ELSEVIER

Discrete Applied Mathematics 123 (2002) 129–154

DISCRETE APPLIED MATHEMATICS

# Lift-and-project for Mixed 0–1 programming: recent progress ☆

Egon Balas*, Michael Perregaard

*Graduate School of Industrial Admin., Carnegie Mellon University, Schenley Park, Pittsburgh, PA 15213-3890, USA*

Received 28 September 1999; received in revised form 3 May 2000; accepted 15 May 2000

**Abstract**

This article reviews the disjunctive programming or lift-and-project approach to 0-1 programming, with an emphasis on recent developments. Disjunctive programming is optimization over unions of polyhedra. The first three sections of the paper define basic concepts and introduce the two fundamental results underlying the approach. Thus, section 2 describes the compact higher dimensional representation of the convex hull of a union of polyhedra, and its projection on the original space; whereas section 3 is devoted to the sequential convexifiability of facial disjunctive programs, which include mixed 0-1 programs. While these results originate in Balas' work in the early- to mid-seventies, some new results are also included: it is shown that on the higher dimensional polyhedron representing the convex hull of a union of polyhedra, the maximum edge-distance between any two vertices in 2. Also, it is shown that in the process of sequential convexification of a 0-1 program, fractional intermediate values of the variables can occur only under very special circumstances. The next section relates the above results to the matrix-cone approach of Lovász and Schrijver and of Sherali and Adams. Section 5 introduces the lift-and-project cuts of Balas, Ceria and Cornuéjols from the early nineties, and discusses the cut generating linear program (CGLP), cut lifting and cut strengthening. The next section briefly outlines the branch and cut framework in which the lift-and-project cuts turned out to be computationally useful, while section 7 discusses some crucial aspects of the cut generating procedure: alternative normalizations of (CGLP), complementarity of the solution components, size reduction of (CGLP), and ways of deriving multiple cuts from a disjunction. Finally, section 8 discusses computational results in branch-and-cut mode as well as in cut-and-branch mode. 

*Keywords:* Lift-and-project; Disjunctive programming; Mixed 0–1 programming; Cut generation

☆ Research supported by the National Science Foundation through grant DMI-9802773 and by the Office of Naval Research through contract N00014-97-1-0196.

* Corresponding author.
*E-mail address:* eb17@andrew.cmu.edu (E. Balas).

0166-218X/02/$ - see front matter 
PII: S0166-218X(01)00340-7

## 1. Introduction

This is a state-of-the-art survey of the disjunctive programming approach to mixed 0–1 programming, also known as lift-and-project, with an emphasis on current research on the subject. The foundations of this approach were laid in a July 1974 Technical Report, published 24 years later as an invited paper [1] with a foreword. For additional work on disjunctive programming in the 1970s and 1980s see [2,3,8,9,12,13,17–19,22]. In particular, [2] contains a detailed account of the origins of the disjunctive approach and the relationship of disjunctive cuts to Gomory's mixed integer cut, intersection cuts and others. Disjunctive programming received a new impetus in the early 1990s from the work on matrix cones by Lovász and Schrijver [20], see also Sherali and Adams [21]. The version that led to the computational breakthroughs of the nineties is described in the two papers by Balas et al. [5,6], the first of which discusses the cutting plane theory behind the approach, while the second deals with the branch-and-cut implementation and computational testing. Related recent developments are discussed in [7,4,10,14–16,23,25,26].

In this survey, known results are not proved, but referenced. Results that are proved are believed to be new.

### 1.1. Disjunctive programming

Disjunctive programming is optimization over unions of polyhedra. While polyhedra are convex sets, their unions of course are not. The name reflects the fact that the objects investigated by this theory can be viewed as the solution sets of systems of linear inequalities joined by the logical operations of conjunction, negation (taking of complement) and disjunction, where the nonconvexity is due to the presence of disjunctions. Pure and mixed integer programs, in particular pure and mixed 0–1 programs can be viewed as disjunctive programs; but the same is true of a host of other problems, like for instance the linear complementarity problem. Our focus will be on pure and mixed 0–1 programs.

The constraint set of a disjunctive program, called a disjunctive set, can be expressed in many different forms, of which the following two extreme ones have special significance. Let

$$P_i := \{x \in \mathbb{R}^n : A^i x \geqslant b^i\}, \quad i \in Q$$

be convex polyhedra, with $Q$ a finite index set and $(A^i, b^i)$ an $m_i \times (n+1)$ matrix, $i \in Q$, and let $P := \{x \in \mathbb{R}^n : Ax \geqslant b\}$ be the polyhedron defined by those inequalities (if any) common to all $P_i$, $i \in Q$. Then the disjunctive set $\bigcup_{i \in Q} P_i$ over which we wish to optimize some linear function can be expressed as

$$\left\{ x \in \mathbb{R}^n : \bigvee_{i \in Q} (A^i x \geqslant b^i) \right\}, \tag{1}$$

which is its *disjunctive normal form* (a disjunction whose terms do not contain further disjunctions). The same disjunctive set can also be expressed as

$$\left\{ x \in \mathbb{R}^n : Ax \geqslant b,\ \bigvee_{h \in Q_j} (d^h x \geqslant d_0^h),\ j = 1, \ldots, t \right\}, \tag{2}$$

which is its *conjunctive normal form* (a conjunction whose terms do not contain further conjunctions). Here $(d^h, d_0^h)$ is a $(n+1)$-vector for $h \in Q_j$, all $j$. The connection between (1) and (2) is that each term $A^i x \geqslant b^i$ of the disjunctive normal form (1) contains $Ax \geqslant b$ and exactly one inequality $d^h x \geqslant d_0^h$ of each disjunction of (2) indexed by $Q_j$ for $j = 1, \ldots, t$, and that all distinct systems $A^i x \geqslant b^i$ with this property are present among the terms of (1). See [3] for details on how to go from (1) to (2) and from (2) to (1).

### 1.2. Two basic ideas

The lift-and-project approach relies mainly on the following two ideas (results), the first of which uses the disjunctive normal form (1), while the second one uses the conjunctive normal form (2):

1. There is a compact representation of the convex hull of a union of polyhedra in a higher dimensional space, which in turn can be projected back into the original space. The first step of this operation may be viewed as lifting, the second step, projection. As a result one obtains the convex hull in the original space.
2. A large class of disjunctive sets, called *facial*, can be convexified sequentially, i.e. their convex hull can be derived by imposing the disjunctions one at a time, generating each time the convex hull of the current set.

## 2. Compact representation of the convex hull

**Theorem 1** (Balas [1]). *Given polyhedra* $P_i := \{x \in \mathbb{R}^n : A^i x \geqslant b^i\} \neq \emptyset$, $i \in Q$, *the closed convex hull of* $\bigcup_{i \in Q} P_i$ *is the set of those* $x \in \mathbb{R}^n$ *for which there exist vectors* $(y^i, y_0^i) \in \mathbb{R}^{n+1}$, $i \in Q$, *satisfying*

$$\begin{aligned} & x - \sum (y^i : i \in Q) = 0, \\ & A^i y^i - b^i y_0^i \geqslant 0, \\ & y_0^i \geqslant 0, \quad i \in Q, \\ & \sum (y_0^i : i \in Q) = 1. \end{aligned} \tag{3}$$

*In particular, denoting by* $P_Q := \operatorname{conv} \bigcup_{i \in Q} P_i$ *the closed convex hull of* $\bigcup_{i \in Q} P_i$ *and by* $\mathscr{P}$ *the set of vectors* $(x, \{y^i, y_0^i\}_{i \in Q})$ *satisfying* (3),

(i) *if* $x^*$ *is an extreme point of* $P_Q$, *then* $(\bar{x}, \{\bar{y}^i, \bar{y}_0^i\}_{i \in Q})$ *is an extreme point of* $\mathscr{P}$, *with* $\bar{x} = x^*$, $(\bar{y}^k, \bar{y}_0^k) = (x^*, 1)$ *for some* $k \in Q$, *and* $(\bar{y}^i, \bar{y}_0^i) = (0, 0)$ *for* $i \in Q \setminus \{k\}$.

(ii) *if* $(\bar{x}, \{\bar{y}^i, \bar{y}_0^i\}_{i\in Q})$ *is an extreme point of* $\mathscr{P}$, *then* $\bar{y}^k = \bar{x} = x^*$ *and* $\bar{y}_0^k = 1$ *for some* $k \in Q$, $(\bar{y}^i, \bar{y}_0^i) = (0,0)$, $i \in Q \setminus \{k\}$, *and* $x^*$ *is an extreme point of* $P_Q$.

Note that in this higher dimensional representation of $P_Q$, the number of variables and constraints is linear in the number $|Q|$ of polyhedra in the union, and so is the number of facets of $\mathscr{P}$. Note also that in any basic solution of the linear system (3), $y_0^i \in \{0,1\}$, $i \in Q$, automatically, without imposing this condition explicitly.

Of course, if the set $Q$ is itself exponential in the number of variables, then system (3) becomes unmanageably large. This is the case for instance if we impose simultaneously *all* the integrality conditions of a mixed 0–1 program with $p$ 0–1 variables, in which case we have a disjunction with $2^p$ terms, one for every $p$-component 0–1 point. But if we impose only disjunctions that yield a set $Q$ of manageable size, then this representation becomes extremely useful (such an approach is facilitated by the sequential convexifiability of facial disjunctive sets, see below).

In the special case of a disjunction of the form $x_j \in \{0,1\}$, when $|Q| = 2$ and

$$P_{j0} := \{x \in \mathbb{R}^n_+ : Ax \geqslant b,\ x_j = 0\},$$

$$P_{j1} := \{x \in \mathbb{R}^n_+ : Ax \geqslant b,\ x_j = 1\},$$

$P_Q := \text{conv}\,(P_{j0} \cup P_{j1})$ is the set of those $x \in \mathbb{R}^n$ for which there exist vectors $(y, y_0), (z, z_0) \in \mathbb{R}^{n+1}_+$ such that

$$x - y - z = 0,$$

$$Ay - by_0 \geqslant 0,$$

$$-y_j = 0,$$

$$Az - bz_0 \geqslant 0,$$

$$z_j - z_0 = 0,$$

$$y_0 + z_0 = 1. \tag{3'}$$

Unlike the general system (3), the system (3′), in which $|Q| = 2$, is of quite manageable size.

