Math. Prog. Comp. (2009) 1:165–199
DOI 10.1007/s12532-009-0006-4

FULL LENGTH PAPER

Generating lift-and-project cuts from the LP simplex tableau: open source implementation and testing of new variants

Egon Balas · Pierre Bonami

Received: 24 November 2008 / Accepted: 4 August 2009 / Published online: 10 September 2009

Abstract Lift-and-project cuts for mixed integer programs (MIP), derived from a disjunction on an integer-constrained fractional variable, were originally (Balas et al. in Math program 58:295–324, 1993) generated by solving a higher-dimensional cut generating linear program (CGLP). Later, a correspondence established (Balas and Perregaard in Math program 94:221–245, 2003) between basic feasible solutions to the CGLP and basic (not necessarily feasible) solutions to the linear programming relaxation LP of the MIP, has made it possible to mimic the process of solving the CGLP through certain pivots in the LP tableau guaranteed to improve the CGLP objective function. This has also led to an alternative interpretation of lift-and-project (L&P) cuts, as mixed integer Gomory cuts from various (in general neither primal nor dual feasible) LP tableaus, guaranteed to be stronger than the one from the optimal tableau. In this paper we analyze the relationship between a pivot in the LP tableau and the (unique) corresponding block pivot (sequence of pivots) in the CGLP tableau. Namely, we show how a single pivot in the LP defines a sequence (potentially as long as the number of variables) of pivots in the CGLP, and we identify this sequence. Also, we give a new procedure for finding in a given LP tableau a pivot that produces the maximum improvement in the CGLP objective (which measures the amount of violation of

Egon Balas's research was supported by the National Science Foundation through grant #DMI-0352885 and by the Office of Naval Research through contract N00014-03-1-0133.
Pierre Bonami's research was supported by ANR grant BLAN06-138894.

E. Balas (✉)
Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: eb17@andrew.cmu.edu

P. Bonami
Laboratoire d'Informatique Fondamentale, CNRS, Aix-Marseille Universités,
163 Av. de Luminy, 13228 Marseille Cedex 9, France
e-mail: pierre.bonami@lif.univ-mrs.fr

the resulting cut by the current LP solution). Further, we introduce a procedure called iterative disjunctive modularization. In the standard procedure, pivoting in the LP tableau optimizes the multipliers with which the inequalities on each side of the disjunction are weighted in the resulting cut. Once this solution has been obtained, a strengthening step is applied that uses the integrality constraints (if any) on the variables on each side of the disjunction to improve the cut coefficients by choosing optimal values for the elements of a certain monoid. Iterative disjunctive modularization is a procedure for approximating the simultaneous optimization of both the continuous multipliers and the integer elements of the monoid. All this is discussed in the context of a CGLP with a more general normalization constraint than the standard one used in (Balas and Perregaard in Math program 94:221–245, 2003), and the expressions that describe the above mentioned correspondence are accordingly generalized. Finally, we summarize our extensive computational experience with the above procedures.

Keywords Integer programming · Lift-and-project · Iterative disjunctive modularization

Mathematics Subject Classification (2000) 90C11

1 Introduction

The revolution of the last 15 years in the state of the art of integer programming was brought about, besides faster computers and more efficient linear programming codes, also by improved cutting plane techniques. Lift-and-project (L&P) cuts were the first mixed integer cuts to be generated in rounds rather than one by one, the first to be generated locally and lifted into globally valid cuts, and the first to be embedded into a branch-and-bound framework. Soon after the success of L&P cuts [4,5], it was shown [6] that mixed integer Gomory (MIG) cuts used in the same manner could also enhance the performance of MIP solvers. Thus, during the nineties a number of different cut families (cover and flow cover inequalities, MIG cuts, simple disjunctive cuts, MIR cuts etc.) became part of the toolkit of commercial MIP solvers and have led to a radical improvement of their performance. The L&P cuts themselves, however, were found to be computationally too expensive to be incorporated into commercial codes, as each such cut came at the price of solving a Cut Generating Linear Program (CGLP) in a higher dimensional space. It was not until a few years later, when a way was found [7] to generate L&P cuts by pivoting in the original simplex tableau, without constructing the higher dimensional CGLP, that these cuts became sufficiently cost-effective to be incorporated into a state-of-the-art MIP solver, where they soon proved their value [13] and became the default cut generator.

Although the algorithm for generating L&P cuts from the original simplex tableau is now in practical use and has contributed to solving countless hard integer programs, its implementation was until recently commercial property not publicly available, which made it harder for researchers to experiment with different versions of it. In this paper we discuss an implementation of this algorithm in the COIN-OR framework, publicly available [10] since September 2006, and compare three different variants of it.

However, before embarking on the description of the three Variants, we devote some space to bringing about a better understanding of the crucial correspondence that makes it possible to mimic the solution of the CGLP through pivots in the LP simplex tableau. First, we restate this correspondence for a CGLP with a more general normalization constraint than the one used in [7], and derive the corresponding new expressions for the reduced costs and the evaluation functions. This is essential in view of the important role the normalization constraint plays in determining the strength of the resulting cut. Next, we explore the correspondence between a pivot in the LP simplex tableau, and a block pivot, i.e. a sequence of pivots, in the CGLP tableau, and illustrate it on numerical examples.

As to the three variants to be tested, Variant 1 is a slightly modified version of the original algorithm [7] for generating L&P cuts by pivoting in the original LP tableau which incorporates the various improvements proposed in [13,14], whereas the other two variants contain substantial changes in the algorithm, which give rise to different pivot sequences and therefore different cuts. Variant 2 uses a new rule for choosing the entering variable in the pivoting procedure. Instead of first choosing a most promising pivot row and then identifying the best column in that row, this version of the algorithm first identifies all candidate rows for an improving pivot, then chooses the pivot element as the best one among the entries of all the candidate rows. Variant 3 uses recursive disjunctive modularization of the source row. Namely, rather than first generating an unstrengthened "deepest" L&P cut through a sequence of pivots in the original LP tableau and then strengthening the end product by modular arithmetic, this version replaces the source row with its disjunctive modularization, and after each pivot it again applies the disjunctive modularization to the resulting transformed source row. Each of the three Variants gives rise to sequences of pivots different from each other. In the case of both Variants 2 and 3, each pivot is guaranteed to produce an improvement in cut strength at least equal to that produced by the corresponding pivot of Variant 1, but this additional improvement comes at some computational cost. After describing each of the three Variants, we compare them on a battery of MIPLIB test problems and assess the results by trying to identify the merits and demerits of each Variant.

2 Lift-and-project cuts

Consider a problem of the form

$$\min\{cx : x \in P, x_j \in \{0, 1\}, j = 1, \ldots, p\} \tag{MIP}$$

and its linear programming relaxation

$$\min\{cx : x \in P\}, \tag{LP}$$

where P is the polyhedron defined by the system

$$\begin{aligned} Ax &\geq b \\ -x_j &\geq -1 \quad j = 1, \ldots, p \\ x &\geq 0 \end{aligned} \tag{1}$$

Here A is $m \times n$, $1 \leq p \leq n$, and (1) will also be denoted as $\tilde{A}x \geq \tilde{b}$. Note that the vector $s \in \mathbb{R}^{m+p+n}$ of surplus variables has n components of the form $s_{m+p+j} = x_j$, which represent just a set of different names for the structural variables x_j.

Let x^* be an optimal solution to (LP) and let

$$x_k = \bar{a}_{k0} - \sum_{j \in J} \bar{a}_{kj} s_j \tag{2}$$

be the row of the optimal simplex tableau corresponding to basic variable x_k, with $0 < \bar{a}_{k0} < 1$ and J the index set of nonbasic variables. The *intersection cut* [1] from the convex set $\{x \in \mathbb{R}^n : 0 \leq x_k \leq 1\}$, also known as the *simple disjunctive cut* from the condition $x_k \leq 0 \vee x_k \geq 1$ applied to (2), is $\pi s \geq \pi_0$, where $\pi_0 = \bar{a}_{k0}(1 - \bar{a}_{k0})$ and

$$\pi_j := \max\{\pi_j^1, \pi_j^2\},\ j \in J, \text{ with } \pi_j^1 = \bar{a}_{kj}(1 - \bar{a}_{k0}),\ \pi_j^2 = -\bar{a}_{kj}\bar{a}_{k0}.$$

This cut can be strengthened [1] using the integrality of some variables in J, by replacing π with $\bar{\pi}$, where $\bar{\pi}_j = \pi_j$ for $j \in J \backslash \{1, \ldots, p\}$, and

$$\bar{\pi}_j := \min\{f_{kj}(1 - \bar{a}_{k0}), (1 - f_{kj})\bar{a}_{k0}\}, \quad j \in J \cap \{1, \ldots, p\},$$

with $f_{kj} = \bar{a}_{kj} - \lfloor \bar{a}_{kj} \rfloor$. This *strengthened intersection cut* or *strengthened simple disjunctive cut* is the same as the *mixed integer Gomory (MIG) cut*.

On the other hand, given the same optimal solution x^* to (LP), a *deepest lift-and-project (L&P) cut* $\alpha x \geq \beta$ is obtained by solving a Cut Generating Linear Program [4] in a higher dimensional space:

$$\begin{aligned}
&\min \alpha x^* - \beta \\
&\text{s.t.} \\
&\alpha - u\tilde{A} + u_0 e_k = 0 \\
&\alpha - v\tilde{A} - v_0 e_k = 0 \\
&-\beta + u\tilde{b} = 0 \\
&-\beta + v\tilde{b} + v_0 = 0 \\
&ue + ve + u_0 + v_0 = 1 \\
&u, v, u_0, v_0 \geq 0
\end{aligned} \tag{CGLP}_k$$

where $e = (1, \ldots, 1)$ and e_k is the k-th unit vector.

While an optimal solution to $(\text{CGLP})_k$ yields a "deepest" cut $\alpha x \geq \beta$, i.e. one that cuts off x^* by a maximum amount, any solution to the constraint set of $(\text{CGLP})_k$ yields a member of the family of L&P cuts. If $(\alpha, \beta, u, v, u_0, v_0)$ is a basic solution to $(\text{CGLP})_k$, the coefficients of the corresponding L&P cut are $\beta = u\tilde{b} = v\tilde{b} + v_0$,

$$\alpha_k = \max\{u\tilde{A}_k - u_{m+p+k} - u_0, v\tilde{A}_k - v_{m+p+k} + v_0\},$$

and

$$\alpha_j = \max\{u\tilde{A}_j - u_{m+p+j}, v\tilde{A}_j - v_{m+p+j}\}, \quad j \neq k$$

where $\tilde{A}_j$ is the j-th column of $\tilde{A}$.

Again, this cut can be strengthened using the integrality of some of the structural variables by replacing α with $\bar{\alpha}$, where $\bar{\alpha}_j = \alpha_j$ for $j = k$ and $j \notin \{1, \ldots, p\}$, and

$$\bar{\alpha}_j = \min\{u\tilde{A}_j - u_{m+p+j} + u_0\lceil m_j \rceil, v\tilde{A}_j - v_{m+p+j} - v_0\lfloor m_j \rfloor\}, \quad j \in \{1, \ldots, p\}\backslash\{k\},$$

with

$$m_j = (v\tilde{A}_j - v_{m+p+j} - u\tilde{A}_j + u_{m+p+j})/(u_0 + v_0).$$

In [7] it was shown that the intersection cut obtained from a given component x_k of a basic feasible solution of (LP) is equivalent to the L&P cut obtained from a basic solution to $(\text{CGLP})_k$, where the bases in question are related to each other in a well defined manner. The same relationship holds between the strengthened version of the intersection cut, i.e. the mixed integer Gomory cut, on the one hand, and the strengthened L&P cut on the other. Furthermore, a strengthened L&P cut is equivalent to a MIG cut from some LP tableau that in general is neither optimal nor feasible, and the search for a deepest L&P cut can be viewed as the search for the appropriate simplex tableau from which to derive the corresponding MIG cut. The next section discusses this connection.

