | 27 | 9.1 | ✓ | 10 | 6.9 | |
| mas76 | 10 | 6.4 | 22 | 9.1 | | 6 | 6.4 | |
| misc03 | 4 | 8.6 | 7 | 8.6 | ✓ | 9 | 8.6 | ✓ |
| misc06 | 16 | 28.5 | 13 | 28.5 | ✓ | 27 | 48.8 | ✓ |
| misc07 | 5 | 0.7 | 9 | 0.7 | ✓ | 13 | 0.7 | ✓ |
| mitre | 101 | 50.7 | 575 | 76.2 | | 828 | 67.3 | |
| mkc | 31 | 1.4 | 46 | 1.8 | | 44 | 1.4 | |
| mod008 | 5 | 21.6 | 15 | 22.6 | ✓ | 4 | 21.6 | |
| mod010 | 5 | 100 | 15 | 100 | ✓ | 15 | 100 | ✓ |
| mod011 | 22 | 31.3 | 22 | 31.3 | | 19 | 31.4 | |
| modglob | 28 | 17.3 | 18 | 18.1 | ✓ | 39 | 31.9 | |

**Table 2** continued

| Name | GMIs | | One-row ($P_{\text{full}}$) | | | Two-row ($P_{\text{full}}$) | | |
|---|---|---|---|---|---|---|---|---|
| | Cuts | %gc | Cuts | %gc | ex. | Cuts | %gc | ex. |
| noswot | 14 | 0.0 | 9 | 0.0 | ✓ | 12 | 0.0 | ✓ |
| nw04 | 2 | 29.8 | 6 | 31.1 | | 4 | 29.9 | |
| p0033 | 4 | 34.4 | 13 | 60.0 | ✓ | 59 | 100.0 | ✓ |
| p0201 | 14 | 0.4 | 18 | 0.4 | ✓ | 33 | 15.8 | ✓ |
| p0282 | 23 | 3.2 | 66 | 14.7 | ✓ | 143 | 41.9 | ✓ |
| p0548 | 31 | 61.7 | 81 | 97.1 | ✓ | 112 | 99.9 | ✓ |
| p2756 | 81 | 51.7 | 129 | 70.9 | ✓ | 404 | 77.5 | ✓ |
| pk1 | 15 | 0.0 | 2 | 0.0 | | 1 | 0.0 | |
| pp08a | 53 | 51.4 | 46 | 51.4 | ✓ | 85 | 76.2 | ✓ |
| pp08acuts | 41 | 31.5 | 32 | 31.7 | | 46 | 46.9 | |
| qiu | 23 | 1.7 | 24 | 1.7 | | 18 | 1.8 | |
| qnet1 | 21 | 11.8 | 14 | 12.0 | | 7 | 12.0 | |
| qnet1_o | 10 | 23.3 | 37 | 27.3 | | 41 | 25.1 | |
| rentacar | 13 | 5.0 | 11 | 5.0 | | 24 | 5.7 | |
| rgn | 12 | 5.0 | 12 | 10.0 | ✓ | 32 | 25.1 | ✓ |
| rout | 29 | 3.6 | 37 | 4.6 | | 13 | 3.8 | |
| set1ch | 121 | 28.2 | 120 | 28.2 | ✓ | 153 | 61.8 | ✓ |
| seymour | 50 | 6.2 | 14 | 7.0 | ✓ | 24 | 11.0 | |
| stein27 | 21 | 0.0 | 1 | 0.0 | ✓ | 4 | 0.0 | ✓ |
| stein45 | 16 | 0.0 | 2 | 0.0 | ✓ | 3 | 0.0 | ✓ |
| swath | 13 | 6.4 | 23 | 11.4 | | 25 | 31.4 | |
| vpm1 | 15 | 6.2 | 10 | 6.5 | ✓ | 25 | 15.9 | ✓ |
| vpm2 | 21 | 7.3 | 20 | 8.1 | ✓ | 47 | 33.6 | ✓ |
| Average | 28.1 | **22.6** | 39.0 | **25.9** | $\frac{34}{62}$ | 77.7 | **35.7** | $\frac{28}{62}$ |

different settings (we work on preprocessed instances, and do not use exact arithmetic), our conclusions are very similar to the ones in [32]. In terms of gap closed, generic one-row cuts achieve barely more (25.9 % in average) than what is already obtained with GMIs (22.6 %), despite the fact that we generate more one-row cuts on average, and add them on top of the GMIs.

## 6 Strengthened intersection cuts

In this section we focus on two-row models, and test in more detail the classes of models presented in Sect. 1. We can now compare the strength of various relaxations for a fixed set of two-row models.

Recall that $P_I$ is the intersection cut model. In particular, its two-row version is

$$P_I = \{(x, s) \in \mathbb{Z}^2 \times \mathbb{R}^n_+ : x = f + Rs\}.$$

**Table 3** Strengthenings of $P_I$: ✓: keep, T: keep binding

| | Basic | | Nonbasic | |
|---|---|---|---|---|
| | $\in \mathbb{Z}$ | bounds | $\in \mathbb{Z}$ | bounds |
| $P_I$ | ✓ | | | T |
| $S$-free | ✓ | ✓ | | T |
| Lifting | ✓ | | ✓ | T |
| $P_{IU}$ | ✓ | | | ✓ |
| $P_{\text{full}}$ | ✓ | ✓ | ✓ | ✓ |

We can strengthen the intersection cut model by reintroducing bounds on the basic variables $x$, yielding

$$P_{S\text{-free}} = \{(x, s) \in S \times \mathbb{R}^n_+ : x = f + Rs\}.$$

We can also exploit the integrality of the nonbasic variables by solving the so-called lifting problem, and obtain valid inequalities for

$$P_{\text{lifting}} = \{(x, s) \in \mathbb{Z}^2 \times \mathbb{R}^n_+ : x = f + Rs,\ s_j \in \mathbb{Z} \text{ for all } j \in J\}.$$

Finally, taking advantage of upper bounds on the nonbasic variables, we can generate inequalities that are valid for

$$P_{IU} = \{(x, s) \in \mathbb{Z}^2 \times \mathbb{R}^n_+ : x = f + Rs, s \leq U\}.$$

If we denote by $P_{\text{full}}$ a two-row model where we keep all integrality and bound constraints on the variables, $P_I$ consists in dropping from $P_{\text{full}}$ bounds on the two basic variables, integrality on nonbasic variables, and any nonbinding bound on the nonbasic variables. This is shown in Table 3, along with the constraints that are reintroduced in $P_{S\text{-free}}$, $P_{\text{lifting}}$ and $P_{IU}$.