### 2.1. Projection and polarity

In order to generate the convex hull $P_Q$, and more generally, to obtain valid inequalities (cutting planes) in the space of the original variables, we project $\mathscr{P}$ onto the $x$-space:

**Theorem 2** (Balas [1]). $Proj_x(\mathscr{P}) = \{x \in \mathbb{R}^n : \alpha x \geqslant \beta \text{ for all } (\alpha, \beta) \in W_0\}$, *where*

$$W_0 := \{(\alpha, \beta) \in \mathbb{R}^{n+1} : \alpha = u^i A^i,\ \beta \leqslant u^i b^i \text{ for some } u^i \geqslant 0,\ i \in Q\}.$$

The polyhedral cone $W_0$ used to project $\mathscr{P}$ can be shown to be the *reverse polar cone* $P_Q^*$ of $P_Q$, i.e. the cone of all valid inequalities for $P_Q$:

**Theorem 3** (Balas [1]).

$$P_Q^* := \{(\alpha,\beta)\in\mathbb{R}^{n+1}\colon \alpha x \geqslant \beta \ \textit{for all}\ x\in P_Q\}$$
$$= \{(\alpha,\beta)\in\mathbb{R}^{n+1}\colon \alpha = u^iA^i,\ \beta\leqslant u^ib^i \ \textit{for some}\ u^i\geqslant 0,\ i\in Q\}.$$

To turn again to the special case of a disjunction of the form $x_j\in\{0,1\}$, projecting the system (3′) onto the $x$-space yields the polyhedron $P_Q$ whose reverse polar cone is

$$\begin{aligned} P_Q^* = \{(\alpha,\beta)\in\mathbb{R}^{n+1}\colon\ & \alpha \geqslant uA - u_0e_j,\\ & \alpha \geqslant vA + v_0e_j,\\ & \beta \leqslant ub,\\ & \beta \leqslant vb + v_0,\\ & u,v\geqslant 0\}, \end{aligned}$$

(where $e_j$ is the $j$th unit vector.)

One of the main advantages of the higher dimensional representation is that in projecting it back we have an easy criterion to distinguish facets of $P_Q$ from other valid inequalities.

**Theorem 4** (Balas [1]). *Assume $P_Q$ is full dimensional. The inequality $\alpha x\geqslant\beta$ defines a facet of $P_Q$ if and only if $(\alpha,\beta)$ is an extreme ray of the cone $P_Q^*$.*

### 2.2. Adjacency on the higher dimensional polyhedron

In the process of generating facets of $P_Q$, sometimes one would like to list the extreme points of $P_Q$ adjacent to a given extreme point $x$. The question arises, can one do this by using the adjacency relations on the higher dimensional polyhedron $\mathscr{P}$, for which a linear description is available?

As usual, we call two extreme points, or zero-dimensional faces, of a polyhedron adjacent if they are contained in the same one-dimensional face. In terms of the system of linear inequalities defining the polyhedron, two basic solutions are adjacent (correspond to adjacent extreme points of the polyhedron) if one can associate with them two bases that differ in exactly one column.

From Theorem 2.2 of [1], it follows that every basis for the system (3) defining $\mathscr{P}$ is of the form (modulo row and column permutations)

$$B = \begin{pmatrix} I & -E^1 & \cdots & -E^{q-1} & -I & \\ & B^1 & & & & \\ & & \ddots & & & \\ & & & B^{q-1} & & \\ & & & & B^q & -b^q \\ & \delta^1 & \cdots & \delta^{q-1} & 0 & 1 \end{pmatrix},$$

where $B^q$ is a basis for the system $A^q y^q - s^q = b^q$ (with $s^q$ a vector of surplus variables), $q = |Q|$ (i.e. $q$ plays the role of the index $k$ in statements (i) and (ii) of Theorem 1); further, for $i \in \{1, \ldots, q-1\}$, $B^i$ is a basis for the system $A^i y^i - b^i y_0^i - s^i = 0$; $\delta^i$ is a row vector whose entries are all zero if $B^i$ does not contain the column $b^i$, otherwise the entry corresponding to $b^i$ is 1 and the remaining entries are 0; $I$ is the identity matrix corresponding to the basic components of $x$ (and of $y^q$), while $E^i$ is the diagonal matrix with 1 in the positions that are basic for both $x$ and $y^i$, 0 in the remaining positions; and all blanks are zeros. Clearly, the solution corresponding to $B$ is of the form $(y^q, y_0^q) = (x, 1)$ and $(y^i, y_0^i) = 0$ for all $i \in Q \setminus \{q\}$.

Evidently, every extreme point of $P_Q$ is an extreme point of some $P_i$, $i \in Q$. In the next theorem and its Corollary, we assume that $|Q| \geqslant 2$.

**Theorem 5.** *Let $x^1$ and $x^2$ be arbitrary extreme points of $P_Q$ such that $x^1 \in P_i$, $x^2 \in P_j$, with $i \neq j$. Then there exist vectors $(y^{1i}, y_0^{1i})$, $(y^{2i}, y_0^{2i})$, $i \in Q$, such that $(x^1, \{y^{1i}, y_0^{1i}\}_{i \in Q})$ and $(x^2, \{y^{2i}, y_0^{2i}\}_{i \in Q})$ are adjacent extreme points of $\mathscr{P}$.*

**Proof.** Given $x^1$, there is exactly one $k \in Q$ such that $(y^{1k}, y_0^{1k}) = (x^1, 1)$ is a feasible solution to $A^k y^{1k} - b^k y_0^{1k} \geqslant 0$. Assigning this value to $(y^{1k}, y_0^{1k})$ and setting $(y^{1i}, y_0^{1i}) = (0, 0)$ for all $i \in Q \setminus \{k\}$ defines an extreme point $(x^1, \{y^{1i}, y_0^{1i}\}_{i \in Q})$ of $\mathscr{P}$. Similarly, given $x^2$, there is exactly one $\ell \in Q$ such that $(y^{2\ell}, y_0^{2\ell}) = (x^2, 1)$ is a feasible solution to $A^\ell y^{2\ell} - b^\ell y_0^{2\ell} \geqslant 0$; which can be used to define an extreme point $(x^2, \{y^{2i}, y_0^{2i}\}_{i \in Q})$ of $\mathscr{P}$ in which $(y^{2\ell}, y_0^{2\ell}) = (x^2, 1)$ and $(y^{2i}, y_0^{2i}) = 0$ for all $in \in Q \setminus \{\ell\}$. Furthermore, since $x^1 \neq x^2$, $\ell \neq k$.

Now although the two extreme points of $\mathscr{P}$ described above have the values of their components uniquely defined, each can be associated with a multitude of bases: indeed, while the basis associated with the $k$th subsystem in the case of $x^1$, and with the $\ell$th subsystem in the case of $x^2$, is uniquely determined by the vectors $y^{1k} = x^1$ and $y^{2\ell} = x^2$, respectively, the bases associated with the remaining subsystems, whose variables are all zero, can be chosen freely. Furthermore, moving between such bases entails only degenerate pivots. In view of this, given a basis $B$ associated with $(x^1, \{y^{1i}, y_0^{1i}\}_{i \in Q})$, in which $(y^{1k}, y_0^{1k}) = (x^1, 1)$, one can perform degenerate pivots involving only changes in the sub-basis $B^\ell$ associated with $(y^{1\ell}, y_0^{1\ell})$ until it is brought to the form required for $(y^{2\ell}, y_0^{2\ell}) = (x^2, 1)$, at which point a nondegenerate pivot can replace the entire basis $B$ associated with $(x^1, \{y^{1i}, y_0^{1i}\}_{i \in Q})$ with a basis $B'$ associated with $(x^2, \{y^{2i}, y_0^{2i}\}_{i \in Q})$. □

**Corollary 6.** *The maximum edge-distance between any pair of vertices of $\mathscr{P}$ is* 2.

**Proof.** Let $(x^1, \{y^{1i}, y_0^{1i}\}_{i \in Q})$ and $(x^2, \{y^{2i}, y_0^{2i}\}_{i \in Q})$ be two arbitrary vertices of $\mathscr{P}$, and suppose $x^1 \in P_k$, $x^2 \in P_\ell$. If $k \neq \ell$, then from Theorem 5 there exist vectors $(\bar{y}^{1i}, \bar{y}_0^{1i})$, $(\bar{y}^{2i}, \bar{y}_0^{2i})$, $i \in Q$, such that $(x^1, \{\bar{y}^{1i}, \bar{y}_0^{1i}\}_{i \in Q})$ and $(x^2, \{\bar{y}^{2i}, \bar{y}_0^{2i}\}_{i \in Q})$ are adjacent extreme points of $\mathscr{P}$. Further, from Theorem 1(ii), $(\bar{y}^{1k}, \bar{y}_0^{1k}) = (x^1, 1) = (y^{1k}, y_0^{1k})$ and $\bar{y}^{2\ell}, \bar{y}_0^{2\ell}) = (x^2, 1) = (y^{2\ell}, y_0^{2\ell})$, with $(\bar{y}^{1i}, \bar{y}_0^{1i}) = (y^{1i}, y_0^{1i}) = (0, 0)$ for all $i \in Q \setminus \{k\}$, and $(\bar{y}^{2i}, \bar{y}_0^{2i}) = (y^{2i}, y_0^{2i}) = (0, 0)$ for all $i \in Q \setminus \{\ell\}$). If $k = \ell$, i.e. $x^1$ and $x^2$ belong to the same polyhedron $P_k$, choose any vertex $(x^3, \{y^{3i}, y_0^{3i}\}_{i \in Q})$ of $\mathscr{P}$ such that $x^3 \notin P_k$.

Then by the argument above, this vertex is adjacent to both $(x^1, \{y^{1i}, y_0^{1i}\}_{i\in Q})$ and $(x^2, \{y^{2i}, y_0^{2i}\}_{i\in Q})$; hence the latter two vertices are at an edge-distance of 2 from each other. □

Thus our answer to the question raised initially is on the whole negative, in the sense that adjacency on $\mathscr{P}$ tells us almost nothing about adjacency on $P_Q$.

Next we turn to the class of disjunctive programs that are sequentially convexifiable.