3 The correspondence between L&P cuts and MIG cuts

Let $\alpha x \geq \beta$ be a L&P cut corresponding to a basic feasible solution $(\alpha, \beta, u, v, u_0, v_0)$ of $(\text{CGLP})_k$, and let $\bar{\alpha}x \geq \beta$ be its strengthened version. Further, let $u_0 > 0$, $v_0 > 0$ (these are known to be the only solutions yielding cuts that are not combinations of the rows of $\tilde{A}x \geq \tilde{b}$), and let M_1 and M_2 be the index sets of the basic components of u and v, respectively. Then $M_1 \cap M_2 = \phi$, $|M_1 \cup M_2| = n$, and the square submatrix $\hat{A}$ of $\tilde{A}$ whose rows are indexed by $M_1 \cup M_2$ is nonsingular (see [7]). Now define $J := M_1 \cup M_2$. Then letting $\hat{b}$ denote the subvector of $\tilde{b}$ corresponding to $\hat{A}$ and writing s_J for the surplus variables indexed by J, we have

$$\hat{A}x - s_J = \hat{b} \quad \text{or} \quad x = \hat{A}^{-1}\hat{b} - \hat{A}^{-1}s_J \tag{3}$$

and the row of (3) corresponding to x_k (a basic variable, since $k \notin J$) can be written as

$$x_k = \bar{a}_{k0} - \sum_{j \in J} \bar{a}_{kj} s_j, \tag{4}$$

where $\bar{a}_{k0} = e_k\hat{A}^{-1}\hat{b}$ and $\bar{a}_{kj} = -\hat{A}^{-1}_{kj}$. Notice that (4) is the same as (2). Furthermore, it can be shown that $0 < \bar{a}_{k0} < 1$, and we have

Theorem 1 [7] *The MIG cut $\bar{\pi} s \geq \pi_0$ from (4) is equivalent to the strengthened L&P cut $\bar{\alpha} x \geq \beta$.*

Conversely, suppose (4) is the row associated with x_k in a basic solution to (LP), not necessarily optimal or even feasible, such that $0 < \bar{a}_{k0} < 1$. Then we have

Theorem 2 [7] *Let (M_1, M_2) be any partition of J such that $j \in M_1$ if $\bar{a}_{kj} < 0$ and $j \in M_2$ if $\bar{a}_{kj} > 0$. Then the solution to (CGLP)$_k$ corresponding to the basis with components $(\alpha, \beta, u_0, v_0, \{u_i : i \in M_1\}, \{v_i : i \in M_2\})$ defines a L&P cut $\alpha x \geq \beta$ whose strengthened version $\bar{\alpha} x \geq \beta$ is equivalent to the MIG cut $\bar{\pi} s \geq \pi_0$ derived from (4).*

Note that the partition (M_1, M_2) of J, and therefore the basis of (CGLP)$_k$ defined by it, is not unique, since the variables $j \in J$ such that $\bar{a}_{kj} = 0$ can be assigned either to M_1 or to M_2. This means that the correspondence between bases described above maps each basis B of (LP) into a set of bases $\varphi(B)$ of (CGLP)$_k$, where typically $|\varphi(B)| > 1$. However, all bases in $\varphi(B)$ correspond to the same solution of (CGLP)$_k$, i.e. they are degenerate, and the correspondence between basic solutions (as opposed to bases) of (LP) and (CGLP)$_k$ is one to one.

The above correspondence was established in [7] for the (CGLP)$_k$ of Sect. 2, which uses the normalization constraint $ue + ve + u_0 + v_0 = 1$. We now address the question of what happens to the above correspondence if this normalization constraint is replaced by the more general one

$$\sum_{i=1}^{m+p+n} (u_i + v_i)\lambda_i + u_0 + v_0 = \lambda_0 \tag{5}$$

where λ_0 is a positive integer and $\lambda_i \geq 0$, $i = 1, \ldots, m + p + n$. The reason we use the same multiplier-variable λ_i for both u_i and v_i is that $u_i v_i = 0$ for any basic solution to CGLP (see [4]). It turns out that the equivalence stated in Theorems 1 and 2 is not affected by this change, but the scaling factor which establishes the precise correspondence between the cuts does depend on the normalization constraint. This scaling factor becomes important, for instance, when we measure the strength of a cut by the amount the cut is violated by $\bar{x}$. Indeed, the equivalence of Theorems 1 and 2 above is derived from the relations

$$\begin{array}{ll} \theta\alpha = \pi\hat{A}, & \theta\beta = \pi_0 + \pi\hat{b} \\ \theta u_J = \pi - \pi^1, & \theta v_J = \pi - \pi^2 \\ \theta u_0 = 1 - \bar{a}_{k0}, & \theta v_0 = \bar{a}_{k0} \end{array} \tag{6}$$

proved in [7] to hold for some scalar $\theta > 0$. In order to turn the equivalence into an exact correspondence, the scalar θ has to be chosen such that $u_{M_1}\mathbf{1}_{M_1} + v_{M_2}\mathbf{1}_{M_2} + u_0 + v_0 = 1$. Thus when the normalization constraint (5) is used, (6) and with it the equivalence between the cuts still holds, but in order to get the exact correspondence, the scaling factor has to be chosen such that

$$u_{M_1}\lambda_{M_1} + v_{M_2}\lambda_{M_2} + u_0 + v_0 = \lambda_0. \tag{7}$$

4 The lift-and-project procedure in the original LP tableau

The lift-and-project procedure in the (LP) tableau uses the above correspondence to mimic the optimization of $(\text{CGLP})_k$ by the simplex algorithm. Consider the row corresponding to x_k of the form (2) which we call the source row. At each iteration of the procedure, we perform a pivot in a row $i \neq k$, which brings about a linear combination of the source row with row i

$$x_k + \gamma x_i = \overline{a}_{k0} + \gamma \overline{a}_{i0} - \sum_{j \in J} (\overline{a}_{kj} + \gamma \overline{a}_{ij}) s_j \tag{8}$$

such that the intersection cut obtained from this new row is more violated by x^* than the one obtained from the source row. Here $\gamma = \gamma_j = -\overline{a}_{kj}/\overline{a}_{ij}$, where $j \in J$ is the pivot column. This combination (the choice of the row i and of γ), is guided by the correspondence with $(\text{CGLP})_k$. Each row i of the (LP) simplex tableau corresponds to a pair of columns of $(\text{CGLP})_k$ with associated nonbasic variables u_i, v_i.

The first main step in the procedure is to compute the reduced costs r_{u_i} and r_{v_i} in $(\text{CGLP})_k$ for all $i \notin J \cup \{k\}$. As shown in [7], these reduced costs can be expressed in terms of the entries $\bar{a}_{ij}$ of the (LP) tableau and the solution x^*. We use these expressions in our computations. If there is no negative reduced cost, the current basis is optimal for $(\text{CGLP})_k$ and the optimal strengthened lift-and-project cut is obtained as the MIG cut from the source row of (LP) (using the correspondence of Theorem 2).

On the other hand, if at least one negative reduced cost exists, then the cut can be improved by performing a pivot in $(\text{CGLP})_k$ where the corresponding variable u_i or v_i enters the basis. In the (LP) tableau, this negative reduced cost (r_{u_i} or r_{v_i}) corresponds to a basic variable x_i which has to leave the basis. Choosing the variable x_j to enter the basis is the second main step of the procedure. In [7], two evaluation functions $f^+(\gamma)$ [resp. $f^-(\gamma)$] were defined, which represent the objective function value of $(\text{CGLP})_k$, i.e. the violation of the cut resulting from the combination of row k and row i for positive, respectively, negative values of γ. These two functions are minimized to select the variable to enter the basis which leads to the largest improvement in cut violation among all variables that can replace the exiting variable. Once the exiting and entering variables have been selected, the pivot in (LP) is performed and the procedure is iterated from the new basis until $(\text{CGLP})_k$ is optimized. The pseudo-code of Fig. 1 describes this procedure.

As shown in [5], the lift-and-project cuts are more efficiently generated in a subspace where all the non-basic structural variables of (LP) are fixed to their values in the optimal solution. Performing the separation in the subspace while working in the (LP) tableau is done simply by removing all the structural nonbasic variables from it before starting the pivoting procedure. At the end of the procedure a lifting step is performed to obtain a valid cut for the original problem by recomputing the source row in the full space and generating the corresponding MIG cut.

5 Computation of the reduced costs and of the evaluation functions

A key point for efficiently implementing the lift-and-project procedure is the computation of the reduced costs and the evaluation functions.

Let x^* be the optimal solution to (LP).
Let $k \in \{1, \ldots, p\}$with x^*_k fractional.
Let I and J be the index sets of basic and non-basic variables in an optimal basis of (LP).
Let $\overline{A}$ be the optimal tableau.
Let *num_pivots*:= 0.
while *num_pivots* < *pivot_limit* **do**
  Compute the reduced costs r_{u_i}, r_{v_i} for each $i \notin J \cup \{k\}$
  if There exists i such that $r_{u_i} < 0 \vee r_{v_i} < 0$
    then
      Let $\hat{\imath} := \arg \min_{i \notin J \cup \{k\}} \min\{r_{u_i}, r_{v_i}\}$,
      Let $J' = \{j \in J : |\overline{a}_{\hat{\imath}j}| \geq \epsilon\}$ be the set of admissible pivots.
      Let $J^+ = J' \cap \{j \in J : \gamma_j = -\overline{a}_{kj}/\overline{a}_{\hat{\imath}j} > 0\}$.
      Let $\hat{\jmath} := \arg\min\{\arg\min_{j \in J^+} f^+(\gamma_j), \arg\min_{j \in J' \setminus J^+} f^-(\gamma_j)\}$.
      Perform a pivot in (LP) by pivoting out $\hat{\imath}$ and pivoting in $\hat{\jmath}$.
      Let $I := I \cup \{\hat{\jmath}\} \setminus \{\hat{\imath}\}$.
      Let $\overline{A}$ be the updated tableau in the new basis.
      Let *num_pivots*+= 1.
    else /* *cut is optimal.* */
      Generate the MIG cut from row k of the current tableau.
      exit
  fi
od

Fig. 1 Lift-and-project procedure

The expression for the reduced costs in case of normalization constraint (5) is somewhat different than in the standard case, so Theorem 9 of [7] needs to be replaced by the following.

Theorem 3 *When using the normalization constraint* (5)*, the expressions for the reduced costs are*

$$\begin{aligned} r_{u_i} &= \sigma \left(-\sum_{j \in M_1} \bar{a}_{ij}\lambda_j + \sum_{j \in M_2} \bar{a}_{ij}\lambda_j - \lambda_i \right) - \sum_{j \in M_2} \bar{a}_{ij}\bar{s}_j + \bar{a}_{i0}(1 - \bar{x}_k) \\ r_{v_i} &= \sigma \left(\sum_{j \in M_1} \bar{a}_{ij}\lambda_j - \sum_{j \in M_2} \bar{a}_{ij}\lambda_j - \lambda_i \right) - \sum_{j \in M_1} \bar{a}_{ij}\bar{s}_j + \bar{a}_{i0}\bar{x}_k \end{aligned} \tag{9}$$

where

$$\sigma = \frac{\sum_{j \in M_2} \bar{a}_{kj}\bar{s}_j - \bar{a}_{k0}(1 - \bar{a}_k)}{1 + \sum_{j \in J} |\bar{a}_{kj}|\lambda_j}$$

Proof (Parallels the proof of Theorem 9 of [7].) By eliminating α, β and the nonbasic u_h, v_h for $h \neq i$ from the constraint set of $(\text{CGLP})_k$, we obtain

$$\begin{aligned} u_j &= -(u_0 + v_0)\bar{a}_{kj} + (u_i - v_i)\bar{a}_{ij} \qquad \text{for } j \in M_1 \\ v_j &= (u_0 + v_0)\bar{a}_{kj} - (u_i - v_i)\bar{a}_{ij} \qquad \text{for } j \in M_2 \\ v_0 &= (u_0 + v_0)\bar{a}_{k0} - (u_i - v_i)\bar{a}_{i0} \end{aligned} \tag{10}$$

The normalization constraint (5) can be written in terms of the current solution (in which $u_0, v_0 > 0$ and $u_i = 0, i \in M_2$, $v_i = 0, i \in M_1$) as

$$\lambda_0 = \sum_{j \in M_1} u_j \lambda_j + \sum_{j \in M_2} v_j \lambda_j + (u_i + v_i)\lambda_i + u_0 + v_0$$

or, after substituting for u_j and v_j from (10),

$$\begin{aligned} \lambda_0 = (u_0 + v_0) &\left(-\sum\nolimits_{j \in M_1} \bar{a}_{kj}\lambda_j + \sum\nolimits_{j \in M_2} \bar{a}_{kj}\lambda_j + 1 \right) \\ &+ (u_i - v_i) \left(\sum\nolimits_{j \in M_1} \bar{a}_{ij}\lambda_j - \sum\nolimits_{j \in M_2} \bar{a}_{ij}\lambda_j \right) \\ &+ u_i\lambda_i + v_i\lambda_i . \end{aligned}$$

Since (10) is satisfied for the current basic solution with $u_i = v_i = 0$ and since $u_{M_1}, v_{M_2} \geq 0$, it follows that $\bar{a}_{kj} \leq 0$ for $j \in M_1$ and $\bar{a}_{kj} \geq 0$ for $j \in M_2$, so

$$-\sum_{j \in M_1} \bar{a}_{kj}\lambda_j + \sum_{j \in M_2} \bar{a}_{kj}\lambda_j = \sum_{j \in J} |\bar{a}_{kj}|\lambda_j ,$$

and thus we have

$$u_0 + v_0 = \frac{\lambda_0 - (u_i - v_i)\left(\sum_{j \in M_1} \bar{a}_{ij}\lambda_j - \sum_{j \in M_2} \bar{a}_{ij}\lambda_j \right) - (u_i + v_i)\lambda_i}{1 + \sum_{j \in J} |\bar{a}_{kj}|\lambda_j} \tag{11}$$