Table 4 shows the gap closed with a unique selection of two-row models, but different relaxations, namely $P_I$, $P_{S\text{-free}}$, $P_{\text{lifting}}$, $P_{IU}$ and $P_{\text{full}}$. Note that for $P_I$, we use the exact separator described in [36], which also served as a crosscheck for the generic separator presented here. If, for an instance, the amount of gap closed by one type of relaxation is lower than the gap closed by a weaker type of relaxation (due to our separator not being able to perform exact separation on that instance), then we display the number corresponding to the weaker relaxation. This situation is denoted by a * in the table. The instance `arki001` is missing from this table, because we encounter numerical issues in some of the computations.

It appears in the table that individual strengthenings do not bring a lot of improvement over intersection cuts. The gap closed goes from 23.0 to 29.1 % by adding intersection cuts (i.e. cuts from $P_I$) over GMIs. But cuts from $P_{S\text{-free}}$, $P_{\text{lifting}}$ and $P_{IU}$ barely make the percentage of gap closed exceed 31 %. The gap closed jumps to 37.5 % only when considering cuts from $P_{\text{full}}$.

Unfortunately, our results regarding $P_{\text{lifting}}$ are of limited significance. It is indeed extremely hard for Algorithm 1 to separate from $P_{\text{lifting}}$, as indicated by the very low

**Table 4** Strengthenings of $P_I$

| Name | GMIs | $P_I$ | | $P_{S-\text{free}}$ | | $P_{\text{lifting}}$ | | $P_{IU}$ | | $P_{\text{full}}$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | %gc | %gc | ex. | %gc | ex. | %gc | ex. | %gc | ex. | %gc | ex. |
| 10teams | 57.1 | 57.1 | ✓ | 57.1 | ✓ | 57.1 | | 57.1 | ✓ | 57.1 | |
| air03 | 100 | 100 | ✓ | 100 | ✓ | 100 | ✓ | 100 | ✓ | 100 | ✓ |
| air04 | 7.9 | 7.9 | | 8.2 | | 7.9 | | 7.9 | | *8.2 | |
| air05 | 4.6 | 4.8 | ✓ | 5.1 | | *4.8 | | 4.8 | ✓ | *5.1 | |
| bell3a | 39.0 | 52.5 | ✓ | 54.9 | ✓ | *52.5 | | 52.9 | ✓ | 55.7 | ✓ |
| bell5 | 14.5 | 14.9 | ✓ | 14.9 | ✓ | 15.1 | ✓ | 19.2 | ✓ | 19.3 | ✓ |
| blend2 | 16.0 | 16.9 | ✓ | 17.0 | | 17.2 | ✓ | 19.8 | ✓ | 20.0 | ✓ |
| cap6000 | 39.9 | 40.1 | ✓ | 40.1 | ✓ | *40.1 | | 40.8 | | *40.8 | |
| danoint | 0.3 | 1.7 | ✓ | 1.7 | ✓ | 1.7 | | 1.7 | | 1.7 | |
| dcmulti | 47.8 | 50.3 | ✓ | 56.6 | ✓ | 51.9 | | 53.4 | | 59.8 | ✓ |
| egout | 31.9 | 94.2 | ✓ | 97.9 | ✓ | 94.2 | ✓ | 98.4 | ✓ | 98.4 | ✓ |
| fast0507 | 1.7 | 1.7 | | 1.7 | | 1.7 | | 1.7 | | 1.7 | |
| fiber | 69.2 | 70.8 | ✓ | 70.8 | ✓ | *70.8 | | 71.5 | ✓ | 81.3 | ✓ |
| fixnet6 | 22.3 | 30.6 | ✓ | 31.2 | ✓ | *30.6 | | 36.5 | ✓ | 36.5 | ✓ |
| flugpl | 10.8 | 13.3 | ✓ | 13.2 | | 43.0 | ✓ | 13.3 | | 50.8 | ✓ |
| gen | 1.3 | 9.5 | | 24.6 | | *9.5 | | 9.9 | ✓ | 39.2 | |
| gesa2 | 27.6 | 56.5 | | 56.9 | | *56.5 | | 57.4 | | 70.8 | |
| gesa2_o | 30.7 | 42.5 | ✓ | 42.4 | | 47.1 | | 42.8 | | 48.0 | |
| gesa3 | 20.5 | 47.1 | ✓ | 53.5 | | *47.1 | | 47.7 | ✓ | 56.1 | |
| gesa3_o | 50.5 | 69.8 | ✓ | 72.8 | | *69.8 | | 70.2 | ✓ | 77.8 | |
| gt2 | 47.2 | 47.2 | ✓ | 51.0 | ✓ | 51.0 | | 51.0 | ✓ | 58.6 | |
| harp2 | 22.8 | 25.2 | | 26.7 | ✓ | *25.2 | | *25.2 | | *26.7 | |
| khb05250 | 73.2 | 73.2 | ✓ | 73.2 | ✓ | 73.2 | ✓ | 91.4 | ✓ | 91.4 | ✓ |
| l152lav | 2.0 | 2.0 | | 9.7 | ✓ | 2.0 | | 2.7 | | *9.7 | |
| lseu | 20.5 | 21.7 | ✓ | 22.4 | ✓ | *21.7 | | 22.3 | ✓ | 70.2 | ✓ |
| markshare1 | 0.0 | 0.0 | ✓ | 0.0 | ✓ | 0.0 | | 0.0 | ✓ | 0.0 | |
| markshare2 | 0.0 | 0.0 | ✓ | 0.0 | | 0.0 | | 0.0 | | 0.0 | |
| mas74 | 6.7 | 6.7 | ✓ | 6.7 | ✓ | 6.7 | | 6.7 | | 6.9 | |
| mas76 | 6.4 | 6.4 | ✓ | 6.4 | ✓ | 6.4 | | 6.4 | ✓ | 6.4 | |
| misc03 | 8.6 | 8.6 | ✓ | 8.6 | ✓ | 8.6 | ✓ | 8.6 | ✓ | 8.6 | ✓ |
| misc06 | 28.5 | 48.3 | | 48.8 | ✓ | 48.3 | | 48.3 | | 48.8 | ✓ |
| misc07 | 0.7 | 0.7 | ✓ | 0.7 | ✓ | 0.7 | ✓ | 0.7 | ✓ | 0.7 | ✓ |
| mitre | 50.7 | 81.7 | | 81.7 | | *81.7 | | *81.7 | | *81.7 | |
| mkc | 1.4 | 1.4 | | 20.5 | | 1.6 | | 10.8 | | *20.5 | |
| mod008 | 21.6 | 21.6 | ✓ | 21.6 | ✓ | 21.6 | | 21.6 | ✓ | 21.6 | |
| mod010 | 100 | 100 | ✓ | 100 | ✓ | 100 | ✓ | 100 | ✓ | 100 | ✓ |
| mod011 | 31.3 | 36.3 | ✓ | *36.3 | | *36.3 | | *36.3 | | *36.3 | |
| modglob | 17.3 | 26.2 | ✓ | 26.6 | | 26.9 | | *26.2 | | 31.9 | |
| noswot | 0.0 | 0.0 | ✓ | 0.0 | ✓ | 0.0 | | 0.0 | ✓ | 0.0 | ✓ |