## 3. Sequential convexification

A disjunctive set is called *facial* if every inequality in (2) induces a face of $P$. Zero–one programs (pure or mixed) are facial disjunctive programs, general integer programs are not. Sequential convexifiability is one of the basic properties that distinguish 0–1 programs from general integer programs.

**Theorem 7** (Balas [1]). *Let*

$$D:=\left\{x\in\mathbb{R}^n\colon Ax\geqslant b,\ \bigvee_{h\in Q_j}(d^hx\geqslant d_0^h),\ j=1,\dots,t\right\},$$

*where* $1\leqslant t\leqslant n$, $|Q_j|\geqslant 1$ *for* $j=1,\dots,t$, *and* $D$ *is facial. Let* $P_D:=conv(D)$.

*Define*

$$P^0(=P):=\{x\in\mathbb{R}^n\colon Ax\geqslant b\}$$

*and for* $j=1,\dots,t$,

$$P^j:=conv\left(P^{j-1}\cap\left\{x\colon\bigvee_{h\in Q_j}(d^hx\geqslant d_0^h)\right\}\right).$$

*Then*

$$P^t=P_D.$$

While faciality is a sufficient condition for sequential convexifiability, it is not necessary. A necessary and sufficient condition is given in [9]. The most important class of facial disjunctive programs are mixed 0–1 programs, and for that case Theorem 7 asserts that if we denote

$$P_D:=\text{conv}\,\{x\in\mathbb{R}^n_+\colon Ax\geqslant b,\ x_j\in\{0,1\},\ j=1,\dots,p\},$$

$$P^0:=\{x\in\mathbb{R}^n_+\colon Ax\geqslant b\}$$

and define recursively for $j=1,\dots,p$

$$P^j:=\text{conv}\,(P^{j-1}\cap\{x\colon x_j\in\{0,1\}),$$

then

$$P^p = P_D.$$

Thus, in principle, a 0–1 program with $p$ 0–1 variables can be solved in $p$ steps. Here each step consists of imposing the 0–1 condition on one new variable and generating all the inequalities that define the convex hull of the set defined in this way.

### 3.1. Disjunctive rank

Based on this property, one can define the *disjunctive rank* of an inequality $\alpha x \geqslant \beta$ for a mixed 0–1 program as the smallest integer $k$ for which there exists an ordering $\{i,\ldots,i_p\}$ of $\{1,\ldots,p\}$ such that $\alpha x \geqslant \beta$ is valid for $P^k$. In other words, an inequality is of rank $k$ if it can be obtained by $k$, but not by fewer than $k$, applications of the recursive procedure defined above. Clearly, the disjunctive rank of a cutting plane for 0–1 programs is bounded by the number of 0–1 variables. It is known that the number of 0–1 variables is not a valid bound for the Chvatal rank of an inequality.

The above definition of the disjunctive rank is based on using the disjunctions $x_j \in \{0,1\}$, $j = 1,\ldots,p$. Tighter bounds can be derived by using stronger disjunctions. For instance, a 0–1 program whose constraints include the generalized upper bounds $\sum(x_j: j \in Q_i) = 1$, $i = 1,\ldots,t$, with $|Q_i| = |Q_j| = q$, $Q_i \cap Q_j = \emptyset$, $i,j \in \{1,\ldots,t\}$, and $|\bigcup_{i=1}^{t} Q_i| = p$, can be solved as a disjunctive program with the disjunctions

$$\bigvee_{j \in Q_i} (x_j = 1), \quad i = 1,\ldots,t(=p/q)$$

in which case the disjunctive rank of any cut is bounded by the number $t = p/q$ of *GUB* constraints.

Here is a new result concerning the nature of solutions generated during the sequential convexification process. In this process, the following question arises.

### 3.2. Fractionality of intermediate points

Define

$$P := \{x \in \mathbb{R}^n_+ : Ax \geqslant b\},$$

$$P^1 := \text{conv}\,\{x \in P : x_1 \in \{0,1\}\}$$

and suppose the system defining $P^1$ has already been generated, i.e.

$$P^1 := \{x \in P : \alpha^i x \geqslant \beta^i,\ i \in I\}$$

is at hand. Now for $j \in \{2,\ldots,n\}$, consider

$$P^{1j} := \text{conv}\,\{x \in P^1 : x_j \in \{0,1\}\}.$$

By virtue of the sequential convexifiability of 0–1 programs (Theorem 7),

$$P^{1j} := \text{conv}\,\{x \in P : x_1 \in \{0,1\}, x_j \in \{0,1\}\};$$

in other words, by imposing the 0–1 condition on $x_j$ we automatically enforce the condition $x_1 \in \{0,1\}$ too. But what about the intermediate solutions generated "on the

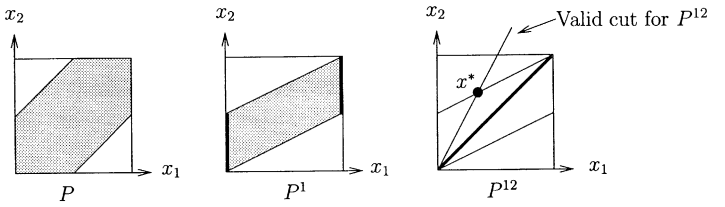


Fig. 1. $0 < x_1^* < 1,\ 0 < x_2^* < 1$.

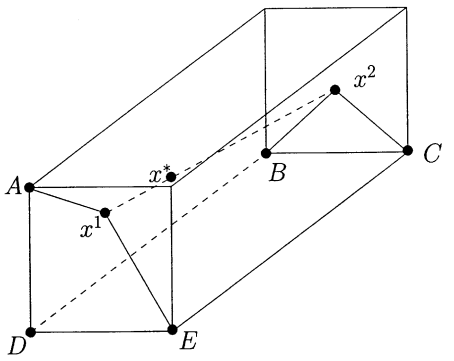


Fig. 2. Hyperplane $ABC$ (not shown) intersects edge $[x^1, x^2]$ in a point $x^*$ with $0 < x_j^* < 1$, $j = 1, 2, 3$.

way" from $P^1$ to $P^{1j}$ for some $j$? As we add new cutting planes to our linear program, will the resulting solutions satisfy $x_1 \in \{0, 1\}$? In general, this cannot be guaranteed, as illustrated by the two-dimensional example of Fig. 1, where a valid cut generated in the process of getting from $P^1$ to $P^{12}$ produces the fractional solution $x^*$.

The desired property cannot be maintained even if we restrict ourselves to the use of facet defining cutting planes, as illustrated in Fig. 2. Here $P^1$ is the convex hull of $A$, $D$, $E$, $x^1$, $B$, $C$, $x^2$, and the cutting plane through the points $A$, $B$, $C$ is facet defining for both $P^{12}$ and $P^{13}$. Yet, this hyperplane intersects the edge $[x^1, x^2]$ in its interior, hence in a point fractional in all three components.

However, a property almost as strong as the one whose absence we have just illustrated, still holds:

**Theorem 8.** *Let $P^1$ and $P^{1j}$, $j \in \{2, \ldots, n\}$, be as above. Let $\alpha x \geqslant \beta$ be a valid inequality for $P^{1j}$, $j \in \{2, \ldots, n\}$, and let $x^*$ be an extreme point of $P^1 \cap \{x: \alpha x \geqslant \beta\}$. Then $0 < x_1^* < 1$ implies $0 < x_j^* < 1$ for all $j \in \{2, \ldots, n\}$.*

**Proof.** Suppose $x_j^* \in \{0, 1\}$ for some $j \in \{2, \ldots, n\}$; then $x^* \in P^{1j}$. If $x^*$ is an extreme point of $P^{1j}$, then from Theorem 7, $x_1^* \in \{0, 1\}$ and we are done. If $x^*$ is not extreme, then $\alpha x^* = \beta$ and there exist points $x^1, x^2 \in P^{1j}$, $x^1 \neq x^2$, such that $x^* = \lambda x^1 + (1 - \lambda)x^2$ for some $0 < \lambda < 1$, and $\alpha x^1 > \beta$, $\alpha x^2 < \beta$. But this contradicts the assumption that $\alpha x \geqslant \beta$ is valid for $P^{1j}$. □

What is the situation if we use more general cuts, not necessarily valid for all $P^{1j}$, $j \in \{2,\dots,n\}$? The result of Theorem 8 can be extended to this case as follows.

Let $N$ be the index set of 0–1 variables, and for any $S \subseteq N$, denote

$$P^S := \text{conv}\{x \in P:\ x_j \in \{0,1\},\ j \in S\}.$$

Suppose we have derived the convex hull of $P \cap \{x:\ x_j \in \{0,1\},\ j \in S_1\}$ for some $S_1 \subset N$, namely

$$P^{S_1} := \{x \in P:\ \alpha^i x \geqslant \beta^i,\ i \in M\};$$

let $Dx \geqslant d$ be an arbitrary set of inequalities, and denote

$$\tilde{P}^{S_1} := \{x \in P^{S_1}:\ Dx \geqslant d\}.$$

**Theorem 9.** *Let $S_2,\dots,S_q$ be all the distinct minimal subsets of $N \setminus S_1$ such that $Dx \geqslant d$ is valid for $P^{S_1 \cup S_j}$ for $j = 2,\dots,q$; and let $x^*$ be any extreme point of $\tilde{P}^{S_1}$. Then $0 < x_i^* < 1$ for some $i \in S_1$ implies $0 < x^*_{k(j)} < 1$ for some $k(j) \in S_j$ for each $j \in \{2,\dots,n\}$.*

**Proof.** Suppose there exists $j_* \in \{2,\dots,n\}$ such that $x_k^* \in \{0,1\}$ for all $k \in S_{j_*}$. Then $x^* \in P^{S_1 \cup S_{j_*}}$. If $x^*$ is an extreme point of $P^{S_1 \cup S_{j_*}}$, then from Theorem 7 $x_i^* \in \{0,1\}$ for all $i \in S_1$ and we are done. If $x^*$ is not extreme, then there exists an inequality $D_i x \geqslant d_i$ of the system $Dx \geqslant d$ such that $D_i x^* = d_i$, and $x^*$ is the convex combination of two points $x^1, x^2 \in P^{S_1 \cup S_j}$ such that $D_i x^1 > d_i$ and $D_i x^2 < d_i$, a contradiction. □

## 4. Another derivation of the basic results

The two basic ingredients of our approach, the lifting/projection technique and sequential convexification, can also be derived by the following procedure [5]. Define

$$P := \{x \in \mathbb{R}^n:\ \tilde{A}x \geqslant \tilde{b}\} \subseteq \mathbb{R}^n$$

and

$$P_D := \text{conv}\{x \in P:\ x_j \in \{0,1\},\ j = 1,\dots,p\}$$

with the inequalities $x \geqslant 0$ and $x_j \leqslant 1$, $j = 1,\dots,p$, included in $\tilde{A}x \geqslant \tilde{b}$.