As in [7], we can now write the objective function of $(\text{CGLP})_k$ in terms of u_i and v_i as

$$\begin{aligned} \alpha\bar{x} - \beta = (u_0 + v_0) &\left(\sum_{j \in M_2} \bar{a}_{kj}\bar{s}_j - \bar{a}_{k0}(1 - \bar{x}_k) \right) \\ &+ (u_i - v_i) \left(-\sum_{j \in M_2} \bar{a}_{ij}\bar{s}_j + \bar{a}_{i0}(1 - \bar{x}_k) \right) + v_i\bar{s}_i . \end{aligned}$$

If we substitute for $u_0 + v_0$ from (11), we have

$$\alpha\bar{x} - \beta = \frac{\lambda_0 - (u_i - v_i)\left(\sum_{j\in M_1}\bar{a}_{ij}\lambda_j - \sum_{j\in M_2}\bar{a}_{ij}\lambda_j\right) - (u_i + v_i)\lambda_i}{1 + \sum_{j\in J}|\bar{a}_{kj}|\lambda_j}$$
$$\times\left(\sum_{j\in M_2}\bar{a}_{kj}\bar{s}_j - \bar{a}_{k0}(1-\bar{x}_k)\right) + (u_i - v_i)$$
$$\times\left(-\sum_{j\in M_2}\bar{a}_{ij}\bar{s}_j + \bar{a}_{i0}(1-\bar{x}_k)\right) + v_i\bar{s}_i,$$

or, defining σ as in the Theorem,

$$a\bar{x} - \beta = \sigma\lambda_0 + u_i\left(-\sigma\sum_{j\in M_1}\bar{a}_{ij}\lambda_j + \sigma\sum_{j\in M_2}\bar{a}_{ij}\lambda_j - \sigma\lambda_i - \sum_{j\in M_2}\bar{a}_{ij}\bar{s}_j + \bar{a}_{i0}(1-\bar{x}_k)\right)$$
$$+v_i\left(\sigma\sum_{j\in M_1}\bar{a}_{ij}\lambda_j - \sigma\sum_{j\in M_2}\bar{a}_{ij}\lambda_j - \sigma\lambda_i + \sum_{j\in M_2}\bar{a}_{ij}\bar{s}_j - \bar{a}_{i0}(1-\bar{x}_k) + \bar{s}_i\right).$$

Substituting for $\bar{s}_i$ from $\bar{s}_i = \bar{a}_{i0} - \sum_{j\in M_1}\bar{a}_{ij}\bar{s}_j - \sum_{j\in M_2}\bar{a}_{ij}\bar{s}_j$ replaces the expression in parentheses following v_i by

$$\sigma\sum_{j\in M_1}\bar{a}_{ij}\lambda_j - \sigma\sum_{j\in M_2}\bar{a}_{ij}\lambda_j - \sigma\lambda_i - \sum_{j\in M_1}\bar{a}_{ij}\bar{s}_j + \bar{a}_{i0}\bar{x}_k$$

and thus the coefficients of u_i and v_i in the above expression for $\alpha\bar{x} - \beta$, i.e. their reduced costs, become those shown in the Theorem.

As shown in [14], for a given partition (M_1, M_2) (as defined in Sect. 3) the expressions for the reduced costs depend only linearly on the coefficients of the tableau, and therefore the reduced costs of all non-basic variables in $(\text{CGLP})_k$ can be computed by doing only one multiplication with the basis inverse.

A critical element in computing the reduced costs is the choice of the partition (M_1, M_2). If for all $j \in J$, $\bar{a}_{kj}$ is non-zero, this partition is uniquely defined; but if this is not the case, several partitions can be chosen. The rule given in [7] is to take $M_1 = \{j \in J : \bar{a}_{kj} < 0 \ \wedge\ (\bar{a}_{kj} = 0 \ \wedge\ \bar{a}_{ij} > 0)\}$ (and $M_2 = J\backslash M_1$) for computing r_{u_i}, and $M_1 = \{j \in J : \bar{a}_{kj} < 0 \ \wedge\ (\bar{a}_{kj} = 0 \wedge \bar{a}_{ij} < 0)\}$ for computing r_{v_i}. This rule has the advantage that if a negative reduced cost is found, then the corresponding pivot leads to a strictly better cut. On the other hand, to determine this partition, one has to compute the coefficients $\bar{a}_{ij}$ for all j such that $\bar{a}_{kj} = 0$ and all i. Therefore we use another rule. Namely, following [14], we randomly assign all the zero elements of the source row to either M_1 or M_2. This rule has the disadvantage that although the reduced

cost for the perturbed row is negative, it may happen that all the pivots with the corresponding variable u_i or v_i entering the basis are degenerate in $(\text{CGLP})_k$. Nevertheless, in our experiments (not reported here), this rule had a clear computational advantage.

The second main step of the procedure is the computation of the evaluation functions $f^+(\gamma)$ and $f^-(\gamma)$. Again the expressions for these evaluation functions are slightly different when using the normalization constraint (5), and thus Theorem 10 of [7] needs to be modified as follows. In the next Theorem, if $\bar{a}_{i0} = 0$ then $(1 - \bar{a}_{k0}/\bar{a}_{i0}) = \infty$ and $-\bar{a}_{kj}/\bar{a}_{i0} = -\infty$.

Theorem 4 *When using the normalization constraint (5), the $(\text{CGLP})_k$ objective function value resulting from the pivot sequence corresponding to the pivot in row i and column ℓ of the (LP) tableau is given by $\gamma = -\bar{a}_{k\ell}/\bar{a}_{i\ell}$ and*

$$f^+(\gamma) := \frac{\left(\sum_{j\in J}(-(\bar{a}_{k0} + \gamma\bar{a}_{i0})\bar{a}_{kj} + \max\{\bar{a}_{kj}, -\gamma\bar{a}_{ij}\})\bar{x}_j - (1 - \bar{a}_{k0} - \gamma\bar{a}_{i0})\bar{a}_{k0}\right)\lambda_0}{1 + |\gamma|\lambda_i + \sum_{j\in J}|\bar{a}_{kj} + \gamma\bar{a}_{ij}|\lambda_j} \tag{12}$$

if $0 \le \gamma < (1 - \bar{a}_{k0})/\bar{a}_{i0}$, and by

$$f^-(\gamma) := \frac{\left(\sum_{j\in J}(-(\bar{a}_{k0}+\gamma\bar{a}_{i0})\bar{a}_{kj} + \max\{\bar{a}_{kj} + \gamma\bar{a}_{ij}, 0\})\bar{x}_j - (1-\bar{a}_{k0})(\bar{a}_{k0} + \gamma\bar{a}_{i0})\right)\lambda_0}{1 + |\gamma|\lambda_i + \sum_{j\in J}|\bar{a}_{kj} + \gamma\bar{a}_{ij}|\lambda_j} \tag{13}$$

if $-\bar{a}_{k0}/\bar{a}_{i0} < \gamma \le 0$.

Proof (Parallels the proof of Theorem 10 in [7]). Adding row i with weight $\gamma \in \mathbb{R}$ to the source row k of the LP tableau we obtain the composite row

$$x_k + \gamma x_i + \sum_{j\in J}(\bar{a}_{kj} + \gamma\bar{a}_{ij})s_j = \bar{a}_{k0} + \gamma\bar{a}_{i0} \tag{14}$$

from which we can derive a simple disjunctive cut $\pi^\gamma s_J \ge \pi_0^\gamma$ if $0 < \bar{a}_{k0} + \gamma\bar{a}_{i0} < 1$, i.e. if $\frac{-\bar{a}_{k0}}{\bar{a}_{i0}} < \gamma < \frac{1-\bar{a}_{k0}}{\bar{a}_{i0}}$.

For any γ in the above interval, this cut has coefficients

$$\begin{aligned}
\pi_i^\gamma &= \max\{(1 - \bar{a}_{k0} - \gamma\bar{a}_{i0})\gamma, -(\bar{a}_{k0} + \gamma\bar{a}_{i0})\gamma\} \\
\pi_j^\gamma &= \max\{(1 - \bar{a}_{k0} - \gamma\bar{a}_{i0})(\bar{a}_{kj} + \gamma\bar{a}_{ij}), -(\bar{a}_{k0} + \gamma\bar{a}_{i0})(\bar{a}_{kj} + \gamma\bar{a}_{ij})\} \qquad \text{for } j \in J \\
\pi_0^\gamma &= (1 - \bar{a}_{k0} - \gamma\bar{a}_{i0})(\bar{a}_{k0} + \gamma\bar{a}_{i0}).
\end{aligned}$$

The $(\text{CGLP})_k$ solution corresponding to this cut is given by the relations (6), where π^1 and π^2 become

$$\pi_i^{\gamma 1} = (1 - \bar{a}_{k0} - \gamma \bar{a}_{i0})\gamma, \quad \pi_j^{\gamma 1} = (1 - \bar{a}_{k0} - \gamma \bar{a}_{i0})(\bar{a}_{kj} + \gamma \bar{a}_{ij})$$
$$\pi_i^{\gamma 2} = -(\bar{a}_{k0} + \gamma \bar{a}_{i0})\gamma, \quad \pi_j^{\gamma 2} = -(\bar{a}_{k0} + \gamma \bar{a}_{i0})(\bar{a}_{kj} + \gamma \bar{a}_{ij}) \quad \text{for } j \in J.$$

Thus from (6) it follows that the components of (u, v, u_0, v_0) satisfy

$$u_i + v_i = |\pi_i^{\gamma 1} - \pi_i^{\gamma 2}| = |\gamma|$$
$$u_j + v_j = |\pi_j^{\gamma 1} - \pi_j^{\gamma 2}| = |\bar{a}_{kj} + \gamma \bar{a}_{ij}| \quad \text{for } j \in J$$
$$v_0 + v_0 = (1 - \bar{a}_{k0} - \gamma \bar{a}_{i0}) + (\bar{a}_{k0} + \gamma \bar{a}_{i0}) = 1.$$

This solution satisfies all constraints of $(\text{CGLP})_k$ except for the normalization constraint (5). To satisfy the latter, we have to scale the cut by

$$(u\lambda + v\lambda + u_0 + v_0)/\lambda_0,$$

which amounts to

$$\left((u_i + v_i)\lambda_i + \sum_{j \in J} (u_j + v_j)\lambda_j + u_0 + v_0 \right) \Big/ \lambda_0$$
$$= \left(1 + |\gamma|\lambda_i + \sum_{j \in J} |\bar{a}_{kj} + \gamma \bar{a}_{ij}|\lambda_j \right) \Big/ \lambda_0.$$

The objective function of $(\text{CGLP})_k$ corresponding to this cut is thus obtained by scaling the function $\pi_i^\gamma \bar{x}_i + \pi^\gamma \bar{s}_J - \pi_0^\gamma$ by the above scalar value, i.e.

$$\frac{(\pi_i^\gamma \bar{x}_i + \pi^\gamma \bar{s}_j - \pi_0^\gamma)\lambda_0}{1 + |\gamma|\lambda_i + \sum_{j \in J} |\bar{a}_{kj} + \gamma \bar{a}_{ij}|\lambda_j}$$

The expressions for $(\pi^\gamma, \pi_0^\gamma)$ involve terms with γ^2. As shown in [7], such terms can be eliminated by subtracting π_i^γ times row i from the cut $\pi_i^\gamma x_i + \pi^\gamma s_J \geq \pi_0^\gamma$. The result depends on the sign of γ, and thus we have for $\gamma > 0$

$$\pi_j^{\gamma +} = -(\bar{a}_{k0} + \gamma \bar{a}_{i0})\bar{a}_{kj} + \max\{\bar{a}_{kj}, -\gamma \bar{a}_{ij}\},$$
$$\pi_0^{\gamma +} = (1 - \bar{a}_{k0} - \gamma \bar{a}_{i0})\bar{a}_{k0}$$

and for $\gamma < 0$

$$\pi_j^{\gamma -} = -(\bar{a}_{k0} + \gamma \bar{a}_{i0})\bar{a}_{kj} + \max\{\bar{a}_{kj} + \gamma \bar{a}_{ij}, 0\}$$
$$\pi_0^{\gamma -} = (1 - \bar{a}_{k0})(\bar{a}_{k0} + \gamma \bar{a}_{i0}).$$

Using $\pi^{\gamma+}\bar{s}_J - \pi_0^{\gamma+}$ and $\pi^{\gamma-}\bar{s}_j - \pi_0^{\gamma-}$ in place of $\pi_i^{\gamma}\bar{x}_i + \pi^{\gamma}\bar{s}_J - \pi_0^{\gamma}$, the objective function becomes

$$\frac{(\pi^{\gamma+}\bar{s}_J - \pi_0^{\gamma+})\gamma_0}{1 + |\gamma|\lambda_i + \sum_{j\in J} |\bar{a}_{kj} + \gamma\bar{a}_{ij}|\lambda_j} \qquad \text{for } \gamma > 0$$

and

$$\frac{(\pi^{\gamma-}\bar{s}_J - \pi_0^{\gamma-})\gamma_0}{1 + |\gamma|\lambda_i + \sum_{j\in J} |\bar{a}_{kj} + \gamma\bar{a}_{ij}|\lambda_j} \qquad \text{for } \gamma < 0.$$

Finally, substituting the corresponding expressions for $\pi^{\gamma+}, \pi_0^{\gamma+}, \pi^{\gamma-}, \pi_0^{\gamma-}$, we obtain $f^+(\gamma)$ and $f^-(\gamma)$, respectively, as in the Theorem.