**Table 4** continued

| Name | GMIs | $P_I$ | | $P_{S-\text{free}}$ | | $P_{\text{lifting}}$ | | $P_{IU}$ | | $P_{\text{full}}$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | %gc | %gc | ex. | %gc | ex. | %gc | ex. | %gc | ex. | %gc | ex. |
| nw04 | 29.8 | 29.8 | ✓ | 29.8 | | 29.8 | ✓ | 29.8 | | 29.9 | |
| p0033 | 34.4 | 34.8 | ✓ | 44.3 | ✓ | 45.8 | | 38.0 | ✓ | 100.0 | ✓ |
| p0201 | 0.4 | 6.5 | ✓ | 8.6 | ✓ | *6.5 | | 6.5 | ✓ | 15.8 | ✓ |
| p0282 | 3.2 | 4.7 | | 14.0 | | *4.7 | | 15.0 | | 41.9 | ✓ |
| p0548 | 61.7 | 61.7 | ✓ | 62.1 | ✓ | 69.3 | | 87.9 | | 99.9 | ✓ |
| p2756 | 51.7 | 53.0 | ✓ | 61.9 | | 55.5 | ✓ | 67.0 | ✓ | 77.5 | ✓ |
| pk1 | 0.0 | 0.0 | ✓ | 0.0 | ✓ | 0.0 | | 0.0 | ✓ | 0.0 | |
| pp08a | 51.4 | 76.2 | ✓ | 76.2 | | *76.2 | | *76.2 | | 76.2 | ✓ |
| pp08acuts | 31.5 | 41.7 | ✓ | 44.2 | | *41.7 | | *41.7 | | 46.9 | |
| qiu | 1.7 | 1.9 | | *1.9 | | *1.9 | | 1.9 | | *1.9 | |
| qnet1 | 11.8 | 16.3 | | 22.7 | ✓ | *16.3 | | *16.3 | | *22.7 | |
| qnet1_o | 23.3 | 30.2 | ✓ | 36.5 | ✓ | *30.2 | | *30.2 | | *36.5 | |
| rentacar | 5.0 | 5.5 | | *5.5 | | *5.5 | | 5.5 | | 5.7 | |
| rgn | 5.0 | 5.0 | ✓ | 5.0 | | 5.0 | | 5.0 | ✓ | 25.1 | ✓ |
| rout | 3.6 | 4.1 | | 6.5 | | *4.1 | | *4.1 | | *6.5 | |
| set1ch | 28.2 | 61.6 | ✓ | 61.7 | ✓ | 61.7 | ✓ | 61.7 | ✓ | 61.8 | ✓ |
| seymour | 6.2 | 7.1 | ✓ | 9.4 | | 7.1 | | 7.1 | ✓ | 11.0 | |
| stein27 | 0.0 | 0.0 | ✓ | 0.0 | ✓ | 0.0 | | 0.0 | ✓ | 0.0 | ✓ |
| stein45 | 0.0 | 0.0 | ✓ | 0.0 | ✓ | 0.0 | ✓ | 0.0 | ✓ | 0.0 | ✓ |
| swath | 6.4 | 6.7 | ✓ | 6.7 | | 31.4 | | 6.7 | | 31.4 | |
| vpm1 | 6.2 | 14.6 | ✓ | 14.9 | ✓ | 14.6 | | 15.3 | ✓ | 15.9 | ✓ |
| vpm2 | 7.3 | 25.0 | ✓ | 28.4 | ✓ | 25.0 | ✓ | 30.3 | | 33.6 | ✓ |
| Average | **23.0** | **29.1** | $\frac{47}{61}$ | **31.2** | $\frac{35}{61}$ | **30.5** | $\frac{14}{61}$ | **31.0** | $\frac{32}{61}$ | **37.5** | $\frac{28}{61}$ |

number of instances with exact separation for these models. Intuitively, this can be explained by the fact that to build $P_{\text{lifting}}$ from $P_{\text{full}}$, we drop a number of bound constraints on the variables, but keep all integrality constraints (see Table 3). And while all the difficulty in solving MIPs comes from the integrality of the variables, the bound constraints are very useful for the branch-and-bound method to reduce the enumeration space. As a consequence, the average percentage of gap closed by our separator is often even lower for the $P_{\text{lifting}}$ models than for the weaker $P_I$ models (as denoted by a * in Table 4). We can somewhat mitigate the issue by looking only at instances for which the separation is exact for all of the tests, at the cost of reducing the testset. We have exact results with $P_{\text{lifting}}$ for only 14 instances, but if they are any indication, they tend to confirm that $P_{\text{lifting}}$ does not clearly outperform $P_I$.

The conclusion of [36] states that two-row intersection cuts are not strong enough to consistently beat one-row cuts, if considering one-row models from linear combinations of rows of the simplex tableau. So while the situation for $P_{\text{lifting}}$ is less definite, Table 4 shows that cuts from $P_{S\text{-free}}$ and $P_{IU}$ are probably not strong enough either. On the other hand, if one could generate practically cuts for the fully-strengthened

model $P_{\text{full}}$, it may be a promising provider of useful cuts beyond GMIs. Note that this is not a side-effect of taking the maximum of all previous columns. Indeed, without this operation, $P_{\text{full}}$ would still close 36.3 % of gap on average, despite having fewer proofs of exact separation than $P_I$, $P_{S-\text{free}}$ or $P_{IU}$.

## 7 Cuts from several tableaux

In all the computations performed so far, we limited ourselves to rank-1 valid inequalities from *one* simplex tableau. We mentioned in the previous section that most of the two-row cuts generated in [36] could be obtained as GMIs from linear combinations of rows of the tableau. And the various simplex tableaux describing a linear problem are precisely linear combinations of the rows of each other. Moreover, Balas and Perregaard [9] showed that the most-violated inequality for a simple disjunction on one variable could be obtained through the lift-and-project method as a GMI from a specific simplex tableau. They demonstrate that the corresponding cuts can be generated efficiently and are useful. Dash and Goycoolea [19] also showed that, in general, one can generate stronger rank-1 cuts by considering multiple LP tableaux. This emphasizes the need to reproduce our results for more than one basis of the LP relaxation.

In particular, the small but consistent advantage that two-row intersection cuts enjoy over GMIs read directly from the tableau in terms of gap closure could vanish in the presence of GMIs from other bases. This could be the case, for example, if two-row cuts close more gap only because so many more of them can be generated. Or, to the contrary, the two-row cuts could become significantly stronger when read from several bases.