1. Select an index $j \in \{1,\dots,p\}$. Multiply $\tilde{A}x \geqslant \tilde{b}$ with $1 - x_j$ and $x_j$ to obtain the nonlinear system

$$(1 - x_j)(\tilde{A}x - \tilde{b}) \geqslant 0,$$

$$x_j(\tilde{A}x - \tilde{b}) \geqslant 0. \tag{4}$$

2. Linearize (4) by substituting $y_i$ for $x_i x_j$, $i = 1,\dots,n$, $i \neq j$, and $x_j$ for $x_j^2$.
3. Project the resulting polyhedron onto the $x$-space.

**Theorem 10** (Balas et al. [5]). *The outcome of steps* $1,2,3$ *is*

$$conv(P \cap \{x\colon x_j \in \{0,1\}\}).$$

**Corollary 11** (Balas et al. [5]). *Repeating steps* $1,2,3$ *for each* $j \in \{1,\ldots,p\}$ *in turn yields* $P_D$:

The fact that this procedure is isomorphic to the one introduced earlier can be seen by examining the outcome of step 2. In fact, the linearized system resulting from step 2 is precisely $(3')$, the higher dimensional representation of the disjunctive set defined by the constraint $x_j \in \{0,1\}$ (see [5] for details).

The above 3-step procedure is a streamlined version of the matrix cone procedure of Lovász and Schrijver [20]. The latter involves in step 1 multiplication with $1-x_j$ and $x_j$ for every $j \in \{1,\ldots,p\}$ rather than just one. While obtaining $P_D$ by this procedure still involves $p$ iterations of steps 1, 2, 3, the added computational cost brings a reward: after each iteration, the coefficient matrix of the linearized system must be positive semidefinite, a condition that can be used in various ways to derive strong bounds or cuts (see [20,7]).

Another similar procedure, due to Sherali and Adams [21], is based on multiplication with every product of the form $(\pi_{j\in J_1} x_j)(\pi_{j\in J_2}(1-x_j))$, where $J_1$ and $J_2$ are disjoint subsets of $\{1,\ldots,p\}$ such that $|J_1 \cup J_2| = t$ for some $1 \leqslant t \leqslant p$. Linearizing the resulting nonlinear system leads to a higher dimensional polyhedron whose strength (tightness) is intermediate between $P$ and $P_D$, depending on the choice of $t$: for $t = p$, the polyhedron becomes identical to $\mathscr{P}$ defined by the system (3) for a 0–1 polytope.

## 5. Generating cuts

The implementation of the disjunctive programming approach into a practical 0–1 programming algorithm had to wait until the early 1990s. It required not only the choice of a specific version of disjunctive cuts, but also a judicious combination of cutting with branching, made possible in turn by the discovery of an efficient procedure for lifting cuts generated in a subspace (for instance, at a node of the search tree) to be valid in the full space (i.e. throughout the search tree).

### 5.1. Deepest cuts

As mentioned earlier, if $P_D$ is full dimensional, then facets of $P_D$ correspond to extreme rays of the reverse polar cone $P_D^*$. To generate such extreme rays, for each 0–1 variable $x_j$ that is fractional at the linear programming optimum, we solve a linear program over a normalized version of the cone $P_D^*$ corresponding to the disjunction $x_j = 0 \vee x_j = 1$, with an objective function aimed at cutting off the linear programming optimum $\bar{x}$ by as much as possible. This "cut generating linear program" for the $j$th

variable is of the form

$$\begin{aligned} \min\ & \alpha\bar{x} - \beta \\ \text{s.t. } & \alpha - uA + u_0 e_j \geqslant 0, \\ & \alpha - vA - v_0 e_j \geqslant 0, \\ & -\beta + ub = 0, \\ & -\beta + vb + v_0 = 0, \\ & u, v \geqslant 0 \end{aligned} \qquad (\text{CGLP})_j$$

and (i) $\beta \in \{1, -1\}$, or (ii) $\sum_j |\alpha_j| \leqslant 1$.

For details, see [5,6].

The normalization constraint (i) or (ii) has the purpose of turning the cone $P_D^*$ into a polyhedron. In case of (i) this is achieved by using $\beta = 1$ or $\beta = -1$, whichever is indicated. In case of (ii), by substituting $\alpha_j^+ - \alpha_j^-$ for $\alpha_j$, with $\alpha_j^+$, $\alpha_j^- \geqslant 0$, $\forall j$. A third normalization, proposed later and used in computational experiments subsequent to [6], is

$$\text{(iii)} \quad \sum_i u_i + u_0 + \sum_i v_i + v_0 = 1.$$

The merits and demerits of various normalizations will be discussed later.

Solving $(\text{CGLP})_j$ yields a cut $\alpha x \geqslant \beta$, where

$$\alpha_k = \begin{cases} \max\{ua_k, va_k\} & k \in N \setminus \{j\} \\ \max\{ua_j - u_0, va_j + v_0\} & k = j, \end{cases}$$

with $a_k$ the $k$th column of $A$, and

$$\beta = \min\{ub, vb + v_0\}.$$

This cut maximizes the amount $\beta - \alpha\bar{x}$ by which $\bar{x}$ is cut off. The experiments of [6] indicated that the most efficient way of generating cuts is to stop short of solving (CGLP) to optimality. This idea was implemented by ignoring those columns of (CGLP) associated with constraints of $P$ not tight at the optimum, except for the lower and upper bounding constraints on the 0–1 variables.

### 5.2. Cut lifting

In general, a cutting plane derived at a node of the search tree defined by a subset $F_0 \cup F_1$ of the 0–1 variables, where $F_0$ and $F_1$ index those variables fixed at 0 and 1, respectively, is only valid at that node and its descendants in the tree (where the variables in $F_0 \cup F_1$ remain fixed at their values). Such a cut can in principle be made valid at other nodes of the search tree, where the variables in $F_0 \cup F_1$ are no longer fixed, by calculating appropriate values for the coefficients of these variables—a procedure called lifting. However, calculating such coefficients is in general a daunting task, which may require the solution of an integer program for every coefficient. One important advantage of the cuts discussed here is that the multipliers $u$, $u_0$, $v$, $v_0$

obtained along with the cut vector $(\alpha, \beta)$ by solving $(\mathrm{CGLP})_j$ can be used to calculate by closed form expressions the coefficients $\alpha_h$ of the variables $h \in F_0 \cup F_1$.

While this possibility of calculating efficiently the coefficients of variables absent from a given subproblem (i.e. fixed at certain values) is crucial for making it possible to generate cuts during a branch-and-bound process that are valid throughout the search tree, its significance goes well beyond this aspect. Indeed, most columns of $A$ corresponding to nonbasic components of $\bar{x}$ typically play no role in determining the optimal solution of $(\mathrm{CGLP})_j$ and could therefore be ignored. In other words, the cuts can be generated in a subspace involving only a subset of the variables, and then lifted to the full space. This is the procedure followed in [5,6], where the subspace used is that of the variables indexed by some $R \subset N$ such that $R$ includes all the 0–1 variables that are fractional and all the continuous variables that are positive at the LP optimum. The lifting coefficients for the variables not in the subspace, which are all assumed to be at their lower bound, are then given by

$$\alpha_\ell := \max\{ua_\ell, va_\ell\}, \quad h \in N \setminus R$$

where $u$ and $v$ are the optimal vectors obtained by solving $(\mathrm{CGLP})_j$.

These coefficients always yield a valid lifted inequality. If normalization (i) is used in $(\mathrm{CGLP})_j$, the resulting lifted cut is exactly the same as the one that would have been obtained by applying $(\mathrm{CGLP})_j$ to the problem in the full space. If other normalizations are used, the resulting cut may differ in some coefficients.

### 5.3. Cut strengthening

The cut $\alpha x \geqslant \beta$ derived from a disjunction of the form $x_j \in \{0, 1\}$ can be strengthened by using the integrality conditions on variables other than $x_j$, as shown in [8] (see also Section 7 of [2]). Indeed, if $x_k$ is such a variable, the coefficient

$$\alpha_k := \max\{ua_k, va_k\}$$

can be replaced by

$$\alpha'_k := \min\{ua_k + u_0\lceil m_k \rceil, va_k - v_0\lfloor m_k \rfloor\},$$

where

$$m_k := \frac{va_k - ua_k}{u_0 + v_0}.$$

For a proof of this statement, see [2,5] or [6]. The strengthening "works", i.e. produces an actual change in the coefficient, only if

$$|ua_k - va_k| \geqslant u_0 + v_0. \tag{5}$$

Indeed, if (5) does not hold, then either $ua_k > va_k$ and $0 \geqslant m_k > -1$, or $ua_k < va_k$ and $0 \leqslant m_k < 1$; in either case, $\alpha'_k = \alpha_k$. Furthermore, the larger the difference $|ua_k - va_k|$, the more room there is for strengthening the coefficient in question.

This strengthening procedure can also be applied to cuts derived from disjunctions other than $x_j \in \{0, 1\}$, including disjunctions with more than two terms. In the latter case, however, the closed form expression for the value $m_k$ used above has to be

replaced by a procedure for calculating those values, whose complexity is linear in the number of terms in the disjunction (see [8] or [2] for details).

### 5.4. The overall cut generating procedure

Considering what we said about solving the cut generating LP in a subspace and then lifting the resulting cut to the full space and strengthening it, the actual cut generating procedure is not just "lift and project", but rather RLPLS, an acronym for

- RESTRICT the problem to a subspace defined from the LP optimum, and choose a disjunction;
- LIFT the disjunctive set to describe its convex hull in a higher dimensional space;
- PROJECT the polyhedron describing the convex hull onto the original (restricted) space, generating cuts;
- LIFT the cuts into the original full space;
- STRENGTHEN the lifted cuts.