As shown in [14], the evaluation functions are unimodal piecewise continuously differentiable and their minimum can be found efficiently, once rows k and i of the tableau are specified. It is easy to see that for each $j \in J$, the function $f^+(\gamma)$ or $f^-(\gamma)$ has a breakpoint at the value γ_j of γ for which $\bar{a}_{kj} + \gamma\bar{a}_{ij} = 0$. If $\gamma_j > 0$, this is a breakpoint of $f^+(\gamma)$; and if $\gamma_j < 0$, it is a breakpoint of $f^-(\gamma)$. Thus $f^+(\gamma)$ and $f^-(\gamma)$ together have $|J|$ breakpoints.

6 The correspondence between pivots in the LP and the CGLP tableaus

In Sect. 3 we described the correspondence between L&P cuts and MIG cuts, which is based on a one-to-many correspondence between the bases of the LP and those of the CGLP. We recall that this correspondence maps a basis B of the LP into a collection of bases $\varphi(B)$ of the CGLP, where typically $|\varphi(B)| > 1$ due to degeneracy of the CGLP. Thus, despite the potential multiplicity of CGLP bases corresponding to an LP basis, the correspondence between basic *solutions* to the LP and to the CGLP is one-to-one.

While the correspondence between the bases of the LP and those of the CGLP should be clear from Sect. 3 (Theorem 2; see [7] for details), the correspondence between pivots in the LP and in the CGLP does not immediately follow from this and is not treated in any detail in [7]. In this section we examine the nature of this correspondence and illustrate it on an example. The central fact of this correspondence is that: a *single pivot* in the LP corresponds to a *sequence of (mostly nondegenerate) pivots* in the CGLP. Furthermore, the length of this sequence can be quite significant, bounded only by $|J|$, and it depends on the value of γ for which $f^+(\gamma)$ or $f^-(\gamma)$ attains it minimum. Suppose, for the sake of simplicity, that $\min\{f^+(\gamma), f^-(\gamma)\}$ is attained by $f^+(\gamma)$ for some value $\gamma_{j_t} > 0$ of γ. (The situation is analogous if the minimum is attained by $f^-(\gamma)$.) This means that as γ increases from 0, $f^+(\gamma)$ decreases from $f^+(0)$ until it reaches the value of $f^+(\gamma_{j_t})$, after which it increases.

Let

$$x_k + \gamma x_i + \sum \left((\bar{a}_{kj} + \gamma\bar{a}_{ij})x_j : j \in J\right) = \bar{a}_{k0} + \gamma\bar{a}_{i0} \qquad (6.1)_\gamma$$

be the combined source row resulting from a pivot in row i, where γ is the generic notation for $\gamma_1 = -\frac{\bar{a}_{k\ell}}{\bar{a}_{i\ell}}$ for some $\ell \in J$. Let $\gamma_{j_1}, \ldots, \gamma_{j_t}$ be the sequence of non-decreasing values of γ for which some coefficient $\bar{a}_{kj} + \gamma \bar{a}_{ij}$ of $(6.1)_\gamma$ becomes 0, and let $f^+(\gamma)$ attain its minimum for $\gamma = \gamma_{j_t}$.

Theorem 5 *If $(6.1)_{\gamma_{j_t}}$ is the combined source row resulting from a pivot on $\bar{a}_{ij_t}$ and $\pi x \geq \pi_0$ is the cut from the disjunction $x_k + \gamma_{j_t} x_i \leq 0 \ \vee \ x_k + \gamma_{j_t} x_i \geq 1$, then the equivalent L&P cut $\alpha x \geq \beta$ corresponds to a basic feasible solution to $(CGLP)_k$ resulting from a sequence of t pivots defined as follows.*

(*a*) *The first pivot introduces into the basis the nonbasic variable u_i or v_i (the one with negative reduced cost) corresponding to the x_i row of the LP tableau, and removes from the basis the variable u_{j_1} or v_{j_1} corresponding to column j_1 of the LP tableau*

(*b*) *Each pivot except for the first and the last one, exchanges*

(*b1*) *a nonbasic variable u_{j_h} for a basic variable $u_{j_{h+1}}$ if $\bar{a}_{kj_h}$ switches its sign from positive to negative when γ is increased beyond γ_{j_h} for $h = 2, \ldots, t-1$; or*

(*b2*) *a nonbasic variable v_{j_h} for a basic variable $v_{j_{h+1}}$ if $\bar{a}_{kj_h}$ switches its sign from negative to positive when γ is increased beyond γ_{j_h} ($h = 2, \ldots, t-1$).*

(*c*) *The last pivot exchanges the nonbasic variable $u_{j_{t-1}}$ (or $v_{j_{t-1}}$) with the basic variable u_{j_t} (or v_{j_t}) corresponding to column j_t of the LP tableau.*

Throughout the sequence, the pivot corresponding to the transition from γ_{j_h} to $\gamma_{j_{h+1}}$ is nondegenerate if and only if $\gamma_{j_h} \neq \gamma_{j_{h+1}}$.

Proof Let (M_1, M_2) be a valid partition of J, the nonbasic set of the starting LP tableau, and let (M_1', M_2') be a valid partition of the nonbasic set $J' := (J \cup \{i\}) \backslash \{j_t\}$ of the LP tableau resulting from the pivot on $\bar{a}_{ij_t}$. Whenever (M_1', M_2') differs from (M_1, M_2) in more than one element, the two associated solutions to $(CGLP)_k$ are not adjacent. Thus executing the single pivot in $(CGLP)_k$ that corresponds to the pivot on $\bar{a}_{ij_t}$, i.e. exchanging the nonbasic variable u_i (or v_i) for the basic variable u_{j_t} (or v_{j_t}) would yield an infeasible CGLP solution that is not associated with any valid cut. To obtain a basic *feasible* solution to $(CGLP)_k$ one needs to perform a sequence of primal feasible pivots. This sequence is specified by the sequence of sign changes of coefficients of the combined row $(6.1)_\gamma$ as a result of the increase of γ from 0 to γ_{j_t}. Each sign change corresponds to moving the corresponding index from M_1 to M_2 or vice versa, which in turn corresponds to replacing a basic u_j with the corresponding v_j or vice versa. However, since pivots only allow for exchanging two u_j variables among themselves or two v_j variables among themselves, replacing a certain basic u_{j_*} with v_{j_*} takes two pivots, not one: first u_{j_*} is pivoted out (in exchange for some nonbasic u_h), then v_{j_*} is pivoted in (in exchange for some basic v_ℓ). The result is the sequence of pivots described in the Theorem. To see why the pivots in the sequence are nondegenerate if and only if $\gamma_{j_h} \neq \gamma_{j_{h+1}}$, note that each pivot in the sequence corresponds to a different value of $f^+(\gamma)$, i.e. causes a different amount of change in the CGLP objective, if and only if this condition holds.

Next we illustrate the correspondence between pivots in the two tableaux on a small example, a set covering instance with 9 variables and 13 inequalities, obtained from the well known instance `stein9` by adding the inequality $\sum x_j \geq 4$.

The structural variables are $x_1, \ldots, x_9$, the surplus variables associated with the inequalities are $s_1, \ldots, s_{13}$ while those associated with the upper and lower bounds on the structurals are $s_{14}, \ldots, s_{22}$ and $s_{23}, \ldots, s_{31}$, respectively. The variables of $(\text{CGLP})_3$, after eliminating α and β, are u_0, v_0 and u_i, v_i, for $i = 1, \ldots, 31$, i.e. each pair u_i, v_i gets the same subscript as the surplus variable s_i of the corresponding inequality of $\tilde{A}x \geq \tilde{b}$. The optimal solution to the LP relaxation for the objective function we chose ($\min \sum j x_j$) is

$$\bar{x} = \left(1, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0, 0\right).$$

We choose as source row for a cut the one corresponding to basic variable x_3:

$$x_3 + \frac{1}{3}s_4 + \frac{1}{3}s_5 + \frac{1}{3}s_6 + \frac{2}{3}s_8 - \frac{1}{3}s_9 - \frac{2}{3}s_{13} - \frac{2}{3}s_{14} + \frac{2}{3}s_{30} - \frac{1}{3}s_{31} = \frac{2}{3}$$

The intersection cut from $0 \leq x_3 \leq 1$, if generated, would be

$$\frac{1}{9}s_4 + \frac{1}{9}s_5 + \frac{1}{9}s_6 + \frac{2}{9}s_8 + \frac{2}{9}s_9 + \frac{4}{9}s_{13} + \frac{4}{9}s_{14} + \frac{2}{9}s_{30} + \frac{2}{9}s_{31} \geq \frac{2}{9}$$

This cut would have to be scaled in order to obtain the corresponding lift-and-project cut (to satisfy the normalization constraint, in this case $\sum u_i + u_0 + \sum v_i + v_0 = 1$) by a factor of $\frac{3}{16}$, which would yield a violation (i.e. an objective function value of $(\text{CGLP})_3$) of $-\left(\frac{2}{9} * \frac{3}{16}\right) = -0.04167$.

Since there are no 0 coefficients in the source row, J has the unique valid partition (M_1, M_2), with

$$M_1 = \{9, 13, 14, 31\} \quad \text{and} \quad M_2 = \{4, 5, 6, 8, 30\},$$

and the corresponding basis of $(\text{CGLP})_3$ consists, besides u_0 and v_0, of the variables u_9, u_{13}, u_{14}, u_{31}, and v_4, v_5, v_6, v_8 and v_{30}. Thus the feasible solution to $(\text{CGLP})_3$ corresponding to the partition of J dictated by the coefficients of the x_3 row is *non-degenerate*, i.e. the associated basis is *unique*.

Computing the reduced costs r_{u_i}, r_{v_i} of the variables u_i, v_i of $(\text{CGLP})_3$ associated with the rows of the LP simplex tableau, we find that the variable u_{12} has a negative reduced cost of -0.25. Thus we perform a pivot in the corresponding row of LP,

$$s_{12} - 0s_4 - 0s_5 + 1s_6 + 1s_8 - 0s_9 - 1s_{13} - 1s_{14} + 1s_{30} - 2s_{31} = 0.$$

The variable s_{12} leaves the basis. To choose the entering variable, i.e. the pivot column, we compute the functions $f_{12}(\gamma)$ whose value represents the amount by which $\bar{x}$ violates the intersection cut generated from the source row (the row of x_3) after the pivot,

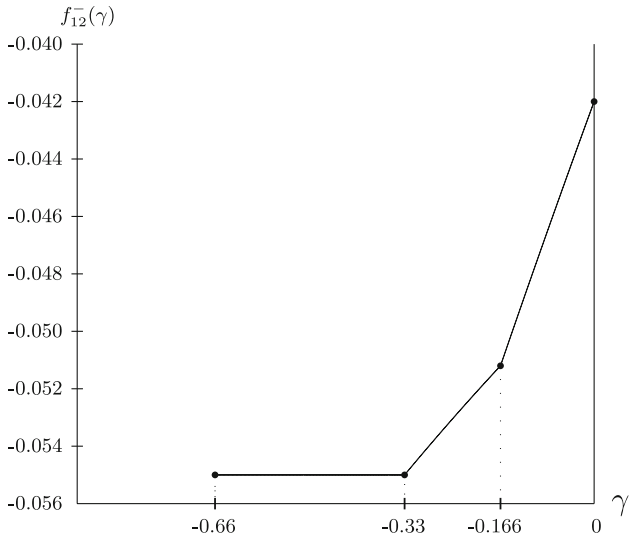

Fig. 2 Plot of the evaluation function $f_{12}^-(\gamma)$

i.e. after adding to the source row γ_j times the row of s_{12}, where $\gamma_j = -\bar{a}_{3j}/\bar{a}_{12,j}$. Depending on the column where we choose to pivot, γ_j may be positive or negative, and in general the evaluation function f is calculated for both cases. In this instance, however, the ratio γ_j happens to be negative for all j with $\bar{a}_{12,j} \neq 0$. Figure 2 plots the function $f_{12}^-(\gamma_j)$, which has 3 breakpoints, namely at $\gamma_{31} = -\frac{1}{6}$, at $\gamma_6 = -\frac{1}{3}$, and at $\gamma_8 = \gamma_{13} = \gamma_{14} = \gamma_{30} = -\frac{2}{3}$, with

$$f_{12}\left(-\frac{1}{6}\right) = -0.05128$$
$$f_{12}\left(-\frac{1}{3}\right) = -0.05556$$
$$f_{12}\left(-\frac{2}{3}\right) = -0.05556$$

We choose s_{14} as the entering variable and perform the pivot; s_{12} leaves and s_{14} enters the basis. As a result, $-\frac{2}{3}$ times the s_{12} row is added to the x_3 row, which becomes

$$x_3 + \frac{1}{3}s_4 + \frac{1}{3}s_5 - \frac{1}{3}s_6 + 0s_8 - \frac{1}{3}s_9 - \frac{2}{3}s_{12} + 0s_{13} + 0s_{30} + 1s_{31} = \frac{2}{3}$$

As a result of the pivot, besides the coefficient of s_{14} (which entered the basis), those of several nonbasic variables (s_8, s_{13}, s_{30}) have become 0. If we were to generate the intersection cut from $0 \leq x_3 \leq 1$ applied to the new x_3 row, it would be

$$\frac{1}{9}s_4 + \frac{1}{9}s_5 + \frac{2}{9}xs_6 + 0x_8 + \frac{2}{9}s_9 + \frac{4}{9}s_{12} + 0s_{13} + 0s_{30} + \frac{1}{3}s_{31} \geq \frac{2}{9},$$

which cuts off $\bar{x}$ by a larger amount than the earlier cut.