In order to answer these questions, we need a method to explore different bases of the LP relaxation. We use the relax-and-cut method proposed by Fischetti and Salvagnin [29] in the context of the generation of GMI cuts. It is a simple way to find feasible bases of the LP relaxation, and it has been successfully used as one of several main components to approximate the first split closure [30].

The method consists in computing a round of GMIs from the optimal tableau, then adding them as Lagrangean penalties to the objective function instead of as additional constraints to the feasible region. When re-optimizing over the LP relaxation with this modified objective function, one may obtain a different basis. The process can be iterated in order to find more bases. We refer to [29] for a more detailed exposition of the method.

Tables 5 and 6 show the results for two-row intersection cuts and full two-row models, respectively. Not shown in the table, for clarity, is that 21 bases are considered for each instance when considering multiple bases, except for `air03`, `mod010`, `pp08a`, `pp08acuts` and `swath` (where the relax-and-cut method only explores 1, 1, 2, 2 and 12 bases respectively). Note also that we limit ourselves to 100 models per tableau in this experiment, in order to keep the total number of models per instance manageable. The instances `10teams` and `arki001` are missing from these tables, because we could not find an integer feasible solution within the prescribed iteration limit (100 branch-and-bound nodes). We need such a solution in our relax-and-cut framework.

**Table 5** Relax and cut: two-row intersection cuts

| Name | From 1 basis | | | | | From up to 21 bases | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | GMIs | | $P_I$ | | | GMIs | | $P_I$ | | |
| | Cuts | %gc | Cuts | %gc | ex. | Cuts | %gc | Cuts | %gc | ex. |
| air03 | 3 | 100 | 31 | 100 | ✓ | 3 | 100 | 31 | 100 | ✓ |
| air04 | 100 | 7.9 | 20 | 7.9 | | 100 | 7.9 | 20 | 7.9 | |
| air05 | 101 | 4.6 | 8 | 4.8 | ✓ | 101 | 4.6 | 8 | 4.8 | |
| bell3a | 14 | 39.0 | 11 | 52.5 | ✓ | 28 | 54.2 | 24 | 59.9 | ✓ |
| bell5 | 17 | 14.5 | 11 | 14.9 | ✓ | 23 | 34.7 | 58 | 35.6 | ✓ |
| blend2 | 5 | 16.0 | 3 | 16.9 | ✓ | 14 | 20.7 | 7 | 23.5 | ✓ |
| cap6000 | 2 | 39.9 | 6 | 40.1 | ✓ | 21 | 44.6 | 12 | 44.8 | ✓ |
| danoint | 31 | 0.3 | 35 | 1.7 | | 80 | 0.7 | 47 | 1.7 | |
| dcmulti | 36 | 47.8 | 43 | 49.9 | ✓ | 77 | 64.3 | 74 | 68.1 | ✓ |
| egout | 8 | 31.9 | 12 | 94.2 | ✓ | 20 | 50.3 | 24 | 99.7 | ✓ |
| fast0507 | 100 | 1.7 | 7 | 1.7 | | 100 | 1.7 | 7 | 1.7 | |
| fiber | 22 | 69.2 | 21 | 70.8 | ✓ | 60 | 79.7 | 55 | 79.9 | ✓ |
| fixnet6 | 11 | 22.3 | 18 | 30.6 | ✓ | 29 | 33.0 | 42 | 47.4 | ✓ |
| flugpl | 7 | 10.8 | 8 | 13.3 | ✓ | 8 | 11.3 | 8 | 13.3 | ✓ |
| gen | 6 | 1.3 | 13 | 9.5 | | 28 | 40.1 | 36 | 40.3 | ✓ |
| gesa2 | 40 | 27.6 | 62 | 30.4 | | 104 | 35.2 | 200 | 37.6 | |
| gesa2_o | 70 | 30.7 | 93 | 32.0 | ✓ | 174 | 47.4 | 259 | 58.2 | |
| gesa3 | 37 | 20.5 | 28 | 43.4 | ✓ | 55 | 33.7 | 215 | 49.7 | |
| gesa3_o | 64 | 50.5 | 85 | 63.1 | | 109 | 54.0 | 399 | 66.0 | |
| gt2 | 11 | 47.2 | 8 | 47.2 | ✓ | 33 | 58.1 | 23 | 58.7 | ✓ |
| harp2 | 22 | 22.8 | 20 | 25.2 | | 99 | 28.7 | 37 | 30.1 | |
| khb05250 | 19 | 73.2 | 15 | 73.2 | ✓ | 40 | 86.9 | 48 | 91.0 | ✓ |
| l152lav | 7 | 2.0 | 39 | 2.0 | | 38 | 13.8 | 24 | 13.8 | ✓ |
| lseu | 5 | 20.5 | 9 | 21.7 | ✓ | 44 | 41.2 | 19 | 41.2 | ✓ |
| markshare1 | 6 | 0.0 | 1 | 0.0 | ✓ | 12 | 0.0 | 1 | 0.0 | ✓ |
| markshare2 | 7 | 0.0 | 2 | 0.0 | ✓ | 14 | 0.0 | 2 | 0.0 | ✓ |
| mas74 | 11 | 6.7 | 7 | 6.7 | ✓ | 28 | 7.0 | 8 | 7.0 | ✓ |
| mas76 | 10 | 6.4 | 6 | 6.4 | ✓ | 53 | 6.7 | 6 | 6.7 | ✓ |
| misc03 | 4 | 8.6 | 7 | 8.6 | ✓ | 17 | 17.6 | 25 | 17.6 | ✓ |
| misc06 | 16 | 28.5 | 17 | 48.3 | | 24 | 44.7 | 42 | 88.6 | |
| misc07 | 5 | 0.7 | 9 | 0.7 | ✓ | 8 | 0.7 | 20 | 0.7 | ✓ |
| mitre | 101 | 50.7 | 225 | 50.7 | ✓ | 316 | 87.8 | 915 | 87.8 | ✓ |
| mkc | 31 | 1.4 | 53 | 1.4 | | 139 | 25.5 | 74 | 32.9 | ✓ |
| mod008 | 5 | 21.6 | 1 | 21.6 | ✓ | 33 | 35.9 | 7 | 35.9 | ✓ |
| mod010 | 5 | 100 | 15 | 100 | ✓ | 5 | 100 | 15 | 100 | ✓ |
| mod011 | 22 | 31.3 | 39 | 34.8 | ✓ | 153 | 40.8 | 127 | 46.2 | ✓ |
| modglob | 28 | 17.3 | 30 | 24.7 | ✓ | 91 | 50.6 | 70 | 59.2 | ✓ |