## 6. Branch-and-cut

No cutting plane approach known at this time can solve large, hard integer programs just by itself. Repeated cut generation tends to produce a flattening of the region of the polyhedron where the cuts are applied, as well as numerical instability which can only partly be mitigated by a tightening of the tolerance requirements. Therefore, the successful use of cutting planes requires their combination with some enumerative scheme. One possibility is to generate cutting planes as long as that seems profitable, thereby creating a tighter linear programming relaxation than the one given originally, and then to solve the resulting problem by branch and bound. Another possibility, known as branch-and-cut, consists of branching and cutting intermittently; i.e., when the cut generating procedure "runs out of steam", move elsewhere in the feasible set by branching. This approach depends crucially on the ability to lift the cuts generated at different nodes of the search tree so as to make them valid everywhere.

The first successful implementation of lift-and-project for mixed 0–1 programming in a branch-and-cut framework was the mixed integer program optimizer (MIPO) code described in [6]. The procedure it implements can be outlined as follows.

Nodes of the search tree (subproblems created by branching) are stored along with their optimal LP bases and associated bounds. Cuts that are generated are stored in a pool. The cut generation itself involves the RLPLS process described earlier. Furthermore, cuts are not generated at every node, but at every $k$th node, where $k$ is a cutting frequency parameter.

At any given iteration, a subproblem with best (weakest) lower bound is retrieved from storage and its optimal LP solution $\bar{x}$ is recreated. Next, all those cuts in the pool that are tight for $\bar{x}$ or violated by it, are added to the constraints and $\bar{x}$ is updated by reoptimizing the LP.

At this point, a choice is made between generating cuts or skipping that step and going instead to branching. The frequency of cutting is dictated by the parameter $k$ that is problem dependent, and calculated after generating cuts at the root node. Its value is a function of several variables believed to characterize the usefulness of cutting planes for the given problem, primarily the average depth of the cuts obtained. In our experiments, the most frequently used value of $k$ was 8.

If cuts are to be generated, this happens according to the RLPLS scheme described above. First, a subspace is chosen by retaining all the 0–1 variables fractional and all the continuous variables positive at the LP optimum, and removing the remaining variables along with their lower and upper bounds. A lift and project cut is then generated from each disjunction $x_j \in \{0,1\}$ for $j$ such that $0 < \bar{x}_j < 1$ (this is called a round of cuts). Each cut is lifted and strengthened; and if it differs from earlier cuts sufficiently (the difference between cuts is measured by the angle between their normals), it is added to the pool; otherwise it is thrown away. After generating a round of cuts, the current LP is reoptimized again.

If cuts are not to be generated (or have already been generated), a fractional variable is chosen for branching on a disjunction of the form $x_j \in \{0,1\}$; i.e., two new subproblems are created, their lower bounds are calculated, and they are stored. The branching variable is chosen as the one with largest (in absolute value) cost coefficient among those whose LP optimal value is closest to 0.5.

## 7. Variations on the cut generating LP

The solution of the cut generating LP depends on two factors: the objective function and the normalization used. The choice of the former is dictated by the fact that the immediate goal is to cut off the LP optimum by as much as possible. The normalization is a different story.

### 7.1. Alternative normalizations

It was shown in [1] that if normalization (i) is used, then (CGLP) has a finite minimum if and only if $\bar{x}\lambda \in P_D$ for some $\lambda \in \mathbb{R}_+$. This condition is satisfied for certain classes of problems, for instance set covering (for $\beta = 1$) and set packing (for $\beta = -1$), but not for others, and the absence of a finite minimum leads to complications.

In case of normalizations (ii) or (iii), a different difficulty arises. If $P_D$ is full-dimensional, then the inequality $\alpha x \geqslant \beta$ defines a facet of $P_D$ if and only if $(\alpha, \beta)$ is an extreme ray of the reverse polar cone $P_D^*$. If $P_D^*$ is truncated or intersected with a single hyperplane in $(\alpha, \beta)$-space, then the extreme points of the resulting polyhedron correspond to extreme rays of $P_D^*$. But if $P_D^*$ is truncated or intersected by multiple hyperplanes, that will typically result in a polyhedron whose extreme points do not always correspond to extreme rays of $P_D^*$. This is exactly what happens in the case of normalizations (ii) and (iii). In the case of (ii), the constraint $\sum(|\alpha_j|: j \in N) \leqslant 1$, which requires $\alpha$ to belong to an $n$-dimensional octahedron, is equivalent to imposing on $\alpha$ the $2^n$ inequalities $\delta^i \alpha \leqslant 1$, $\delta^i \in \{1,-1\}^n$, $i = 1,\ldots,2^n$, which define the facets

of the octahedron. In the case of (iii), the constraint $\sum_i u_i + u_0 + \sum_i v_i + v_0 = 1$ guarantees that $(\mathrm{CGLP})_j$ will have a finite minimum for every nonnegative objective function. Since the multipliers $u_i, v_i, i = 0, \ldots, m$, are all required to be nonnegative, the normalization (iii) bounds each multipler; and since $\alpha$ is bounded from below by a linear combination of those multipliers, it follows that the objective function of $(\mathrm{CGLP})_j$ is bounded from below for any nonnegative $\bar{x}$.

If (iii) is replaced by

$$(\mathrm{iii}') \quad \sum_i u_i + u_0 + \sum_i v_i + v_0 + \sum_k s_k + \sum_k t_k = 1,$$

where $s_k \geqslant 0$, $t_k \geqslant 0$, $k = 1, \ldots, n$, are surplus variables used to bring $(\mathrm{CGLP})_j$ to equality form, then (iii′) bounds $P_D^*$ in every direction. This follows because the surplus variables are now also required to be nonnegative, and thus $\alpha$ is also bounded from above by a linear combination of the multipliers.

Although the higher dimensional cone is truncated by a single hyperplane through either (iii) or (iii′), the outcome in the $(\alpha, \beta)$-subspace may correspond to a truncation of $P_D^*$ by multiple hyperplanes, and thus an extreme point of the resulting polyhedron may not correspond to an extreme ray of $P_D^*$. To avoid this difficulty, we propose another normalization, whose generic form is (iv) $\alpha y = 1$. Let $(\mathrm{CGLP})^y$ denote the problem with this normalization. The advantage of normalization (iv) is that it intersects $P_D^*$ with a single hyperplane in the $(\alpha, \beta)$-space and thus has the effect that every extreme point of the resulting polyhedron corresponds to an extreme ray of $P_D^*$. This does not imply that every extreme point of the higher-dimensional $(\mathrm{CGLP})^y$ corresponds to an extreme ray of $P_D^*$; but it does imply that if the objective $\min(\alpha\bar{x} - \beta)$ is bounded then there *exists* an optimal extreme point of $(\mathrm{CGLP})^y$ which corresponds to an extreme ray of $P_D^*$.

**Theorem 12.** *Let $(CGLP)^y$ be feasible. Then it has a finite minimum if and only if $\bar{x} + y\lambda \in P_D$ for some $\lambda \in \mathbb{R}$.*

**Proof.** If $(\mathrm{CGLP})^y$ is unbounded in the direction of minimization, then there exists $(\tilde{\alpha}, \tilde{\beta}) \in P_Q^*$ with $\bar{x}^{\mathrm{T}}\tilde{\alpha} < \tilde{\beta}$ and $y^{\mathrm{T}}\tilde{\alpha} = 0$. But then $(\bar{x} + y\lambda)^{\mathrm{T}}\tilde{\alpha} < \tilde{\beta}$ for all $\lambda \in \mathbb{R}$, hence $\bar{x} + y\lambda \notin P_D$.

Conversely, if $\bar{x} + y\lambda \notin P_D$ for all $\lambda \in \mathbb{R}$, there exists $(\hat{\alpha}, \hat{\beta}) \in \mathbb{R}^{n+1}$ such that $\hat{\alpha}x \geqslant \hat{\beta}$ for all $x \in P_D$ and $\hat{\alpha}(\bar{x} + y\lambda) < \hat{\beta}$ for all $\lambda \in \mathbb{R}$, i.e. $\hat{\alpha}y = 0$ and $\hat{\alpha}\bar{x} < \hat{\beta}$. But then $(\hat{\alpha}, \hat{\beta})$ is a direction of unboundedness for $(\mathrm{CGLP})^y$. □

**Theorem 13.** *If $(CGLP)^y$ has an optimal solution $(\tilde{\alpha}, \tilde{\beta})$, then*

$$\bar{x}^{\mathrm{T}}\tilde{\alpha} - \tilde{\beta} = \lambda^* := \min\{\lambda : \bar{x} + y\lambda \in P_D\}$$

*and*

$$(\bar{x} + y\lambda^*)^{\mathrm{T}}\tilde{\alpha} = \tilde{\beta}.$$

**Proof.** Let $(\tilde{\alpha}, \tilde{\beta})$ be an optimal solution to $(\mathrm{CGLP})^y$, and define $\lambda^* := \min\{\lambda : \bar{x} + y\lambda \in P_D\}$. Since $\tilde{\alpha}y \neq 0$, there exists $\lambda^0 \in \mathbb{R}$ such that $(\bar{x} + y\lambda^0)^{\mathrm{T}}\tilde{\alpha} = \tilde{\beta}$. Further,

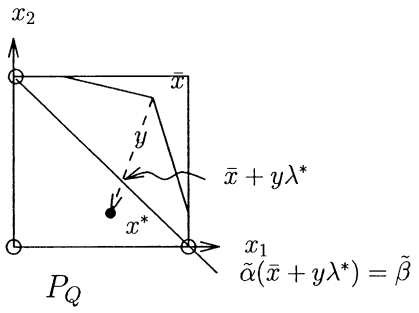


Fig. 3. $y = x^* - \bar{x}$.