Table 1 $(\text{CGLP})_3$ pivots corresponding to the pivot in LP

Pivot #	Entering	Leaving
1	u_{12}	u_{31}
2	v_{31}	v_6
3	u_6	u_{14}

Turning now to what corresponds in $(\text{CGLP})_3$ to the new LP basis resulting from the pivot, we see that the new set $J' = (J \backslash \{14\}) \cup \{12\}$ can be partitioned into (M_1', M_2') in several ways because of the presence of several 0 coefficients. If we adopt the viewpoint that we leave (M_1', M_2') as close as possible to (M_1, M_2), i.e. move only those variables whose coefficients have changed sign, we obtain

$$M_1' = \{6, 9, 12, 13\} \quad \text{and} \quad M_2' = \{4, 5, 8, 30, 31\},$$

a partition that differs from (M_1, M_2) in several, not just one index. Thus, besides the variable u_{12} becoming basic in place of u_{14}, we also see that u_{31} and v_6 have left the basis, while u_6 and v_{31} have entered it. Thus the (feasible) basis of $(\text{CGLP})_3$ corresponding to the new (infeasible) LP basis differs from the earlier one in more than one component, i.e. is nonadjacent to the latter, and can be obtained from it only through a "block pivot," or sequence of pivots, that performs the above exchanges, as shown in Table 1.

All three pivots are nondegenerate, i.e. involve actual changes in the associated solutions to $(\text{CGLP})_3$. In addition, several degenerate pivots could be performed, corresponding to switching between components of the source row in LP (u_{13}, v_8, v_{30}).

Details of the computation, like the successive LP tableaus, can be found under [9].

In this example the pivot sequence in CGLP corresponding to a single pivot in LP was rather short. In larger problems the sequence may consist of hundreds and even thousands of pivots. For example, instance `air04` has 8,904 variables (all 0–1) and 823 constraints. Solving the linear programming relaxation and choosing as source row the one corresponding to basic variable x_{895}, 20 pivots are performed in the LP tableau (in the subspace of fractional variables) to push the cut violation from -0.00009 to -0.00132. The first pivot in LP gives rise to 194 pivots in $(\text{CGLP})_{895}$. Figure 3 plots the evaluation function f_{895} for this first pivot. The minimum is attained for $\gamma = 0.0706161$ at $f^+(\gamma) = -0.00024$. The 20 LP pivots correspond to a total of 1,397 CGLP pivots.

Before discussing our computational experience with lift-and-project cuts generated from the LP simplex tableau, we briefly describe the Variants of the procedure that we implemented and tested. Variant 1 is the original Balas-Perregaard [7] procedure, as sketched in the pseudo-code of Fig. 1. Variants 2 and 3 are described in the next two sections.

7 Most violated cut selection rule

Here we present a variant of the lift-and-project procedure which uses a new rule for choosing the leaving and entering variables in the pivot sequence. The lift-and-project

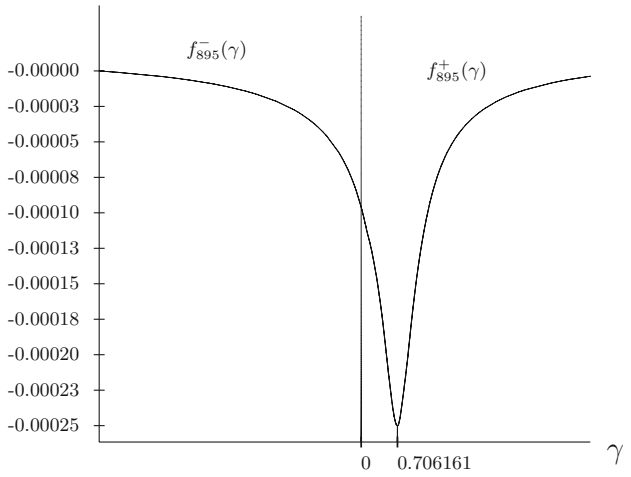

Fig. 3 Evaluation function f_{895} for the first pivot of the cut corresponding to basic variable x_{895} in `air04`

procedure in the (LP) tableau usually requires a remarkably small number of pivots to obtain the optimal L&P cut, nevertheless it may be computationally interesting to reduce this number further by studying alternate rules for this choice. The rule discussed here performs, at each iteration, the pivot to the adjacent basis in (LP) for which the objective of $(\text{CGLP})_k$ is decreased by the largest amount or, equivalently, the one for which the intersection cut obtained from the source row k of the resulting (LP) tableau is the most violated by x^*.

Let us denote by $f_i^+(\gamma)$ [resp. $f_i^-(\gamma)$] the function $f^+(\gamma)$ [resp. $f^-(\gamma)$] defined for source row k and a row i of the tableau. Recall that these functions give the violation of the intersection cut derived from the row obtained by adding γ times row i to row k, depending on the sign of γ. Thus, the violation of the cut in the adjacent basis of (LP) where variable i leaves the basis and variable j enters the basis is given by $f_i^+(\gamma_j)$ if $\gamma_j = -\overline{a}_{kj}/\overline{a}_{ij} > 0$ and $f_i^-(\gamma_j)$ if $\gamma_j = -\overline{a}_{kj}/\overline{a}_{ij} < 0$, and the most violated intersection cut which can be derived from an adjacent basis has violation

$$\hat{\sigma} = \min_{i \in I \setminus \{k\}} \min \left\{ \min_{j \in J^+} f_i^+(\gamma_j), \min_{j \in J^-} f_i^-(\gamma_j) \right\}$$

where I is the basic index set and J^+, J^- are the index sets for $\gamma_j > 0$ and $\gamma_j < 0$, respectively.

Here the variables $\hat{\imath}$ and $\hat{\jmath}$ for which this minimum is attained are selected as the leaving and entering variables, respectively. By computing the reduced costs r_{u_i} and r_{v_i}, we first identify all the candidate rows for an improving pivot. Then for each such row i we minimize the functions f_i^+ and f_i^-.

This clearly amounts to more computation at each iteration than the selection rule used in Variant 1, where only one minimization of the evaluation function is performed at each pivot. But on the other hand, the cut violation is increased at each iteration by an amount at least as large as under the selection rule of Variant 1, and therefore one

may expect to obtain in less iterations a cut with a given violation. In particular, in the presence of zero elements in the source row, it presents the advantage that fewer degenerate pivots in $(\text{CGLP})_k$ are performed.

8 Disjunctive modularization

L&P cuts are obtained from disjunctions of the type

$$(u\tilde{A}x - u_0 x_k \geq u\tilde{b}) \vee (v\tilde{A}x + v_0 x_k \geq v\tilde{b} + v_0)$$

where solving the $(\text{CGLP})_k$ optimizes the multipliers u, u_0, v and v_0. Once the optimal values for these multipliers are obtained, the cut can be further strengthened, as mentioned in Sect. 2, by using modular arithmetic on the coefficients of the integer-constrained components of x. This latter operation can be interpreted as subtracting from x_k on each side of the disjunction a product of the form mx, where m is an integer vector with 0 components for the continuous variables, and then, after deriving the cut coefficients as functions of m, optimizing the components of m over all integer values. In other words, the strengthened deepest intersection cut is the result of a sequence of two optimization procedures, first in the multipliers u, v, u_0 and v_0, then in the components of m. But this raises the quest for a procedure that would simultaneously optimize both the continuous multipliers and the integer vector m. While this is an intricate task, equivalent to finding an optimal split cut, which has been treated elsewhere [8,11], the disjunctive modularization procedure described below is meant to approximate this goal.

Consider again the equation of the source row (2) for an intersection cut or a MIG cut. By applying disjunctive modularization to this equation we mean deriving from it the modularized equation

$$y_k = \varphi_{k0} - \sum_{j \in J} \varphi_{kj} s_j \tag{15}$$

where y_k is a new, integer-constrained variable of unrestricted sign, $\varphi_{k0} = \bar{a}_{k0}$,

$$\varphi_{kj} := \begin{cases} \bar{a}_{kj} - \lfloor \bar{a}_{kj} \rfloor, & j \in J_1^+ := \{j \in J_1 : \bar{a}_{kj} - \lfloor \bar{a}_{kj} \rfloor \leq \bar{a}_{k0}\} \\ \bar{a}_{kj} - \lceil \bar{a}_{kj} \rceil, & j \in J_1^- := \{j \in J_1 : \bar{a}_{kj} - \lfloor \bar{a}_{kj} \rfloor > \bar{a}_{k0}\} \\ \bar{a}_{kj} & j \in J_2 := J \setminus J_1 \end{cases}$$

and $J_1 := J \cap \{1, \ldots, p\}$.

Clearly, every set of s_j, $j \in J$, that satisfies (2) with x_k integer, also satisfies (15) with y_k integer; hence the equation (15) is valid. Also, it is easy to see that the intersection cut derived from (15) is the strengthened intersection cut, or MIG cut derived from (2). However, at this point we do not intend to generate a cut. Instead, we append (15) to the optimal (LP) tableau and declare it the source row in place of (2) for the entire pivoting sequence. Further, after each pivot in row $\hat{\imath}$ and column $\hat{\jmath}$ the transformed row

of y_k, say

$$y_k = \varphi'_{k0} - \sum_{j \in J'} \varphi'_{kj} s_j$$

where $J' := (J \backslash \{\hat{\jmath}\}) \cup \{\hat{\imath}\}$, is treated again with disjunctive modularization. Namely, this time the row of y_k is replaced with

$$y_k = \bar{\varphi}_{k0} - \sum_{j \in J'} \overline{\varphi}_{kj} s_j$$

where $\overline{\varphi}_{k0} = \varphi'_{k0}$, and

$$\overline{\varphi}_{kj} := \begin{cases} \varphi'_{kj} - \lfloor \varphi'_{kj} \rfloor, & j \in (J'_1)^+ \\ \varphi'_{kj} - \lceil \varphi'_{kj} \rceil, & j \in (J'_1)^- \\ \varphi'_{kj} & j \in J'_2 \end{cases} \tag{16}$$

with $(J'_1)^+$, $(J'_1)^-$ and J'_2 defined analogously to J_1^+, J_1^- and J_2.

The expressions used for calculating the reduced costs r_{u_i}, r_{v_i} and the evaluation functions $f^+(\gamma)$, $f^-(\gamma)$ used for selecting the pivot element at each iteration remain valid, except for the fact that the entries $\bar{a}_{kj}$ of the current row (2) of x_k are replaced (since this is no longer the source row) with the entries $\overline{\varphi}_{kj}$ of the current row of y_k (see [2] for details).

It is clear that the modularized source row, if used for cut generation, would yield a cut that dominates the one from the unmodularized source row. It can also be shown that every iteration of the cut generating algorithm that uses disjunctive modularization improves the cut obtainable from the source row.

The iterative disjunctive modularization procedure can be implemented with the pivoting rule of the standard procedure described in Sect. 4, or with that of Sect. 6. In our computational tests we used the rule of Sect. 4.

9 Computational results

Before presenting our results, it will be useful to recall a comparison between the computational efforts required by the original procedure that generates L&P cuts by solving the higher dimensional (CGLP), and the new one that pivots in the (LP) tableau. Based on running XPRESS on about 100 test problems with each of the two methods, Perregaard [13] reported that the new method required 5% of the number of pivots and 1.3% of the time required by the original one for generating a L&P cut.