**Table 5** continued

| Name | From 1 basis | | | | | From up to 21 bases | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | GMIs | | $P_I$ | | | GMIs | | $P_I$ | | |
| | Cuts | %gc | Cuts | %gc | ex. | Cuts | %gc | Cuts | %gc | ex. |
| noswot | 14 | 0.0 | 9 | 0.0 | ✓ | 15 | 0.0 | 53 | 0.0 | |
| nw04 | 2 | 29.8 | 2 | 29.8 | ✓ | 22 | 66.1 | 11 | 66.1 | ✓ |
| p0033 | 4 | 34.4 | 5 | 34.8 | ✓ | 18 | 53.9 | 9 | 53.9 | ✓ |
| p0201 | 14 | 0.4 | 15 | 6.5 | ✓ | 63 | 13.9 | 117 | 18.8 | ✓ |
| p0282 | 23 | 3.2 | 27 | 4.6 | | 78 | 9.9 | 31 | 12.4 | ✓ |
| p0548 | 31 | 61.7 | 32 | 61.7 | ✓ | 76 | 81.5 | 96 | 83.3 | ✓ |
| p2756 | 81 | 51.7 | 72 | 52.2 | ✓ | 167 | 95.0 | 161 | 95.1 | ✓ |
| pk1 | 15 | 0.0 | 1 | 0.0 | ✓ | 16 | 0.0 | 2 | 0.0 | ✓ |
| pp08a | 53 | 51.4 | 79 | 76.2 | ✓ | 72 | 60.0 | 89 | 85.3 | ✓ |
| pp08acuts | 41 | 31.5 | 39 | 38.8 | ✓ | 41 | 31.5 | 42 | 39.6 | ✓ |
| qiu | 23 | 1.7 | 13 | 1.9 | | 107 | 5.4 | 66 | 6.0 | ✓ |
| qnet1 | 21 | 11.8 | 9 | 12.7 | ✓ | 124 | 24.4 | 42 | 29.3 | |
| qnet1_o | 10 | 23.3 | 13 | 30.2 | ✓ | 52 | 39.8 | 61 | 48.3 | ✓ |
| rentacar | 13 | 5.0 | 11 | 5.5 | | 14 | 5.0 | 20 | 5.6 | |
| rgn | 12 | 5.0 | 7 | 5.0 | ✓ | 88 | 35.3 | 30 | 38.3 | ✓ |
| rout | 29 | 3.6 | 11 | 4.1 | | 102 | 8.1 | 21 | 9.4 | ✓ |
| set1ch | 121 | 28.2 | 123 | 55.5 | ✓ | 155 | 29.0 | 154 | 60.2 | ✓ |
| seymour | 50 | 6.2 | 5 | 6.2 | ✓ | 76 | 10.2 | 40 | 10.2 | ✓ |
| stein27 | 21 | 0.0 | 2 | 0.0 | ✓ | 22 | 0.0 | 3 | 0.0 | ✓ |
| stein45 | 16 | 0.0 | 2 | 0.0 | ✓ | 16 | 0.0 | 2 | 0.0 | ✓ |
| swath | 13 | 6.4 | 30 | 6.6 | ✓ | 52 | 22.0 | 68 | 22.0 | |
| vpm1 | 15 | 6.2 | 18 | 14.6 | ✓ | 32 | 9.1 | 40 | 23.2 | ✓ |
| vpm2 | 21 | 7.3 | 24 | 24.4 | ✓ | 57 | 13.8 | 68 | 33.1 | |
| Average | 27.2 | **22.4** | 26.1 | **27.0** | $\frac{46}{60}$ | 62.4 | **33.0** | 70.2 | **38.3** | $\frac{45}{60}$ |

From one basis, GMIs close 22.4 % of gap on average, and two-row intersection cuts close five more percentage points at 27.0 %. When using relax-and-cut, we generate more GMIs and two-row cuts, as we compute them from up to 21 different bases. Then, GMIs close 33.0 % and two-row intersection cuts close again five more percentage points at 38.3 %. Similar results hold for cuts from the full two-row model $P_{\text{full}}$. From one basis, strengthened two-row cuts, at 34.7 %, close 12 more percentage points of gap than GMIs. From up to 21 different bases, they close also 12 more percentage points than GMIs, at 45.5 %.

Contrary to our earlier suggestions, the strengthening provided by two-row cuts on top of GMIs seems to consistently carry over to the situation where we generate cuts from several bases. This conclusion holds both in the case of intersection cuts and in the case of cuts from the full model.

**Table 6** Relax and cut: full two-row models

| Name | From 1 basis | | | | | From up to 21 bases | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | GMIs | | $P_{\text{full}}$ | | | GMIs | | $P_{\text{full}}$ | | |
| | Cuts | %gc | Cuts | %gc | ex. | Cuts | %gc | Cuts | %gc | ex. |
| air03 | 3 | 100 | 31 | 100 | ✓ | 3 | 100 | 31 | 100 | ✓ |
| air04 | 100 | 7.9 | 13 | 7.9 | | 100 | 7.9 | 13 | 7.9 | |
| air05 | 101 | 4.6 | 8 | 4.6 | | 101 | 4.6 | 8 | 4.6 | |
| bell3a | 14 | 39.0 | 29 | 55.7 | ✓ | 28 | 54.2 | 24 | 59.9 | ✓ |
| bell5 | 17 | 14.5 | 26 | 19.3 | ✓ | 23 | 34.7 | 41 | 50.3 | |
| blend2 | 5 | 16.0 | 11 | 20.0 | ✓ | 14 | 20.7 | 24 | 26.6 | ✓ |
| cap6000 | 2 | 39.9 | 2 | 39.9 | | 21 | 44.6 | 8 | 44.6 | |
| danoint | 31 | 0.3 | 37 | 1.7 | | 80 | 0.7 | 49 | 1.7 | |
| dcmulti | 36 | 47.8 | 105 | 58.0 | ✓ | 77 | 64.3 | 139 | 78.7 | ✓ |
| egout | 8 | 31.9 | 21 | 98.4 | ✓ | 20 | 50.3 | 47 | 100.0 | ✓ |
| fast0507 | 100 | 1.7 | 5 | 1.7 | | 100 | 1.7 | 5 | 1.7 | |
| fiber | 22 | 69.2 | 150 | 80.3 | ✓ | 60 | 79.7 | 111 | 84.7 | |
| fixnet6 | 11 | 22.3 | 69 | 36.5 | ✓ | 29 | 33.0 | 79 | 50.5 | |
| flugpl | 7 | 10.8 | 6 | 50.8 | ✓ | 8 | 11.3 | 7 | 68.6 | ✓ |
| gen | 6 | 1.3 | 7 | 39.2 | | 28 | 40.1 | 49 | 76.5 | |
| gesa2 | 40 | 27.6 | 100 | 62.6 | | 104 | 35.2 | 392 | 70.0 | |
| gesa2_o | 70 | 30.7 | 113 | 34.5 | ✓ | 174 | 47.4 | 377 | 70.2 | |
| gesa3 | 37 | 20.5 | 65 | 49.7 | | 55 | 33.7 | 296 | 75.1 | |
| gesa3_o | 64 | 50.5 | 91 | 70.9 | ✓ | 109 | 54.0 | 524 | 79.3 | |
| gt2 | 11 | 47.2 | 78 | 58.6 | | 33 | 58.1 | 52 | 60.4 | |
| harp2 | 22 | 22.8 | 20 | 22.8 | | 99 | 28.7 | 27 | 28.7 | |
| khb05250 | 19 | 73.2 | 218 | 91.4 | ✓ | 40 | 86.9 | 134 | 95.8 | |
| l152lav | 7 | 2.0 | 4 | 3.1 | | 38 | 13.8 | 29 | 14.9 | |
| lseu | 5 | 20.5 | 57 | 70.2 | ✓ | 44 | 41.2 | 37 | 78.2 | |
| markshare1 | 6 | 0.0 | 1 | 0.0 | | 12 | 0.0 | 1 | 0.0 | |
| markshare2 | 7 | 0.0 | 1 | 0.0 | | 14 | 0.0 | 2 | 0.0 | |
| mas74 | 11 | 6.7 | 10 | 6.9 | | 28 | 7.0 | 8 | 7.0 | |
| mas76 | 10 | 6.4 | 6 | 6.4 | | 53 | 6.7 | 6 | 6.7 | |
| misc03 | 4 | 8.6 | 9 | 8.6 | ✓ | 17 | 17.6 | 52 | 17.7 | ✓ |
| misc06 | 16 | 28.5 | 25 | 48.8 | ✓ | 24 | 44.7 | 71 | 91.7 | |
| misc07 | 5 | 0.7 | 13 | 0.7 | ✓ | 8 | 0.7 | 55 | 0.8 | |
| mitre | 101 | 50.7 | 240 | 50.7 | ✓ | 316 | 87.8 | 1131 | 87.8 | |
| mkc | 31 | 1.4 | 46 | 1.4 | | 139 | 25.5 | 75 | 45.2 | |
| mod008 | 5 | 21.6 | 4 | 21.6 | | 33 | 35.9 | 8 | 36.4 | |
| mod010 | 5 | 100 | 15 | 100 | ✓ | 5 | 100 | 15 | 100 | ✓ |
| mod011 | 22 | 31.3 | 19 | 31.4 | | 153 | 40.8 | 69 | 40.9 | |
| modglob | 28 | 17.3 | 36 | 29.5 | | 91 | 50.6 | 203 | 70.4 | |