$\bar{x}^{\mathrm{T}}\tilde{\alpha} - \tilde{\beta} = y^{\mathrm{T}}\tilde{\alpha}\lambda^0 = \lambda^0$. We claim that $\lambda^0 = \lambda^*$. For suppose $\lambda^0 > \lambda^*$. Then

$$\begin{aligned}(\bar{x} + y\lambda^*)^{\mathrm{T}}\tilde{\alpha} - \tilde{\beta} &= \bar{x}^{\mathrm{T}}\tilde{\alpha} - \tilde{\beta} + y^{\mathrm{T}}\tilde{\alpha}\lambda^* \\ &= -\lambda^0 + \lambda^* \quad (\text{since } y^{\mathrm{T}}\tilde{\alpha} = 1) \\ &< 0,\end{aligned}$$

i.e. the point $\bar{x} + y\lambda^*$ violates the inequality $\tilde{\alpha}x \geqslant \tilde{\beta}$, contradicting $\bar{x} + y\lambda^* \in P_D$.

Now suppose $\lambda^0 < \lambda^*$. Then there exists a hyperplane $\bar{\alpha}x = \bar{\beta}$ such that $\bar{\alpha}x \geqslant \bar{\beta}$ for all $x \in P_Q$, $\bar{\alpha}(\bar{x} + y\lambda^*) = \bar{\beta}$, and $\bar{\alpha}(\bar{x} + y\lambda^0) < \bar{\beta}$. Further, since $\bar{\alpha}y\lambda^0 < \bar{\alpha}y\lambda^*$ and $\lambda^0 < \lambda^*$, it follows that $\bar{\alpha}y > 0$ and so w.l.o.g. we may assume that $(\bar{\alpha}, \bar{\beta})$ is scaled so as to make $\bar{\alpha}y = 1$. But then

$$\bar{\alpha}\bar{x} - \bar{\beta} < \bar{\alpha}y\lambda^0 = \tilde{\alpha}\bar{x} - \tilde{\beta},$$

contradicting the optimality of $(\tilde{\alpha}, \tilde{\beta})$ for $(\mathrm{CGLP})^y$. □

**Corollary 14.** *Let $y := x^* - \bar{x}$ for some $x^* \in P_Q$. Then $(CGLP)_y$ has an optimal solution $(\tilde{\alpha}, \tilde{\beta})$ such that (i) $\tilde{\alpha}\bar{x} < \tilde{\beta}$; and (ii) $\tilde{\alpha}x = \tilde{\beta}$ is the supporting hyperplane of $P_Q$ that intersects the line segment $(\bar{x}, x^*]$ at the point closest to $x^*$.*

Theorem 13 and Corollary 14 are illustrated in Fig. 3.

### 7.2. Complementarity of solution components

Consider the linear program $(\mathrm{CGLP})_j$ used to generate a cut from the disjunction $x_j = 0$ or $x_j = 1$. $(\mathrm{CGLP})_j$ has a set of trivial solutions corresponding to the constraint set $Ax \geqslant b$. Assuming that normalization (iii) is used, set $\bar{u}_i = \bar{v}_i = \frac{1}{2}$ for some $i \in M$, $\bar{u}_h = \bar{v}_h = 0$ for all $h \in M \setminus \{i\}$ and $\bar{u}_0 = \bar{v}_0 = 0$. Then $(\bar{\alpha}, \bar{\beta}, \bar{u}, \bar{u}_0, \bar{v}, \bar{v}_0)$ is a solution to $(\mathrm{CGLP})_j$ with $\alpha = a^i$ and $\beta = b_i$, i.e., the coefficient vector of the $i$th constraint of $Ax \geqslant b$. We call a basic solution *nontrivial* if it is not of this type.

**Theorem 15.** *Any nontrivial basic solution $w := (\alpha, \beta, u, u_0, v, v_0)$ to $(CGLP)_j$ satisfies $u \cdot v = 0$.*

**Proof.** We assume that $(\mathrm{CGLP})_j$ uses normalization (iii). An analogous reasoning proves the other cases. Let $A$ have rows $a_h$, $h \in M$, let $\bar{w}$ be a basic solution, and suppose $\bar{u}_i \bar{v}_i > 0$ for some $i \in M$. W.l.o.g., assume $0 < \bar{u}_i \leqslant \bar{v}_i$. Define

$$\hat{u}_h := \begin{cases} 0 & h = i \\ \rho \bar{u}_h & h \in M \setminus \{i\} \end{cases}, \quad \hat{v}_h := \begin{cases} \rho(\bar{v}_h - \bar{u}_h) & h = i \\ \rho \bar{v}_h & h \in M \setminus \{i\} \end{cases}$$

$$\hat{u}_0 = \rho \bar{u}_0, \quad \hat{v}_0 := \rho \bar{v}_0, \quad \hat{\alpha} := \rho(\bar{\alpha} - \bar{u}_i a_i), \quad \hat{\beta} := \rho(\bar{\beta} - \bar{u}_i b_i),$$

with $\rho := 1/(1 + 2\bar{u}_i)$, and

$$\tilde{u}_h := \begin{cases} 2\sigma \bar{u}_h & h = i \\ \sigma \bar{u}_h & h \in M \setminus \{i\} \end{cases}, \quad \tilde{v}_h := \begin{cases} \sigma(\bar{v}_h + \bar{u}_h) & h = i \\ \sigma \bar{v}_h & h \in M \setminus \{i\} \end{cases}$$

$$\tilde{u}_0 := \sigma \bar{u}_0, \quad \tilde{v}_0 := \sigma \bar{v}_0, \quad \tilde{\alpha} := \sigma(\bar{\alpha} + \bar{u}_i a_i), \quad \tilde{\beta} := \sigma(\bar{\beta} + \bar{u}_i b_i)$$

with $\sigma := 1/(1 + 2\bar{u}_i)$.

Then $\hat{w}$ and $\tilde{w}$ are both feasible. But since $w$ is nontrivial, $\hat{w} \neq 0 \neq \tilde{w}$ and so it is easily verified that $(1/2\rho)\hat{w} + (1/2\sigma)\tilde{w} = \bar{w}$, which contradicts the assumption that $\bar{w}$ is basic. □

The complementarity property shown in Theorem 15 means that while the two variables $u_0, v_0$ associated with the inequalities $x_j \leqslant 0$ and $x_j \geqslant 1$ may both be (and typically are) positive at the optimum, the pair $(u_i, v_i)$ associated with the $i$th inequality of $Ax \geqslant b$ is complementary for every $i$: at most one of the two variables can be positive. This is a consequence of the intuitively plausible fact, that a given inequality of $Ax \geqslant b$ can be profitably added with a positive multiplier either to one term of the disjunction, or to the other, but not to both. In fact, in addition to the complementarity of the pairs $(u_i, v_i)$, typically both members of many pairs are 0. In other words, some of the inequalities of $Ax \geqslant b$ do not contribute to the improvement of the cut, whichever term of the disjunction they are added to. This has led us to a search for criteria by which to decide for each inequality of $Ax \geqslant b$, whether it should be added to the first term of the disjunction with multiplier $u_i$, or to the second term with the multiplier $v_i$, or not included at all in the cut generating LP.

### 7.3. Reduced-size (CGLP)

After some experimentation with several different criteria, we concluded that the best indicator of the usefulness of the presence of an inequality of $Ax \geqslant b$, $x \geqslant 0$, in one term or the other of the disjunction is to be found in the optimal simplex tableau of the linear program $\max\{cx: x \in P\}$. Namely, suppose we want to build the cut generating LP for the disjunction $x_k \in \{0, 1\}$, where $\bar{x}_k$ is a fractional component of the LP optimum $\bar{x}$. Let the row of the optimal simplex tableau associated with $x_k$ be

$$x_k = \bar{a}_{k0} - \sum_{j \in J} \bar{a}_{kj} x_j, \tag{6}$$

where $J$ is the index set of nonbasic variables and $0 < \bar{a}_{k0} = \bar{x}_k < 1$, and the nonbasic variables are all at their lower bound. Next we restrict the system $Ax \geqslant b$, $x \geqslant 0$ to the subspace obtained by removing all nonbasic structural variables and all constraints that are not binding at the LP optimum (hence all basic surplus variables). (Here structural variables are the components of $x$, whereas surplus variables stand for the components of $s = Ax - b$.) We are then left with only the basic structural variables and the nonbasic surplus variables, and can write the resulting system as

$$Bx_B - s_M = b_M,$$

$$x_B, s_M \geqslant 0.$$

Note that $B$ has $|M|$ columns and $|M|$ rows and is nonsingular. Multiplying with $B^{-1}$ yields

$$x_B = B^{-1}b_M + B_k^{-1}s_M,$$

a system whose row corresponding to $x_k$ is

$$x_k = B_k^{-1}b_M + B_k^{-1}s_M,$$

where $B_k^{-1}$ is row $k$ of $B^{-1}$. This is just another way of writing the equation that remains after we remove from (6) the nonbasic structural variables, and replace the notation $x_j$ with $s_j$ for the surplus variables:

$$x_k = \bar{a}_{k0} + \sum_{i \in M} (-\bar{a}_{ki}) s_i,$$

where $\bar{a}_{k0} = \bar{x}_k = B_k^{-1}b_M$, and for $i \in M$, $\bar{a}_{ki} = -B_{ki}^{-1}$, with $B_{ki}^{-1}$ the $i$th component of $B_k^{-1}$.

The simplest disjunctive cut derived from the condition $x_k \leqslant 0 \vee x_k \geqslant 1$, namely the intersection cut from the pair of halfspaces $0 \leqslant x_k \leqslant 1$, is known to be (see [2]) $\pi s_M \geqslant \pi_0$, where

$$\pi_0 = \bar{x}_k(1 - \bar{x}_k)$$

and for $i \in M$,

$$\pi_i := \max\{\pi_i^1, \pi_i^2\}$$

with

$$\pi_i^1 := (\bar{x}_k - 1)B_{ki}^{-1}, \quad \pi_i^2 := \bar{x}_k B_{ki}^{-1}.$$

We wish to construct a basic solution of $(\mathrm{CGLP})_k$, whose $(\alpha, \beta)$-component yields the cut $\alpha x \geqslant \beta$ obtained by expressing $\pi s_M \geqslant \pi_0$ in terms of $x$. For this purpose, we write $\alpha = (\alpha_B, \alpha_R)$ where $\alpha_B$ stands for the components associated with the columns of $B$, and $\alpha_R$ for the components that have been removed. We now define

$$\alpha_B := \pi B, \quad \beta := \pi_0 + \pi b_M,$$

$$u := \pi - \pi^1, \quad v := \pi - \pi^2,$$

$$u_0 := 1 - \bar{x}_k, \quad v_0 := \bar{x}_k. \tag{7}$$

**Theorem 16.** *The vector* $w:=(\alpha_B, \beta, u, u_0, v, v_0)$ *defined by* (7) *is a basic feasible solution of* $(\mathrm{CGLP})_k$ *with the normalization* $u_0 + v_0 = 1$.