The algorithm for generating L&P cuts from the (LP) tableau was implemented, in all three of its Variants discussed above, as a cut generator called `CglLandP` [10] in the COIN-OR framework. This generator is open-source and is available since September 2006 as part of the Cut Generation Library [9]. It has been recently updated

to handle the weighted normalization constraint discussed in this paper. All the computations have been carried out on a computer equipped with a 2.93 GHz Intel Xeon CPU. The version of the COIN-OR code used is the release 2.2.0 of Cbc with an enhanced version of the `CglLandP` cut generator. We note that the results we present are slightly different from those of [3] even when the settings of the cut generator are identical. After investigation we suspect this is mainly due to modifications in the integer preprocessing code of Cbc and improvements in the underlying LP code Clp.

Our computational experiments were carried out with three versions of the general normalization constraint

$$\sum_{i=1}^{m+p+n} (u_i + v_i)\lambda_i + u_0 + v_0 = \lambda_0,$$

namely:

(a) The *unweighted* version, with $\lambda_i = 1, i = 1, \ldots, m + p + n$
(b) the *weighted* version, with $\lambda_i = ||A^i||_1$, where A^i is the i-th row of A,
(c) The *euclidean* normalization, with $\lambda_i = ||A^i||_2$ proposed by Fischetti et al. [12].

In all cases, λ_0 was set to $n+1$. Since none of the three versions seems to be uniformly better than the other, we give our results for all three.

Tables 2, 3 and 4 present a comparison of methods for generating 10 rounds of cuts at the root node, where a round means a cut for each of the 50 most fractional integer-constrained variables. For each lift-and-project cut generated a maximum of 10 nondegenerate pivots is performed in the LP tableau. In this experiment, the problems are preprocessed with COIN `CglPreproces` procedure and then 10 rounds of cuts are generated. The test set consists of all 65 problems from the MIPLIB.3 library. The four methods compared are mixed integer Gomory (MIG) cuts, and the three variants of lift-and-project (L&P) cuts presented in this paper: Variant 1 (the algorithm of [7], cf. Sects. 3 and 4), Variant 2 (the algorithm using the most violated cut selection rule, cf. Sect. 7) and Variant 3 (the algorithm using iterative disjunctive modularization, cf. Sect. 8). For each of the methods, we report the running time and the percentage of the initial integrality gap closed. The difference between the three tables is the normalization used.

As can be seen from these tables, generating lift-and-project cuts with the three different variants proposed here is more expensive than generating MIG cuts. For our test set, it took on the average 0.74 s per instance to perform 10 rounds of MIG cuts, while it took between 10 and 23 s per instance for the three variants of lift-and-project cuts. This should be seen in the light of Perregaard's observation (see [13]) that lift-and-project cut generation takes on average less than 5% of the total time needed to solve a mixed integer program. Our own experience in this respect is that on the set of problems we solved, the average and the geometric mean of the time Variant 1 spent on cut generation was 17 and 7%, respectively, of the total time needed to solve the problem; but if we restrict ourselves to the harder instances that took at least 30 s to solve, these numbers shrink to 9 and 2%, respectively. It should be noted that most of the computing time spent in the lift-and-project procedure is spent calling the routines

Table 2 Comparing 10 rounds of different cuts at the root node (*unweighted* normalization)

Name	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap
10teams	0.18	100.00	2.39	100.00	5.60	100.00	2.58	100.00
air03	0.04	100.00	0.58	100.00	0.52	100.00	0.59	100.00
air04	2.55	13.01	19.62	19.08	54.70	20.81	21.70	15.00
air05	1.55	8.34	16.72	18.13	42.93	21.31	13.67	14.46
arki001	0.13	46.17	5.04	49.47	2.35	47.82	4.86	47.00
bell3a	0.00	70.73	0.02	70.74	0.01	70.74	0.02	70.74
bell5	0.00	90.10	0.08	92.20	0.04	92.43	0.09	91.45
blend2	0.01	27.13	0.15	28.83	0.24	33.45	0.15	27.91
cap6000	0.15	60.70	0.56	54.47	0.90	54.47	0.87	54.47
dano3mip	11.01	0.06	86.56	0.10	301.55	0.11	133.77	0.19
danoint	0.06	1.74	2.29	1.41	5.54	1.85	2.35	1.44
dcmulti	0.03	74.33	1.46	89.71	2.13	87.61	1.39	84.40
dsbmip	0.07	no_gap	5.40	no_gap	3.22	no_gap	4.94	no_gap
egout	0.00	99.45	0.01	100.00	0.01	100.00	0.01	100.00
enigma	0.00	no_gap	0.05	no_gap	0.03	no_gap	0.04	no_gap
fast0507	23.97	3.04	237.68	3.91	761.70	7.14	230.59	3.58
fiber	0.04	89.22	1.23	94.72	0.65	93.78	0.85	92.83
fixnet6	0.04	85.85	0.74	87.73	1.06	88.51	0.75	87.73
flugpl	0.00	14.23	0.01	17.99	0.01	17.50	0.01	17.38
gen	0.01	87.43	0.20	96.72	0.15	82.51	0.21	97.27
gesa2	0.03	88.08	5.15	81.92	0.66	79.68	4.82	94.15
gesa2_o	0.04	83.27	3.53	76.48	1.19	94.44	4.29	74.17
gesa3	0.04	59.63	5.29	88.54	2.00	88.79	6.43	85.11
gesa3_o	0.03	62.94	3.75	90.06	1.44	87.84	3.71	84.41
gt2	0.00	100.00	0.04	95.56	0.04	95.58	0.05	96.19
harp2	0.09	38.26	1.58	55.90	1.50	54.40	1.57	58.59
khb05250	0.03	96.11	0.38	98.88	0.22	98.95	0.38	98.88
l152lav	0.18	26.46	2.85	44.36	7.55	45.51	2.83	47.91
lseu	0.00	51.86	0.05	75.73	0.05	84.51	0.08	73.53
markshare1	0.00	0.00	0.02	0.00	0.02	0.00	0.02	0.00
markshare2	0.00	0.00	0.02	0.00	0.03	0.00	0.03	0.00
mas74	0.01	8.30	0.17	9.03	0.22	8.67	0.16	8.60
mas76	0.01	7.28	0.13	8.87	0.19	8.52	0.11	7.78
misc03	0.01	18.97	0.34	27.21	0.81	29.74	0.22	20.19
misc06	0.02	62.00	0.37	100.00	0.16	97.83	0.29	100.00
misc07	0.02	0.79	0.57	3.84	0.55	4.85	0.44	2.77
mitre	0.03	100.00	1.71	100.00	0.48	100.00	2.90	100.00
mkc	0.15	36.44	7.22	45.66	2.20	38.20	6.93	30.41

Table 2 continued

Name	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap
mod008	0.01	32.22	0.04	29.20	0.06	53.71	0.05	41.97
mod010	0.01	100.00	2.31	95.79	0.41	100.00	0.40	100.00
mod011	0.34	41.99	23.44	18.75	12.65	35.59	29.00	18.21
modglob	0.01	45.50	1.16	66.24	0.80	62.52	1.16	66.24
noswot	0.00	no_gap	0.12	no_gap	0.05	no_gap	0.23	no_gap
nw04	6.17	99.20	84.46	80.70	93.41	100.00	93.57	96.57
p0033	0.00	100.00	0.00	100.00	0.00	100.00	0.00	100.00
p0201	0.02	63.66	0.68	79.93	0.83	89.56	0.68	79.72
p0282	0.01	21.83	0.18	88.83	0.24	87.86	0.16	93.73
p0548	0.02	92.96	0.46	99.83	0.26	95.72	0.49	99.48
p2756	0.05	97.42	1.14	97.87	0.43	97.11	1.11	97.03
pk1	0.00	0.00	0.11	0.00	0.16	0.00	0.11	0.00
pp08a	0.03	82.61	0.86	92.89	0.43	89.82	0.80	94.87
pp08aCUTS	0.04	59.97	0.97	83.62	0.46	76.44	0.99	86.47
qiu	0.14	10.81	6.55	24.26	14.77	30.83	6.41	24.26
qnet1	0.14	34.56	2.73	40.28	4.97	44.28	3.84	41.59
qnet1_o	0.12	55.77	1.99	69.49	3.20	68.08	3.50	71.01
rentacar	0.09	18.94	2.06	36.02	2.12	38.19	2.06	36.02
rgn	0.01	31.92	0.08	62.69	0.07	64.72	0.08	57.49
rout	0.06	13.14	1.72	32.58	3.94	41.54	1.72	34.40
set1ch	0.04	71.37	2.14	71.38	0.60	73.62	2.07	69.54
seymour	0.19	17.62	57.53	22.18	15.16	21.71	61.19	21.71
stein27	0.01	0.00	0.13	0.00	0.18	0.00	0.13	0.00
stein45	0.01	0.00	0.53	0.00	0.80	0.00	0.52	0.00
swath	0.46	26.78	16.70	28.45	4.80	29.08	12.77	28.46
vpm1	0.01	61.66	0.11	73.67	0.06	85.45	0.06	89.09
vpm2	0.01	51.52	0.41	69.87	0.34	63.45	0.39	66.82
Average	0.75	49.09	9.578	56.32	20.98	57.85	10.43	56.5
Geo. mean	–	34.08	–	42.59	–	45.32	–	42.27

of the underlying LP solver to compute the rows of the tableau, multiply the basis inverse by a vector and pivot. Overall 61% of the CPU time is spent in these procedures. For the 8 problems for which the cut generation took more than 10 s, 74% of the computing time was spent in these procedures. In our implementation, we perform these operations by calling the procedures of COIN-OR LP solver (CLP) externally through the Open Solver Interface. A tighter integration with the LP solver should reduce the overall time of the procedure.

Table 3 Comparing 10 rounds of different cuts at the root node (*weighted* normalization)

Name	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap
10teams	0.18	100.00	2.73	100.00	6.81	100.00	2.89	100.00
air03	0.04	100.00	0.63	100.00	0.50	100.00	0.55	100.00
air04	2.55	13.01	25.31	16.92	72.07	21.97	24.41	15.22
air05	1.55	8.34	23.01	14.52	51.27	20.75	18.08	12.75
arki001	0.13	46.17	6.09	47.00	2.19	47.82	5.89	47.82
bell3a	0.00	70.73	0.02	70.74	0.01	70.74	0.02	70.74
bell5	0.00	90.10	0.08	92.13	0.06	92.19	0.09	91.45
blend2	0.01	27.13	0.13	29.72	0.25	35.67	0.19	31.35
cap6000	0.15	60.70	0.80	54.47	1.00	54.47	0.86	54.47
dano3mip	11.01	0.06	125.04	0.10	499.19	0.17	99.35	0.07
danoint	0.06	1.74	2.47	1.45	5.71	1.80	2.55	1.44
dcmulti	0.03	74.33	1.59	87.52	1.68	88.02	1.40	84.00
dsbmip	0.07	no_gap	5.69	no_gap	3.59	no_gap	4.88	no_gap
egout	0.00	99.45	0.01	100.00	0.01	100.00	0.01	100.00
enigma	0.00	no_gap	0.04	no_gap	0.05	no_gap	0.04	no_gap
fast0507	23.97	3.04	304.91	3.69	857.15	6.66	306.79	2.99
fiber	0.04	89.22	1.18	92.56	1.28	96.03	1.13	92.04
fixnet6	0.04	85.85	0.95	88.07	1.23	89.79	0.93	88.07
flugpl	0.00	14.23	0.01	17.99	0.01	17.50	0.01	17.38
gen	0.01	87.43	0.35	85.25	0.12	81.42	0.37	96.72
gesa2	0.03	88.08	5.10	81.72	0.72	79.15	5.27	94.08
gesa2_o	0.04	83.27	3.80	77.30	1.01	94.91	3.77	76.18
gesa3	0.04	59.63	5.51	86.44	2.58	90.06	6.98	86.63
gesa3_o	0.03	62.94	3.91	90.32	2.34	86.31	3.71	79.96
gt2	0.00	100.00	0.04	95.72	0.04	95.58	0.05	96.19
harp2	0.09	38.26	1.58	56.96	1.89	52.26	1.74	54.62
khb05250	0.03	96.11	0.46	98.24	0.27	98.56	0.45	98.24
l152lav	0.18	26.46	3.27	43.01	7.44	44.68	3.20	44.35
lseu	0.00	51.86	0.04	75.73	0.11	68.53	0.08	73.46
markshare1	0.00	0.00	0.02	0.00	0.02	0.00	0.02	0.00
markshare2	0.00	0.00	0.02	0.00	0.03	0.00	0.03	0.00
mas74	0.01	8.30	0.17	9.03	0.23	8.67	0.17	8.60
mas76	0.01	7.28	0.13	8.87	0.21	8.52	0.12	7.78
misc03	0.01	18.97	0.43	25.41	0.80	32.47	0.23	19.55
misc06	0.02	62.00	0.34	97.83	0.28	97.83	0.41	100.00
misc07	0.02	0.79	0.48	4.66	0.64	4.57	0.43	3.42
mitre	0.03	100.00	1.93	100.00	0.66	100.00	1.65	100.00