**Table 6** continued

| Name | From 1 basis | | | | | From up to 21 bases | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | GMIs | | $P_{\text{full}}$ | | | GMIs | | $P_{\text{full}}$ | | |
| | Cuts | %gc | Cuts | %gc | ex. | Cuts | %gc | Cuts | %gc | ex. |
| noswot | 14 | 0.0 | 12 | 0.0 | ✓ | 15 | 0.0 | 64 | 0.0 | ✓ |
| nw04 | 2 | 29.8 | 4 | 29.9 | | 22 | 66.1 | 11 | 68.3 | |
| p0033 | 4 | 34.4 | 59 | 100.0 | ✓ | 18 | 53.9 | 76 | 100.0 | ✓ |
| p0201 | 14 | 0.4 | 32 | 15.8 | ✓ | 63 | 13.9 | 90 | 16.2 | |
| p0282 | 23 | 3.2 | 95 | 41.7 | ✓ | 78 | 9.9 | 99 | 49.7 | |
| p0548 | 31 | 61.7 | 108 | 99.9 | ✓ | 76 | 81.5 | 243 | 100.0 | ✓ |
| p2756 | 81 | 51.7 | 193 | 72.9 | | 167 | 95.0 | 179 | 95.6 | |
| pk1 | 15 | 0.0 | 1 | 0.0 | | 16 | 0.0 | 2 | 0.0 | |
| pp08a | 53 | 51.4 | 85 | 76.2 | ✓ | 72 | 60.0 | 111 | 85.4 | ✓ |
| pp08acuts | 41 | 31.5 | 43 | 43.0 | | 41 | 31.5 | 49 | 43.5 | |
| qiu | 23 | 1.7 | 15 | 1.9 | | 107 | 5.4 | 32 | 5.6 | |
| qnet1 | 21 | 11.8 | 7 | 12.0 | | 124 | 24.4 | 27 | 24.4 | |
| qnet1_o | 10 | 23.3 | 39 | 25.1 | | 52 | 39.8 | 29 | 40.1 | |
| rentacar | 13 | 5.0 | 24 | 5.7 | | 14 | 5.0 | 14 | 5.6 | |
| rgn | 12 | 5.0 | 32 | 25.1 | ✓ | 88 | 35.3 | 46 | 68.1 | |
| rout | 29 | 3.6 | 13 | 3.8 | | 102 | 8.1 | 23 | 8.7 | |
| set1ch | 121 | 28.2 | 134 | 55.5 | ✓ | 155 | 29.0 | 168 | 60.4 | ✓ |
| seymour | 50 | 6.2 | 21 | 10.2 | ✓ | 76 | 10.2 | 31 | 13.4 | |
| stein27 | 21 | 0.0 | 4 | 0.0 | ✓ | 22 | 0.0 | 6 | 0.0 | ✓ |
| stein45 | 16 | 0.0 | 2 | 0.0 | ✓ | 16 | 0.0 | 2 | 0.0 | ✓ |
| swath | 13 | 6.4 | 25 | 31.4 | | 52 | 22.0 | 84 | 34.0 | |
| vpm1 | 15 | 6.2 | 25 | 15.9 | ✓ | 32 | 9.1 | 101 | 25.8 | ✓ |
| vpm2 | 21 | 7.3 | 44 | 33.6 | ✓ | 57 | 13.8 | 115 | 48.7 | |
| Average | 27.2 | **22.4** | 45.1 | **34.7** | $\frac{31}{60}$ | 62.4 | **33.0** | 96.7 | **45.5** | $\frac{16}{60}$ |

## 8 More than two rows

So far we have focused on two-row cuts because we have comparison points for them that are better understood than in the general multi-row case. Our separator however is not limited to two-row cuts, and we present in this section results with more than two rows. To simplify the presentation, we only cover the $k$-row extension of the $P_{\text{full}}$ model. The interested reader can find the raw data for the graphs of this section in [40].

We first focus on the speed of our separator. Fig. 1 shows the number of instances for which the separation is exact. The latter number quickly drops when going from 1- to 5-row cuts, but then stays around 15 from 5- through 15-row cuts. Figure 2 shows the geometric mean of the running times over the 62 MIPLIB instances we use. Computing times indeed increase with the number of rows, but for up to $k = 15$, we do not see yet a dramatic growth in the computational cost.