**Proof.** Since $\alpha_R$ has been removed, the expression $\alpha - uA + u_0 e_k$ reduces to $\alpha_B - uB + u_0 e_k$, with $e_k$ the unit vector in the subspace of $\alpha_B$. We then have

$$\begin{aligned}\alpha_B - uB + u_0 e_k &= \pi B - (\pi - \pi^1)B + (1 - \bar{x}_k)e_k \\ &= \pi^1 B + (1 - \bar{x}_k)e_k \\ &= (\bar{x}_k - 1)B_k^{-1}B + (1 - \bar{x}_k)e_k = 0,\end{aligned}$$

since $B_k^{-1}B = e_k$. Next,

$$\begin{aligned}\alpha_B - vB - v_0 e_k &= \pi B - (\pi - \pi^2)B - \bar{x}_k e_k \\ &= \pi^2 B - \bar{x}_k e_k \\ &= \bar{x}_k B_k^{-1} B - \bar{x}_k e_k = 0.\end{aligned}$$

Further,

$$\begin{aligned}-\beta + ub_M &= -\pi_0 - \pi b_M + (\pi - \pi^1)b_M \\ &= -\bar{x}_k(1 - \bar{x}_k) - \pi^1 b_M \\ &= -\bar{x}_k(1 - \bar{x}_k) + (1 - \bar{x}_k)B_k^{-1}b_M = 0,\end{aligned}$$

since $B_k^{-1}b_M = \bar{x}_k$. Also,

$$\begin{aligned}-\beta + vb_M + v_0 &= -\pi_0 - \pi b_M + (\pi - \pi^2)b_M + \bar{x}_k \\ &= -\bar{x}_k(1 - \bar{x}_k) - \pi^2 b_M + \bar{x}_k \\ &= -\bar{x}_k + (\bar{x}_k)^2 - \bar{x}_k B_k^{-1} b_M + \bar{x}_k = 0.\end{aligned}$$

Finally,

$$u_0 + v_0 = (1 - \bar{x}_k) + \bar{x}_k = 1.$$

Furthermore, (7) implies that $u, v \geqslant 0$.

This proves that $w$ is feasible. To see that it is basic, note that there are $2|M| + 2$ constraints satisfied at equality, and the same number of nonnegative variables, whose coefficient vectors are linearly independent. □

Using the basic solution (7), we construct the associated simplex tableau of $(\mathrm{CGLP})_k$, and among the nonbasic variables $u_i, v_i$, we keep only those with negative reduced cost, while removing the others. Our interpretation that these are the only variables likely to improve the cut (in terms of the chosen objective) is more than born out by our computational experience: as shown in the computational section of this paper, the cuts obtained from this smaller (CGLP) tend to be just as strong as those obtained from the full fledged problem.

Our approach for constructing a starting solution for $(\mathrm{CGLP})_k$ highlights the connection between this lift-and-project cut and the mixed integer Gomory cut, which in this case (since the nonbasic 0–1 variables have been removed) is identical to the intersection cut from the pair of half-spaces $0 \leqslant x_k \leqslant 1$. Thus the lift-and-project cut can be viewed as a generalization of the mixed integer Gomory cut $\pi s_M \geqslant \pi_0$, where the generalization consists in optimally combining each of the inequalities $\pi^1 s_M \geqslant \pi_0$ and $\pi^2 s_M \geqslant \pi_0$ with some of the constraints of $P$ before taking the component-wise maximum of $\pi^1$ and $\pi^2$.

### 7.4. Multiple cuts from a disjunction

There are several ways of deriving more than one cut from a given disjunction. The approach proposed in [4] was to generate several facets of $P_Q$ containing its optimal extreme point $x^{\mathrm{opt}}$. This approach asks for the calculation of $x^{\mathrm{opt}}$ (recall, we are talking about a disjunction with two terms, not too expensive to solve), to be used to generate $n$ facets of $P_Q$ containing $x^{\mathrm{opt}}$. The way to accomplish this is to replace the objective function of $(\mathrm{CGLP})_j$ by $\min(x^{\mathrm{opt}})^{\mathrm{T}}\alpha - \beta$, which results in a linear program whose optimal solutions $(\alpha, \beta, u, u^0, v, v^0)$ yield all the valid inequalities $\alpha x \geqslant \beta$ (including those that define facets of $P_Q$) satisfied at equality by $x^{\mathrm{opt}}$ (Theorem 1 of [4]). Thus, having obtained one such optimal solution, one may generate all the others by pivoting in columns with zero reduced cost. In theory this is a way of generating all the facets of $P_Q$ that contain $x^{\mathrm{opt}}$. In practice, the massive degeneracy that is typically present in the optimal tableau of this problem makes the procedure of finding alternative optima with the relevant $(\alpha, \beta)$-components computationally rather expensive.

An alternative way of generating multiple cuts from the same disjunction is to explore near-optimal solutions to $(\mathrm{CGLP})_j$ by forcing to 0 some component of $(u, v)$ positive in the optimal tableau. This has been explored by Ceria and Pataki [14]; we have also tried it, with results slightly better than those obtained with the previous approach.

Finally, a third way which we found considerably more useful than either of these two, is the following. Having found an optimal solution to $(\mathrm{CGLP})_j$, we go back to the optimal simplex tableau of $\min\{cx: x \in P\}$, and generate all adjacent solutions to $\bar{x}$ obtainable by a single pivot: let these solutions be $x^1, \ldots, x^k$. We then use each one of them in turn to replace $\bar{x}$ in the objective function of $(\mathrm{CGLP})_j$. This yields reasonably good results, to be discussed in the computational section.

## 8. Computational experience

### 8.1. Results in branch-and-cut mode

The procedure described in Sections 5 and 6 was implemented in the code MIPO, described in detail in [6]. This implementation of MIPO does not have its own linear programming routine; instead, it calls a simplex code whenever it has to solve or reoptimize an LP. In the experiments of [6] the LP solver used was that of CPLEX

Table 1

| OSL | | CPLEX | | MINTO | | MIPO | |
|---|---|---|---|---|---|---|---|
| First | Second | First | Second | First | Second | First | Second |
| *Ranking by number of search tree nodes* | | | | | | | |
| 15 | 2 | 0 | 4 | 4 | 10 | 11 | 10 |
| *Ranking by CPU time* | | | | | | | |
| 2 | 3 | 6 | 6 | 10 | 3 | 12 | 14 |

2.1. The test bed consisted of 29 test problems from MIPLIB and other sources, ranging in size from about 30 to 9000 0–1 variables, and about 20 to 2000 constraints. The large majority of these problems have a real world origin; they were contributed mostly by people who tried, not always successfully, to solve them. Of the MIPLIB problems, most of those not included into the testbed were omitted as too easily solved by straight branch-and-bound; two problems were excluded because their LP relaxation exceeded our dimensioning. MIPO was compared with MINTO, OSL and CPLEXMIP 2.1 (the most advanced version available at the time of the experiments). The outcome (see [6] for detailed results) is best summarized by showing the number of times a code ranked first, and second, both in terms of search tree nodes and in terms of computing time. This is done in Table 1, whose two parts correspond to Tables 9 and 10 of [6].

In a sense, MIPO turned out to be the most robust among the four codes: it was the only one that managed to solve all 29 test problems, and it ranked first or second in computing time on 26 out of the 29 instances.

Other experiments with lift-and-project in an enumerative framework are reported on in [24], where S. Thienel compares the performance of ABACUS, an object oriented branch-and-cut code, in two different modes of operation, one using lift-and-project cuts and the other using Gomory cuts; with the outcome that the version with lift-and-project cuts is considerably faster on all hard problems, where hard means requiring at least 10 min.

Little experimentation has taken place so far with cuts derived from stronger disjunctions than the 0–1 condition on a single variable. In [7] the MIPO procedure was run on maximum clique problems, where the higher dimensional formulation used to generate cuts was the one obtained by multiplying the constraint set with inequalities of the form $1-\sum(x_j : j\in S)\geqslant 0$, $x_j\geqslant 0$, $j\in S$, where $S$ is a stable set. This is the same as the higher dimensional formulation derived from the disjunction

$$(x_j = 0,\ j\in S)\vee(x_{j_1}=1,\ x_j=0,\ j\in S\setminus\{j_1\})\vee\ldots\vee$$
$$(x_{j_s}=1,\ x_j=0,\ j\in S\setminus\{j_s\}),$$

where $s=|S|$. As this disjunction is more powerful than the standard one, the cuts obtained were stronger; but they were also more expensive to generate, and without some specialized code to solve the highly structured cut generating LP's of this formulation,

the trade-off between the strength of the cuts and the cost of generating them favored the weaker cuts from the standard disjunction.

### 8.2. Results in cut-and-branch mode

Bixby et al. [11] report on their computational experience with a parallel branch-and-bound code, run on 51 test problems after generating several types of cuts at the root node. One of the cut types used was disjunctive or lift-and-project cuts, generated essentially as in [6] with normalization (ii), but without restriction to a subspace and without strengthening. Since deriving these cuts in the full space is expensive, the routine generating them was activated only for 4 of the hardest problems. Their addition to the problem constraints reduced the integrality gap by 58.7%, 94.4%, 99.9% and 94.4%, respectively.

In [14], Ceria and Pataki report computational results with a disjunctive cut generator used in tandem with the CPLEX branch and bound code. Namely, the cut generator was used to produce 2 and 5 rounds of cuts from the 0–1 disjunctions for the 50 most promising variables fractional at the LP optimum, after which the resulting problem with the tightened LP relaxation was solved by the CPLEX 5.0 MIP code. This "cut-and-branch" procedure was tested on 18 of the hardest MIPLIB problems and the results were compared to those obtained by using CPLEX 5.0 without the cut generator. The outcome of the comparison can be summarized as follows (see Table 2 of [14] for details).