Table 3 continued

Name	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap
mkc	0.15	36.44	7.83	37.66	2.56	35.85	7.54	48.25
mod008	0.01	32.22	0.04	29.20	0.06	53.71	0.05	41.97
mod010	0.01	100.00	1.61	100.00	0.43	100.00	0.50	100.00
mod011	0.34	41.99	26.82	20.09	19.20	36.80	28.59	19.04
modglob	0.01	45.50	1.20	63.07	0.83	62.20	1.18	63.33
noswot	0.00	no_gap	0.10	no_gap	0.07	no_gap	0.18	no_gap
nw04	6.17	99.20	108.69	80.43	24.66	100.00	64.21	100.00
p0033	0.00	100.00	0.00	100.00	0.00	100.00	0.00	100.00
p0201	0.02	63.66	0.72	79.14	1.02	86.11	0.71	77.40
p0282	0.01	21.83	0.22	88.97	0.15	92.04	0.15	93.75
p0548	0.02	92.96	0.47	99.93	0.26	95.78	0.51	99.48
p2756	0.05	97.42	1.26	97.87	0.51	97.17	1.21	97.03
pk1	0.00	0.00	0.11	0.00	0.18	0.00	0.10	0.00
pp08a	0.03	82.61	0.72	93.62	0.38	91.45	0.75	94.33
pp08aCUTS	0.04	59.97	0.82	86.44	0.58	74.88	1.04	81.31
qiu	0.14	10.81	7.62	26.40	15.88	31.35	8.71	26.40
qnet1	0.14	34.56	3.09	49.45	5.10	45.55	2.82	37.08
qnet1_o	0.12	55.77	2.14	68.12	3.27	69.28	2.23	68.33
rentacar	0.09	18.94	2.02	36.24	2.28	40.08	2.02	36.24
rgn	0.01	31.92	0.07	74.10	0.09	59.69	0.07	61.49
rout	0.06	13.14	1.87	33.32	4.10	43.54	1.87	34.89
set1ch	0.04	71.37	2.22	70.52	0.68	74.37	2.19	69.39
seymour	0.19	17.62	43.67	21.02	17.41	22.42	45.70	21.89
stein27	0.01	0.00	0.13	0.00	0.17	0.00	0.14	0.00
stein45	0.01	0.00	0.52	0.00	0.87	0.00	0.56	0.00
swath	0.46	26.78	15.87	28.44	3.49	29.23	12.61	28.46
vpm1	0.01	61.66	0.12	77.43	0.07	74.27	0.06	81.81
vpm2	0.01	51.52	0.41	68.89	0.27	61.98	0.32	65.37
Average	0.75	49.09	11.67	56.19	25.05	57.5	10.53	56.4
Geo. mean	–	34.08	–	42.48	–	45.43	–	41.65

The extra computational cost of generating cuts made it possible to close a significantly larger fraction of the integrality gap, namely 56 versus 49% for the MIG cuts, if we look at the average numbers, or between 42 and 44% for the L&P cuts, versus 34% for the MIG cuts, if we look at the geometric means. To put it differently, the geometric mean of the gap closed by the 3 variants of the L&P cuts is roughly between 1.2 and 1.3 times larger than that of the gap closed by MIG cuts.

Table 4 Comparing 10 rounds of different cuts at the root node (*euclidean* normalization)

Name	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap
10teams	0.18	100.00	3.80	100.00	5.80	100.00	2.51	100.00
air03	0.04	100.00	0.63	100.00	0.55	100.00	0.45	100.00
air04	2.55	13.01	20.11	16.92	61.23	19.48	33.17	14.12
air05	1.55	8.34	15.42	15.63	37.73	19.46	29.29	12.49
arki001	0.13	46.17	6.15	48.65	4.55	40.40	6.56	46.17
bell3a	0.00	70.73	0.01	70.74	0.01	70.74	0.02	70.74
bell5	0.00	90.10	0.09	92.56	0.06	92.95	0.08	92.12
blend2	0.01	27.13	0.21	31.45	0.30	34.79	0.29	34.46
cap6000	0.15	60.70	1.36	60.70	0.90	54.47	1.01	60.70
dano3mip	11.01	0.06	100.03	0.09	304.36	0.14	112.92	0.12
danoint	0.06	1.74	2.06	1.46	5.76	1.67	2.08	1.39
dcmulti	0.03	74.33	1.79	81.15	2.06	88.02	1.60	84.98
dsbmip	0.07	nogap	4.04	nogap	5.20	nogap	4.90	nogap
egout	0.00	99.45	0.01	100.00	0.01	100.00	0.01	100.00
enigma	0.00	nogap	0.05	nogap	0.04	nogap	0.06	nogap
fast0507	23.97	3.04	374.63	6.82	769.92	7.63	235.71	3.42
fiber	0.04	89.22	0.99	90.47	1.23	94.56	0.85	90.85
fixnet6	0.04	85.85	1.04	86.34	1.22	88.08	1.03	86.34
flugpl	0.00	14.23	0.01	17.35	0.01	17.59	0.01	17.21
gen	0.01	87.43	0.21	84.70	0.14	81.42	0.17	91.26
gesa2	0.03	88.08	4.52	89.60	0.80	90.72	4.59	88.94
gesa2_o	0.04	83.27	3.71	75.39	1.16	82.97	3.37	89.47
gesa3	0.04	59.63	5.80	88.66	2.48	86.89	5.66	86.63
gesa3_o	0.03	62.94	4.05	86.57	1.32	89.11	3.19	80.85
gt2	0.00	100.00	0.05	94.38	0.05	96.19	0.05	94.60
harp2	0.09	38.26	1.50	59.45	1.55	53.06	1.59	55.55
khb05250	0.03	96.11	0.39	99.57	0.20	98.57	0.39	99.57
l152lav	0.18	26.46	3.29	28.61	7.87	42.73	2.84	37.51
lseu	0.00	51.86	0.06	69.75	0.07	83.63	0.04	86.24
markshare1	0.00	0.00	0.02	0.00	0.02	0.00	0.02	0.00
markshare2	0.00	0.00	0.02	0.00	0.03	0.00	0.02	0.00
mas74	0.01	8.30	0.15	8.25	0.28	8.52	0.15	8.37
mas76	0.01	7.28	0.12	8.74	0.21	9.10	0.12	8.59
misc03	0.01	18.97	0.39	21.53	0.69	31.04	0.18	21.01
misc06	0.02	62.00	0.40	98.91	0.12	98.91	0.47	100.00
misc07	0.02	0.79	0.48	3.41	0.37	4.18	0.43	3.35
mitre	0.03	100.00	1.75	100.00	0.42	100.00	1.74	100.00
mkc	0.15	36.44	6.82	25.04	3.12	52.55	5.71	37.69

Table 4 continued

Name	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap	CPU(s)	%Gap
mod008	0.01	32.22	0.05	34.17	0.05	42.06	0.05	44.77
mod010	0.01	100.00	3.42	51.26	3.50	100.00	0.63	97.11
mod011	0.34	41.99	32.36	18.71	18.23	38.80	23.32	17.16
modglob	0.01	45.50	1.17	65.69	1.03	54.64	1.11	65.53
noswot	0.00	nogap	0.09	nogap	0.04	nogap	0.08	nogap
nw04	6.17	99.20	108.68	74.37	71.94	100.00	71.61	97.59
p0033	0.00	100.00	0.00	100.00	0.00	100.00	0.00	100.00
p0201	0.02	63.66	0.67	89.58	0.92	89.06	0.65	76.94
p0282	0.01	21.83	0.23	92.17	0.20	90.59	0.22	93.72
p0548	0.02	92.96	0.49	99.95	0.21	98.94	0.58	98.68
p2756	0.05	97.42	0.93	97.84	0.40	97.25	0.88	97.95
pk1	0.00	0.00	0.09	0.00	0.13	0.00	0.10	0.00
pp08a	0.03	82.61	0.81	91.53	0.58	87.27	0.76	92.27
pp08aCUTS	0.04	59.97	1.01	87.62	0.54	71.96	0.83	82.29
qiu	0.14	10.81	10.44	23.78	16.10	33.76	6.78	23.78
qnet1	0.14	34.56	2.37	42.66	4.90	44.85	2.43	37.47
qnet1_o	0.12	55.77	2.45	73.18	3.78	69.41	2.35	68.23
rentacar	0.09	18.94	1.98	30.30	2.79	38.44	1.84	30.30
rgn	0.01	31.92	0.11	54.88	0.09	57.07	0.09	62.64
rout	0.06	13.14	1.75	32.34	4.29	40.26	1.23	30.35
set1ch	0.04	71.37	1.98	71.84	0.64	75.96	1.92	74.12
seymour	0.19	17.62	46.77	22.24	28.03	24.22	43.74	21.35
stein27	0.01	0.00	0.13	0.00	0.15	0.00	0.13	0.00
stein45	0.01	0.00	0.49	0.00	0.71	0.00	0.52	0.00
swath	0.46	26.78	16.28	28.60	5.81	28.56	16.44	28.39
vpm1	0.01	61.66	0.09	82.89	0.06	77.27	0.07	76.12
vpm2	0.01	51.52	0.47	68.25	0.38	64.59	0.39	68.55
Average	0.75	49.09	12.33	54.79	21.34	57.49	9.877	56.29
Geo. mean	–	34.08	–	41.17	–	45.15	–	41.78

To make our comparisons easier to interpret, we grouped the instances according to their degree of difficulty, measured by the percentage of the gap closed by 10 rounds of MIG cuts at the root node:

- Group 1 (20 instances): gap closed $> 75\%$
- Group 2 (12 instances): $50\% \leq$ gap closed $\leq 75\%$
- Group 3 (33 instances): gap closed $< 50\%$.

Table 5 Average gap closed by 10 rounds of cuts at the root node

Method	MIG	Variant 1	Variant 2	Variant 3
Group 1	91.44	93.74	94.53	95
Group 2	60.98	78.98	80.03	78.69
Group 3	20.98	30.13	32.69	29.51

Table 5 shows the results for the runs with unweighted normalization. The ones for weighted and euclidean normalization are similar and they can be found at the website [10].

As can be seen from the table, the improvement of lift-and-project cuts over MIG cuts is larger for the harder instances where MIG cuts close less gap. For instances of Group 3, the various lift-and-project procedures close about 50% more gap than the MIG cuts, for instances of Group 2 lift-and-project cuts close about 30% more gap, while for instances in Group 1 all methods are roughly equivalent.

To more thoroughly assess the effectiveness of lift-and-project cuts, it is of course necessary to solve the instances to completion by running a branch-and-cut code and using these cuts to strengthen the LP relaxation. Table 6 describes the results of generating 10 rounds of cuts at the root node with 50 cuts per round, and then solving the problem by branch-and-bound without further cut generation, with unweighted normalization, this time with a righthand side of 1 (our tests showed no noticeable difference between a righthand side of 1 and one of $n + 1$). Again, the four cut generation methods tested are MIG cuts and the three variants of L&P cuts. For all three variants, the limit on the number of pivots for generating a cut is set to 10. These runs were performed by using `Cbc 2.2` (COIN-OR Branch and Cut) with some specific settings: a 42,000 s time limit for solving each problem was imposed; all the default cut generation procedures of `Cbc` were deactivated; the variable selection strategy used was strong branching with the default parameters of `Cbc` (i.e. performing strong branching on the 5 most fractional variables); the node selection strategy was best bound. In Table 6, for each instance and each method, we indicate the computing time and the number of search tree nodes needed to solve the problem. Among the 65 instances of MIPLIB.3, 12 were not solved in the 11.7 h time limit with any of the methods tested (namely `arki01`, `dano3mip`, `danoint`, `fast0507`, `mas74`, `markshare1`, `markshare2`, `mkc`, `rout`, `set1ch`, `seymour` and `swath`). We do not include statistics for these 12 problems. The remaining 53 instances are divided into two groups:

Group A: instances solved with MIG cuts or Variant 1 of the L&P procedure in less than 30 s and 1,000 nodes.

Group B: instances whose solution required more than 30 s or 1,000 nodes with both MIG cuts and Variant 1 of L&P.

An analog of Table 6 for the case when the weighted normalization and the euclidean normalization are used is to be found on the website [10].