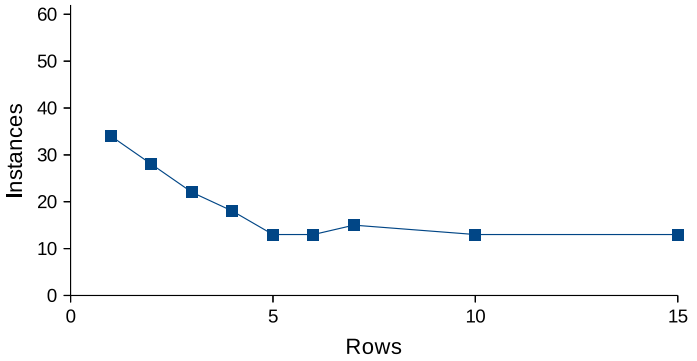


**Fig. 1** Number of instances with exact separation (of 62 instances)

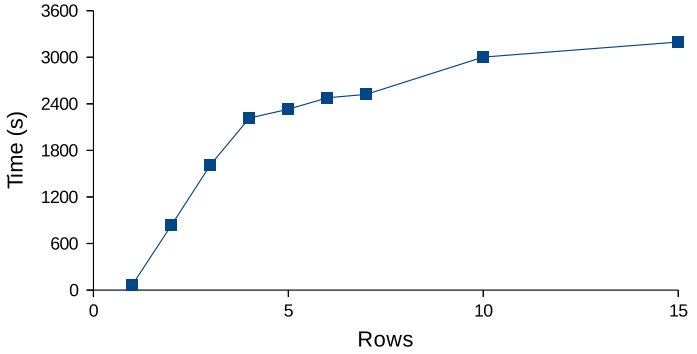


**Fig. 2** Geometric mean of the running time (on 62 instances)

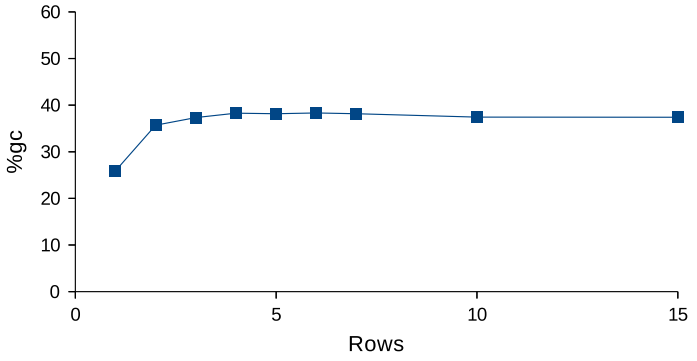


**Fig. 3** Average percentage of gap closed (on 62 instances)

The average of the percentage of gap closed by the $k$-row cuts is plotted on Fig. 3. That value reaches 37 % for $k = 3$, but does not exceed 39 % for any other $k \leq 15$, indicating that there would be limited interest in separating $k$-row cuts with these values of $k$. Of course, this may be due to our separator being unable to separate as many cuts for the models with more rows. However, we will see in the next figure that the limited increase in gap closure happens despite a significant increase in the number of cuts generated.

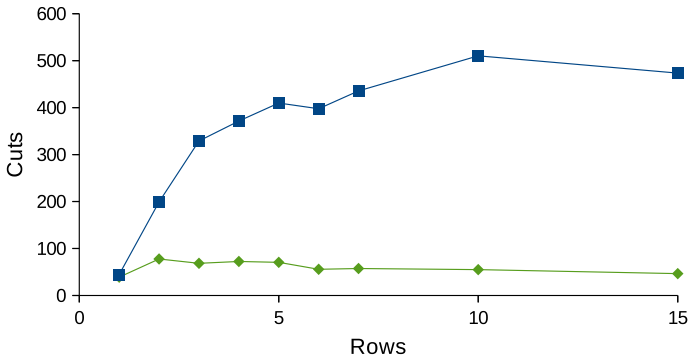


**Fig. 4** Number of cuts generated (□) and tight at the end (◊), on 62 instances

Indeed Fig. 4 displays, for each value of $k$, the average number of cuts generated by our separator and the number among them that are tight at $x^*$ at the end of Algorithm 3. Recall that the number of $k$-row models considered is at most $m$ independently of $k$. The number of cuts that are tight at the end stays around 60 in average and varies very little for the different values of $k$. Meanwhile, the number of cuts that had to be generated raises significantly with increasing $k$. This means that when generating multi-row cuts with more rows, one needs to compute many more cuts, indicating that the complexity of the facial structure of multi-row models may raise one more hurdle for the use of multi-row cuts with many rows.

## 9 Summary

We implemented a separator for arbitrary mixed-integer sets. Computationally, the task is inherently costly, and a separator with such a generic scope is bound to be slow in practice. Our first naive implementation was unable to provide meaningful results, even for some of the smallest instances in MIPLIB 3. To partially mitigate the issue, we developed a series of tricks, mainly based on the concept of lifting inequalities that are valid for faces of the feasible region.

Using our improved separator, we show that on average over our testset, two-row intersection cuts close around 6 % more gap than GMIs. Further, cuts from fully-strengthened two-row models close an additional 8 % of the gap. We remark however that strengthening only partially the two-row models yields almost no improvement over intersection cuts. We then try generating GMIs and two-row cuts from several feasible bases of the LP relaxation, with surprisingly similar results. This leads us to conclude that the usefulness of two-row cuts, although limited, is not canceled by the effect of GMIs when considering cuts from several tableaux.

We also use our implementation to separate multi-row cuts with more than two rows. The running times show that the separator scales acceptably for up to 15 rows. On the other hand, the percentage of gap closure we obtain seems to tail off after 4- or 5-row cuts, indicating that there would be little interest in generating multi-row cuts with more than 5 rows, unless we can use much more than 15 rows.