- The total running time of the cut-and-branch procedure was less than the time without cuts for 14 of the 18 problems; while the opposite happened for the remaining 4 problems.
- Two of the problems solved by cut-and-branch in 8 and 3 min, respectively could not be solved by CPLEX alone in 20 h.
- For six problems the gain in time was more than 4-fold.

Very good results were obtained on two difficult problems outside the above set. One of them, *set1ch*, could not be solved by CPLEX alone, which after exhausting its memory limitations stopped with a solution about 15% away from the optimum. On the other hand, running the cutting plane generator for 10 rounds on this problem produced a lower bound within 1.4% of the integer optimum, and running CPLEX on the resulting tightened formulation solved the problem to optimality in 28 s.

The second difficult problem, *seymour*, was formulated by Paul Seymour in an attempt to find a minimal irreducible configuration in the proof of the four color theorem. It was not solved to optimality until very recently. The value of the LP relaxation is 403.84, and an integer solution of 423.00 was known. The best previous known lower bound of 412.76 was obtained by running a parallel computer with 16 processors for about 60 h, using about 1400 Mbytes of memory. The cut-and-branch procedure applied to this case generated 10 rounds of 50 cuts in about 10.5 h, and produced a lower bound of 413.16, using less than 50 Mbytes of memory. Running CPLEX for another 10 h on this tightened formulation then raised the bound to 414.20. More recently, using the same lift-and-project cuts, but a more potent computing environment, the problem was solved to optimality, cf. Pataki [20a]

Table 2
Computational results with the full size and the reduced (CGLP)

| Problem | Optimum of the preprocessed | Lower bound after adding cuts | | Average number of columns in CGLP | | Average number of pivots in CGLP | |
|---|---|---|---|---|---|---|---|
| | LP | Full CGLP | Reduced CGLP | Full | Reduced | Full | Reduced |
| P0548 | 3126 | 5713 | 5709 | 820 | 305.6 | 25.62 | 0.18 |
| P2756 | 2703 | 2880 | 2880 | 1984 | 313.4 | 5.39 | 0.08 |
| Pp08a | 2748 | 2103 | 2103 | 875 | 490.8 | 19.26 | 5.33 |
| Vpm2 | 10.27 | 11.05 | 11.02 | 873 | 374.1 | 37.18 | 4.50 |
| 10-teams | 897.0 | 904.0 | 904.0 | 1528 | 584.2 | 709.9 | 258.7 |
| Danoint | 62.64 | 62.65 | 62.66 | 1989 | 637.4 | 1230.0 | 619.3 |
| Misc07 | 1415 | 1415 | 1415 | 763 | 203.9 | 154.0 | 46.2 |
| Pk1 | 0 | 0 | 0 | 198 | 48.0 | 21.67 | 17.0 |
| Seymour | 267.8 | 271.1 | 270.8 | 13022 | 2338.0 | 1936.7 | 158.4 |
| Vpm1 | 16.43 | 17.01 | 17.01 | 893 | 388.0 | 54.64 | 5.93 |
| Mod010 | 6532 | 6533 | 6543 | 1315 | 542.6 | 258.4 | 121.6 |
| L152lav | 4656 | 4659 | 4659 | 874 | 366.7 | 186.4 | 92.8 |
| Set1ch | 30427 | 35174 | 35174 | 2768 | 1515.8 | 45.0 | 6.43 |

### 8.3. Results with a reduced-size CGLP

We have extensively tested the reduced-size (CGLP) constructed from the optimal simplex tableau for $\min\{cx: x \in P\}$ by using the complementarity of $(u, v)$, as described in Section 7.

In the experiment summarized in Table 2, we chose 13 of the harder MIPLIB problems, and for each instance we solved, for each 0–1 variable fractional at the LP optimum, a (CGLP) (with normalization (iii)) not restricted to a subspace, in two versions: the full size (CGLP) and the reduced size CGLP. The results are shown in Table 2. Every problem was preprocessed by CPLEX before generating cuts.

It is clear from the table, that the reduced (CGLP), while generating cuts whose strength—as measured by the lower bounds they provide—is fully equal to that of the cuts generated by the full (CGLP), requires a computational effort that is several times smaller.

### 8.4. Results with multiple cuts from a disjunction

In a computational experiment meant to test the efficiency of generating multiple cuts from the same disjunction, we ran two versions of MIPO on a set of MIPLIB problems. Both versions used the same formula for calculating at the root node the cutting frequency. The first version used the standard approach of generating one cut from each disjunction, whereas the second version generated $q + 1$ cuts from each disjunction, solving each $(\mathrm{CGLP})_j$ for the objective functions $\min\{\alpha x^* - \beta\}$, with $x^* = \bar{x}, x^1, \ldots, x^q$, where $\bar{x}$ is the optimal solution to $\min\{cx: x \in P\}$, and $x^1, \ldots, x^q$ are extreme points of $P$ adjacent to $\bar{x}$, obtainable by one pivot in the simplex tableau

Table 3

| | Time (CPU Sec) | | Search tree nodes | |
|---|---|---|---|---|
| Problem | Version 1 | Version 2 | Version 1 | Version 2 |
| C-fat-200-1 | 413 | 113 | 49 | 31 |
| Pp08CUTS | 358 | 299 | 2891 | 1743 |
| San200-0.9-3 | 2358 | 1751 | 495 | 609 |
| Stein45 | 1335 | 1789 | 20761 | 24711 |
| Vpm1 | 295 | 190 | 8811 | 4153 |
| Vpm2 | 432 | 399 | 8901 | 2877 |
| 10 teams | 1490 | [a] | 259 | [a] |

[a]Time or memory limit exceeded.

associated with $\bar{x}$ ($q$ was limited to at most 0.5 times the number of basic variables). The outcome is shown in Table 3 for the problems that required at least 250 s to solve.

As can be seen, the extra work spent on generating more cuts pays off more often than not: the computing time is smaller in version 2 in five out of the seven instances.

## References

[1] E. Balas, Disjunctive programming: properties of the convex hull of feasible points, Invited paper, with a foreword by G. Cornuéjols and W. Pulleyblank, Discrete Appl. Math. 89 (1998) 3–44 (originally MSRR# 348, Carnegie Mellon University, July 1974).
[2] E. Balas, Disjunctive programming, Ann. Discrete Math. 5 (1979) 3–51.
[3] E. Balas, Disjunctive programming and a hierarchy of relaxations for discrete optimization problems, SIAM J. Algebraic Discrete Methods 6 (1985) 466–485.
[4] E. Balas, A modified lift-and-project procedure, Math. Programming 79 (1997) 19–31.
[5] E. Balas, S. Ceria, G. Cornuéjols, A lift-and-project cutting plane algorithm for mixed 0–1 programs, Math. Programming 58 (1993) 295–324.
[6] E. Balas, S. Ceria, G. Cornuéjols, Mixed 0–1 programming by lift-and-project in a branch-and-cut framework, Manag. Sci. 42 (1996) 1229–1246.
[7] E. Balas, S. Ceria, G. Cornuéjols, G. Pataki, Polyhedral methods for the maximum clique problem, in: D. Johnson, M. Trick (Eds.), Clique, Coloring and Satisfiability: The Second DIMACS Challenge, The American Mathematical Society, Providence, RI, 1996, pp. 11–27.
[8] E. Balas, R.G. Jeroslow, Strengthening cuts for mixed integer programs, European J. Oper. Res. 4 (1980) 224–234.
[9] E. Balas, J. Tama, J. Tind, Sequential convexification in reverse convex and disjunctive programming, Math. Programming 44 (1989) 337–350.
[10] N. Beaumont, An algorithm for disjunctive programming, European J. Oper. Res. 48 (1990) 362–371.
[11] R. Bixby, W. Cook, A. Cox, E. Lee, Computational experience with parallel mixed integer programming in a distributed environment, Ann. Oper. Res. 90 (1999) 19–45.
[12] C. Blair, Two rules for deducing valid inequalities for 0–1 programs, SIAM J. Appl. Math. 31 (1976) 614–617.
[13] C. Blair, Facial disjunctive programs and sequences of cutting planes, Discrete Appl. Math. 2 (1980) 173–180.
[14] S. Ceria, G. Pataki, Solving integer and disjunctive programs by lift-and-project, in: R.E. Bixby, E.A. Boyd, R.Z. Rios-Mercado (Eds.), IPCO VI, Lecture Notes in Computer Science, Vol. 1412, Springer, Berlin, 1998, pp. 271–283.
[15] S. Ceria, J. Soares, Disjunctive cuts for mixed 0–1 programming: duality and lifting, GSB, Columbia University, 1997.

[16] S. Ceria, J. Soares, Convex programming for disjunctive optimization, GSB, Columbia University, 1997.
[17] R.G. Jeroslow, Cutting plane theory: disjunctive methods, Ann. Discrete Math. 1 (1977) 293–330.
[18] R.G. Jeroslow, Representability in mixed integer programming I: characterization results, Discrete Appl. Math. 17 (1987) 223–243.
[19] R.G. Jeroslow, Logic based decision support: mixed integer model formulation, Ann. Discrete Math. 40 North-Holland, Amsterdam, 1989.
[20] L. Lovász, A. Schrijver, Cones of matrices and set functions and 0–1 optimization, SIAM J. Optim. 1 (1991) 166–190.
[20a] G. Pataki, Private communications, March 2001.
[21] H. Sherali, W. Adams, A hierarchy of relaxations between the continuous and convex hull representations for zero–one programming problems, SIAM J. Discrete Math. 3 (1990) 411–430.
[22] H. Sherali, C. Shetty, Optimization with disjunctive constraints, Lecture Notes in Economics and Mathematical Systems, Vol. 181, Springer, Berlin, 1980.
[23] R.A. Stubbs, S. Mehrotra, A branch-and-cut method for 0–1 mixed convex programming, Department of Industrial Engineering, Northwestern University, 1996.
[24] S. Thienel, ABACUS: A branch-and-cut system, Doctoral Dissertation, Faculty of Mathematics and The Natural Sciences, University of Cologne, 1995.
[25] M. Turkay, I.E. Grossmann, Disjunctive programming techniques for the optimization of process systems with discontinuous investment costs-multiple size regions, Ind. Engng. Chem. Res. 35 (1996) 2611–2623.
[26] H.P. Williams, An alternative explanation of disjunctive formulations, European J. Oper. Res. 72 (1994) 200–203.