Table 7 gives the averages and geometric means for the two groups of instances of Table 6. As it shows, the branch-and-bound trees generated by each of Variants 1, 2

Table 6 Comparing complete resolutions with cut-and-branch with 10 rounds of cuts and un-weighted normalization

	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes
Group A								
air03	0.66	1	1.06	1	1.09	1	1.02	1
dcmulti	1.98	79	5.22	57	8.11	109	4.05	61
dsbmip	2.06	44	6.60	43	7.15	53	6.28	42
egout	0.03	11	0.02	3	0.02	1	0.02	3
enigma	0.99	819	3.70	9,180	2.19	3,246	0.89	962
fiber	7.81	565	6.73	261	9.33	381	5.49	253
fixnet6	4.39	187	8.48	119	8.52	159	7.90	139
flugpl	0.15	643	0.17	737	0.15	689	0.17	737
gen	0.33	25	0.67	27	0.27	15	0.80	29
gesa3	3.94	199	10.22	87	8.92	115	15.79	113
gesa3_o	4.36	237	9.45	75	9.02	133	8.86	141
gt2	0.08	66	0.24	190	0.06	17	0.13	47
khb05250	0.69	35	0.68	13	0.79	19	0.73	17
l152lav	4.70	63	11.51	109	8.55	51	9.69	123
misc03	1.09	181	1.85	75	1.85	79	2.12	135
misc06	0.57	22	0.40	5	0.41	11	0.53	7
mitre	0.32	2	1.71	1	0.59	1	2.45	1
mod008	0.42	417	0.95	919	0.69	585	0.28	131
mod010	0.37	1	4.17	9	3.98	1	3.08	1
nw04	13.01	1	56.48	1	54.00	1	65.13	1

Table 6 continued

	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes
p0033	0.01	1	0.02	1	0.01	1	0.01	1
p0201	1.49	120	2.67	47	2.25	35	2.41	75
p0282	0.44	335	0.63	323	0.60	365	0.62	337
p0548	1.04	187	0.82	74	13.01	6,476	0.98	90
p2756	4.35	321	5.00	230	5.65	395	7.95	467
qnet1	11.29	63	14.11	55	15.92	59	13.14	69
qnet1_o	11.34	165	12.11	73	12.46	67	12.53	65
rentacar	2.64	13	5.66	17	5.04	19	5.50	17
rgn	0.62	723	1.30	1,121	0.69	703	1.14	1,103
vpm1	2.26	2,970	0.27	68	0.11	14	0.09	4
Group B								
10teams	36.79	304	57.38	405	113.16	911	105.69	704
air04	131.25	223	166.62	243	193.91	265	155.76	189
air05	139.46	471	152.70	325	178.04	433	147.95	351
bell3a	8.70	44,851	8.41	33,761	9.31	34,153	9.21	33,747
bell5	59.35	264,611	3.03	13,315	45.61	216,747	16.18	83,197
blend2	4.49	1,721	9.77	2,398	6.29	2,102	5.82	1,351
cap6000	28.57	10,339	27.95	10,625	29.52	10,307	31.71	10,215
gesa2	30.49	12,333	15.88	3,335	21.78	6,213	24.34	5,325
gesa2_o	39.62	9,195	31.58	8,105	29.72	3,247	37.77	6,405

Table 6 continued

	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes
harp2	14,674.71	3.92463e+06	9,694.80	2.6572e+06	22,686.87	4.68211e+06	2,996.50	1.26206e+06
lseu	4.10	22,641	1.29	3,093	0.78	1,100	0.83	1,087
mas76	650.84	674,109	223.86	820,231	173.68	750,947	315.52	681,103
misc07	74.27	10,851	105.84	21,713	95.47	18,583	82.94	16,639
mod011	476.88	3,717	418.78	3,127	413.58	3,445	394.47	3,139
modglob	124.63	155,829	10.60	5,059	196.32	256,581	10.52	5,059
nw04	13.01	1	56.48	1	54.00	1	65.13	1
pk1	77.70	261,919	88.73	296,929	90.87	302,399	119.94	406,847
pp08a	229.49	149,457	5.89	1,025	22.67	12,079	5.87	1,025
pp08aCUTS	109.62	68,023	11.32	2,991	31.30	10,679	11.15	3,079
qiu	397.35	12,673	598.43	10,049	638.63	13,849	596.93	10,049
rout	1,089.71	333,114	1,037.04	214,307	2,101.87	345,029	993.90	147,329
stein27	0.97	3,679	1.08	3,767	1.04	3,529	1.04	3,617
stein45	28.85	58,021	27.81	44,163	24.12	50,997	26.64	56,059
vpm2	34.56	47,163	30.57	36,298	13.00	15,798	14.68	17,850

Table 7 Comparing complete resolutions with cut-and-branch, 10 rounds of cuts, unweighted normalization

	MIG cuts		Lift-and-project cuts					
			Variant 1		Variant 2		Variant 3	
	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes
Group A								
Average	2.781	283.2	5.763	464	6.048	460	5.993	172.4
Geo. mean	1.0418	62.752	1.7955	48.177	1.6047	44.875	1.5569	38.564
Group B								
Average	802.3	2.639e+05	553.5	1.823e+05	1,179	2.931e+05	265.5	1.198e+05
Geo. mean	73.188	23,449	42.081	10,444	61.397	16,590	42.694	10,021
All instances								
Average	349.7	1.147e+05	243.4	7.937e+04	515.1	1.275e+05	118.6	5.211e+04
Geo. mean	6.5942	820.34	7.0579	497.26	7.8028	583.92	6.5516	430.6

and 3 are considerably smaller than the one obtained with the MIG cuts. In fact, the geometric mean of the tree sizes for the three Variants of L&P cuts is 60, 71 and 52%, respectively, of those for the MIG cuts. On the other hand, the geometric mean of the time needed to solve an instance is somewhat higher for Variants 1 and 2, but slightly lower for Variant 3, than for the MIG cuts.

However, Table 7 also shows that relative performance of the L&P cuts versus the MIG cuts tends to improve as we go from the easy instances to the harder ones. This is in accordance with the well known fact that cutting planes are of little or no use in solving easy problems, but they gain increasing significance as the problems get harder. On the hard instances of group B of Tables 6 and 7, the geometric mean of the number of search tree nodes generated by the three Variants of L&P cuts is 44.5, 70.7 and 42.7%, respectively, of the corresponding number for the MIG cuts. In terms of computing times, the geometric mean of the CPU time needed for the three Variants of L&P cuts is 57.4, 83.8 and 58.3%, respectively, of the CPU time for MIG cuts.

In order to have a more complete comparison on difficult instances, we have tested the various methods on the unsolved instances of MIPLIB.3 and additional instances from MIPLIB2003 using the parallel version of `Cbc`. The experimental setting is identical to the one of the previous experiment except that the version of `Cbc` used is 2.3.0 and the code is run in multithreaded mode using 4 cores of a 2.93 GHz Intel Xeon CPUs on a machine with 132 Gb of RAM. The time limit is set to 42,000 s of CPU time. Out of the 27 instances that we ran, 8 were solved to optimality by at least one of the procedures while the others were not solved by any.

In Table 8, we report the total CPU time and the number of nodes in the branch-and-bound tree for problems which were solved by at least one of the methods. Of the 8 instances solved by at least one of the two approaches, MIG cuts solved 5 while Variant 1 and 2 solved 7 each. Of the 5 instances solved by all methods, Variant 1 performed better on 1, Variant 2 performed better on 3, and on the remaining instance there was practically no difference.

Table 8 Results on MIPLIB2003 instances solved by at least one of the methods

Name	MIG		Variant 1		Variant 2		Variant 3	
	CPU(s)	Nodes	CPU(s)	Nodes	CPU(s)	Nodes	CPU(s)	Nodes
aflow30a.mps	17,054.60	1.50646e+06	15,671.57	949,646	12,768.88	625,301	7,353	375,661
danoint.mps	22,269.16	1.2153e+06	19,705.85	930,501	39,778.65	2.02002e+06	29,816.07	1.28017e+06
disctom.mps	7.60	1	7.66	1	7.55	1	7.57	1
fast0507.mps	>42,000	8,501	20,397.23	5,144	>42,000	1,801	>42,000	4,501
mas74.mps	2,387.95	5.96587e+06	2,281.13	5.7068e+06	1,735.10	4.07135e+06	2,733.64	5.91547e+06
mzzv11.mps	>42,000	148,001	>42,000	19,201	29,978.88	103,903	>42,000	137,501
mzzv42z.mps	>42,000	110,101	25,478.71	76,111	17,295.86	46,698	>42,000	132,601
noswot.mps	9,475.38	1.37674e+07	23,860.85	3.50208e+07	5,247.25	7.81431e+06	5,694.18	8.73469e+06

Table 9 Number of nodes and bound for problems not solved by any of the methods

Name	MIG		Variant 1		Variant 2		Variant 3	
	Nodes	Bound	Nodes	Rel. impr. (%)	Nodes	Rel. impr. (%)	Nodes	Rel. impr. (%)
aflow40b	950,501	1,071.7	625,501	1.70	513,101	1.35	633,501	1.47
arki001.mps	4.03e6	7.58e+06	3.82e6	0.01	3.34e6	0.00	3.09e6	0.01
glass4	1.03e7	8.00e8	1.42e7	12.50	6.92e6	12.50	1.24e7	12.5
manna81	4.22e6	−13, 203	4.45e6	0.00	4.21e6	0.00	4.39e6	0.00
momentum1	3,401	96,254.2	6,001	−0.18	1	0.00	13,601	6.67
momentum2	7,901	9,896.35	3,901	8.09	3,901	0.00	3,901	0.00
msc98-ip	13,601	1.97e+07	7,401	−0.67	3,601	−0.53	7,301	−0.63
net12	14,901	121.449	50,101	37.69	51,001	25.80	25,287	76.21
nsrand-ipx	2.69e6	50,320.3	2.59e6	0.48	2.36e6	0.26	2.14e6	0.40
opt1217	1.62e7	−18	1.85e7	3.18	1.62e7	2.42	1.52e7	2.71
protfold	12,001	−36.8729	8,001	6.22	4,001	6.70	11,001	5.21
rd-rplusc-21	1	100	1	0.00	1	0.00	1	0.00
roll3000	2.31e6	11,820.5	2.08e6	2.23	3.41e6	−0.56	2.25e6	−1.54
set1ch	1.94e7	51,322.3	2.88e7	−0.27	2.61e7	0.90	2.54e7	1.76
sp97ar	680,501	6.56e8	557,601	0.06	285,401	−0.01	557,801	0.02
swath	3.19e6	384.724	2.69e6	0.30	2.49e6	−0.11	2.61e6	0.17
timtab1	2.86e7	404,340	2.57e7	5.91	2.34e7	15.60	2.35e7	13.2
timtab2	1.56e7	318,052	1.43e7	29.10	1.31e7	14.69	1.41e7	16.64
tr12-30	1.49e7	89,235.5	2.06e7	−14.28	1.87e7	−10.52	2.07e7	−14.43
Average	6.48e+06		7.32e+06	4.85	6.37e+06	3.61	6.36e+06	6.33

In Table 9, we report the total number of nodes and information on the lower bound when the time limit was attained for problems which were not solved by any of the methods. For MIG cuts we report the final bound, for the various lift-and-project variants we report the percentage of bound improvement compared to MIG cuts.

References

1. Balas, E.: Disjunctive programming. Ann. Discret. Math. **5**, 3–51 (1979)
2. Balas, E.: Generating Deepest Mixed Integer Cuts by Disjunctive Modularization. Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA (2002)
3. Balas, E., Bonami, P.: New variants of lift-and-project cut generation from the LP tableau: open source implementation and testing. In: Fischetti, M., Williamson, D. (eds.) Integer Programming and Combinatorial Optimization (IPCO 12), Lecture Notes in Computer Science, vol. 4513, pp. 89–104. Springer, New York (2007)
4. Balas, E., Ceria, S., Cornuéjols, G.: A lift-and-project cutting plane algorithm for mixed 0-1 programs. Math. Program. **58**, 295–324 (1993)
5. Balas, E., Ceria, S., Cornuéjols, G.: Mixed 0-1 programming by lift-and-project in a branch-and-cut framework. Manag. Sci. **112**, 1229–1246 (1996)
6. Balas, E., Ceria, S., Cornuéjols, G., Natraj: Gomory cuts revisited. Oper. Res. Lett. **19**, 1–10 (1996)
7. 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. B. **94**, 221–245 (2003)

8. Balas, E., Saxena, A.: Optimizing over the split closure. Math. Program. A. **113**, 219–240 (2008)
9. COIN-OR website: http://www.coin-or.org/
10. CglLandP: https://projects.coin-or.org/Cgl/wiki/CglLandP
11. Dash, S., Gunluk, O., Lodi, A.: MIR closures of polyhedral sets. Math. Program. A. **121**, 33–60 (2009)
12. Fischetti, M., Lodi, A., Tramontani: On the separation of disjunctive cuts. Math. Program. A. (2009), to appear
13. Perregaard, M.: A practical implementation of lift-and-project cuts. International Symposium on Mathematical Programming, Copenhagen (2003)
14. Perregaard, M.: Generating disjunctive cuts for mixed integer programs. Ph.D. Thesis, Carnegie Mellon University (2003)