**Acknowledgments** We are grateful to three anonymous referees for their valuable comments.

## References

1. Achterberg, T., Koch, T., Martin, A.: MIPLIB 2003. Oper. Res. Lett. **34**(4), 361–372 (2006)
2. Andersen, K., Louveaux, Q., Weismantel, R.: Mixed-integer sets from two rows of two adjacent simplex bases. Math. Program. **124**, 455–480 (2010)
3. Andersen, K., Louveaux, Q., Weismantel, R., Wolsey, L.: Inequalities from two rows of a simplex tableau. In: Fischetti, M., Williamson, D. (eds.) Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science, vol. 4513, pp. 1–15. Springer, Berlin (2007)
4. Applegate, D., Bixby, R., Chvátal, V., Cook, W.: TSP cuts which do not conform to the template paradigm. Lect. Notes Comput. Sci. **2241**, 261–303 (2001)
5. Atamtürk, A.: http://ieor.berkeley.edu/~atamturk/data/
6. Atamtürk, A.: Sequence independent lifting for mixed-integer programming. Oper. Res. **52**(3), 487–490 (2004)
7. Balas, E.: Intersection cuts—a new type of cutting planes for integer programming. Oper. Res. **1**(19), 19–39 (1971)
8. Balas, E., Perregaard, M.: Lift-and-project for mixed 0–1 programming: recent progress. Discrete Appl. Math. **123**(1–3), 129–154 (2002)
9. Balas, E., Perregaard, M.: A precise correspondence between lift-and-project cuts, simple disjunctive cuts, and mixed integer Gomory cuts for 0–1 programming. Math. Program. **94**(2–3), 221–245 (2003)
10. Basu, A., Bonami, P., Cornuéjols, G., Margot, F.: Experiments with two-row cuts from degenerate tableaux. INFORMS J. Comput. **23**, 578–590 (2011)
11. Basu, A., Conforti, M., Cornuéjols, G., Giacomo, Z.: Minimal inequalities for an infinite relaxation of integer programs. SIAM J. Discrete Math. **24**, 158–168 (2010)
12. Basu, A., Cornuéjols, G., Molinaro, M.: A probabilistic analysis of the strength of the split and triangle closures. In: Günlük, O., Woeginger, G.J. (eds.) Integer Programming and Combinatoral Optimization. Lecture Notes in Computer Science, vol. 6655, pp. 27–38. Springer, Berlin (2011)
13. Bixby, R.E., Ceria, S., McZeal, C.M., Savelsbergh, M.W.P.: An updated mixed integer programming library: MIPLIB 3.0. Optima **58**, 12–15 (1998)
14. Boyd, A.E.: Generating Fenchel cutting planes for knapsack polyhedra. SIAM J. Optim. **3**(4), 734–750 (1993)
15. Chvátal, V., Cook, W., Espinoza, D.: Local cuts for mixed-integer programming. Math. Program. Comput. **5**(2), 171–200 (2013)
16. Conforti, M., Cornuéjols, G., Zambelli, G.: A geometric perspective on lifting. Oper. Res. **59**, 569–577 (2011)
17. Cornuéjols, G., Margot, F.: On the facets of mixed integer programs with two integer variables and two constraints. Math. Program. **120**(2), 429–456 (2009)
18. Dash, S., Dey, S.S., Günlük, O.: Two dimensional lattice-free cuts and asymmetric disjunctions for mixed-integer polyhedra. Math. Program. **135**(1–2), 221–254 (2012)
19. Dash, S., Goycoolea, M.: A heuristic to generate rank-1 GMI cuts. Math. Program. Comput. **2**(3–4), 231–257 (2010)
20. Dash, S., Günlük, O., Vielma, J.P.: Computational experiments with cross and crooked cross cuts. IBM Technical Report (2011)
21. Dey, S.S., Wolsey, L.A.: Two row mixed-integer cuts via lifting. Math. Program. **124**, 143–174 (2010)
22. Dey, S.S., Lodi, A., Tramontani, A., Wolsey, L.A.: Experiments with two row tableau cuts. In: Eisenbrand, F., Bruce Shepherd, F. (eds.) Integer Programming and Combinatorial Optimization. Proceedings of the 14th International Conference, IPCO 2010, Lausanne, Switzerland, June 9–11, 2010. Lecture Notes in Computer Science, vol. 6080, pp. 424–437. Springer, Berlin (2010)
23. Dey, S.S., Lodi, A., Tramontani, A., Wolsey, L.A.: On the practical strength of two-row tableau cuts. INFORMS J. Comput. **26**(2), 222–237 (2014)
24. Dey, S.S., Louveaux, Q.: Split rank of triangle and quadrilateral inequalities. Math. Oper. Res. **36**(3), 432–461 (2011)
25. Dey, S.S., Richard, J.-P.P.: Linear-programming-based lifting and its application to primal cutting-plane algorithms. INFORMS J. Comput. **21**(1), 137–150 (2010)

26. Dey, S.S., Wolsey, L.A.: Lifting integer variables in minimal inequalities corresponding to lattice-free triangles. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) Integer Programming and Combinatorial Optimization. Proccedings of the13th International Conference, IPCO 2008, Bertinoro, Italy, May 26–28, 2008. Lecture Notes in Computer Science, vol. 5035, pp. 463–475. Springer, Berlin (2008)
27. Dey, S.S., Wolsey, L.A.: Constrained infinite group relaxations of MIPs. CORE Discussion Papers 2009033, Université catholique de Louvain, Center for Operations Research and Econometrics (CORE) (2009)
28. Espinoza, D.G.: Computing with multi-row Gomory cuts. Oper. Res. Lett. **38**(2), 115–120 (2010)
29. Fischetti, M., Salvagnin, D.: A relax-and-cut framework for gomory's mixed-integer cuts. Math. Program. Comput. **3**, 79–102 (2011)
30. Fischetti, M., Salvagnin, D.: Approximating the split closure. INFORMS J. Comput. **25**(4), 808–819 (2013)
31. Fukasawa, R., Günlük, O.: Strengthening lattice-free cuts using non-negativity. Discrete Optim. **8**(2), 229–245 (2011)
32. Fukasawa, R., Goycoolea, M.: On the exact separation of mixed integer knapsack cuts. Math. Program. **128**, 19–41 (2011)
33. Gomory, R.E.: On the relation between integer and noninteger solutions to linear programs. Proc. Natl. Acad. Sci. **53**, 260–265 (1965)
34. Gomory, R.E.: Some polyhedra related to combinatorial problems. Linear Algebra Appl. **2**(4), 451–558 (1969)
35. Gomory, R.E., Johnson, E.L.: Some continuous functions related to corner polyhedra, part I. Math. Program. **3**, 23–85 (1972)
36. Louveaux, Q., Poirrier, L.: An algorithm for the separation of two-row cuts. Math. Program. **143**(1–2), 111–146 (2014)
37. Margot, F.: MIPLIB3 C V2. http://wpweb2.tepper.cmu.edu/fmargot/ (2009)
38. Nemhauser, G.L., Wolsey, L.A.: A recursive procedure to generate all cuts for 0–1 mixed integer programs. Math. Program. **46**, 379–390 (1990)
39. Perregaard, M., Balas, E.: Generating cuts from multiple-term disjunctions. In: Aardal, K., Gerards, B. (eds.) Integer Programming and Combinatorial Optimization. Lecture Notes in Computer Science, vol. 2081, pp. 348–360. Springer, Berlin (2001)
40. Poirrier, L.: Multi-row approaches to cutting plane generation. PhD thesis, University of Liège (2012)
41. Richard, J.-P.P., de Farias Jr, I.R., Nemhauser, G.L.: Lifted inequalities for 0–1 mixed integer programming: basic theory and algorithms. Math. Program. **98**(1–3), 89–113 (2003)
42. Schrijver, A.: Theory of linear and integer programming. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley, New York (1998)