J Optim Theory Appl (2007) 134: 353–370
DOI 10.1007/s10957-007-9263-4

New Branch-and-Cut Algorithm for Bilevel Linear Programming

C. Audet · G. Savard · W. Zghal

Published online: 11 July 2007
© Springer Science+Business Media, LLC 2007

Abstract Linear mixed 0–1 integer programming problems may be reformulated as equivalent continuous bilevel linear programming (BLP) problems. We exploit these equivalences to transpose the concept of mixed 0–1 Gomory cuts to BLP. The first phase of our new algorithm generates Gomory-like cuts. The second phase consists of a branch-and-bound procedure to ensure finite termination with a global optimal solution. Different features of the algorithm, in particular, the cut selection and branching criteria are studied in details. We propose also a set of algorithmic tests and procedures to improve the method. Finally, we illustrate the performance through numerical experiments. Our algorithm outperforms pure branch-and-bound when tested on a series of randomly generated problems.

Keywords Bilevel linear programming · Gomory cuts · Linear mixed 0–1 integer programming · Branch-and-cut algorithms

1 Introduction

Bilevel programming is an optimization problem in which a subset of the decision variables are constrained to lie in the optimal set of a second optimization problem, parameterized by the other subset of variables. Closely related to the static Stackelberg equilibrium problem [1] and the mathematical programs with equilibrium constraints [2], the bilevel programming problem may be expressed as

$$\max_{x,y}\{f(x,y) : (x,y) \in X,\ y \in \mathcal{S}(x)\}, \tag{1}$$

Communicated by H.P. Benson.

Work of the authors was partially supported by FCAR, MITACS and NSERC grants.

C. Audet · G. Savard (✉) · W. Zghal
GERAD and École Polytechnique de Montréal, Montreal, PQ, Canada
e-mail: gilles.savard@polymtl.ca

where $\mathcal{S}(x)$ denotes the set of optimal solutions of a mathematical program parameterized by the vector x, i.e.,

$$\mathcal{S}(x) = \arg\max_{w} \{g(x, w) : (x, w) \in Y\}, \tag{2}$$

where $f, g : \mathbb{R}^{n_x+n_y} \to \mathbb{R}$ and X, Y are subsets of $\mathbb{R}^{n_x+n_y}$. In game theory terminology, the first player, called the *leader*, optimizes his own problem (1) by anticipating the optimal reaction w of the second player, called the *follower*. This formulation corresponds to the *optimistic* formulation and implies that whenever the optimal solution set $\mathcal{S}(x)$ does not reduce to a singleton for some x, the leader may select any solution w among this *indifferent* reaction set that suits him best. Conversely, the *pessimistic* formulation models the case where the leader protects himself against the worst possible situation (see Loridan and Morgan [3] for a discussion on these two concepts). In this paper, we consider the optimistic formulation.

Bilevel programs are intrinsically hard to solve, being typically non convex and non differentiable. In particular, it has been shown by Jeroslow [4] that linear bilevel programming BLP, where all functions involved are linear, is NP-hard. This result has been strengthened by Hansen et al. [5] by showing that BLP is strongly NP-hard, and refined by Vicente, Savard and Júdice [6] who proved that obtaining a mere certificate of local optimality is also strongly NP-hard. Due to this intractability, researches on algorithms followed two main areas. The first concerns the theoretical study of optimality conditions and the development of globally convergent algorithms for the general bilevel program with a guarantee of suitable stationary conditions. In this area, we mention work based on an implicit function approach (Outrata, Kocvara and Zowe [7]), on a classic approach such as SQP on a one-level reformulation (Scholtes and Stohr [8]) and on smoothing approaches (Fukushima and Pang [9] and Dussault et al. [10]). The second area concerns the development of exact algorithms but is limited to particular subclasses of bilevel programs possessing some combinatorial properties such as linearity or bilinearity which allow the development of efficient approaches. In particular, the property of extremal solution for BLP allowed the development of exact algorithms based on vertex enumeration (see e.g. Candler and Townsley [11], Bialas and Karwan [12] and Tuy, Migdalas and Värbrand [13]). Complementary pivoting approaches (see e.g. Bialas, Karwan and Shaw [14] and Júdice and Faustino [15]) have been proposed on the one-level optimization problem obtained by replacing the second level optimization problem by its optimality conditions. Exploiting the complementarity structure of this one-level reformulation, Bard and Moore [16] and Hansen et al. [5] have proposed branch-and-bound algorithms which appear to be among the most efficient.

As opposed to branch-and-bound methods, the cutting-plane approach is not frequently used in BLP. This is probably due to the lack of convexity of the induced region (formally defined in Sect. 2) and the difficulty of characterizing the optimal solution. Cutting-plane methods found in literature are essentially based on Tuy's concavity cuts [17]. White et al. [18] use the aforementioned cuts in their penalty function approach for solving BLP. For each penalty parameter value, Tuy's method is applied in order to maximize a convex function over a polytope. They show that their approach is not as good as the Bard and Moore [16] branch-and-bound method

but outperforms the algorithm of Bialas and Karwan [12]. Marcotte, Shiquan and Chen [19] proposed the first, stand alone, cutting-plane algorithm for solving BLP with a guarantee of finite termination. The algorithm is based on the duality gap and makes use of Tuy-like cuts [17] in order to shrink the feasible set of the relaxed problem. The algorithm has not been tested on numerical instances and the authors were not expecting good results on medium size instances.

In this paper, we propose a finite and exact branch-and-cut algorithm for solving the linear bilevel programming based on three new classes of valid cuts. We show that these cuts are useful in reducing the relaxation gap of the one-level formulation relaxation typically used in the branch-and-bound approaches. Combined within an enumeration procedure, the resulting algorithm appears to outperform the existing ones.

This paper is organized as follows. Section 2 contains equivalences between BLP and the linear mixed 0–1 programming problem MIP_{0-1}. We recall how to reformulate any BLP instance into a MIP_{0-1} one and vice versa. These equivalences are exploited in Sect. 3 to transpose the concept of MIP_{0-1} Gomory cuts into the BLP framework. The branch-and-cut algorithm is presented in Sect. 4. Finally, numerical experiments are conducted in Sect. 5 on some standard randomly generated test problems. An extended version of this work containing numerical results can be found in [20].

2 Links between BLP and MIP 0–1

In its linear form, the optimistic bilevel programming problem (BLP) is expressed as[1]

$$\begin{aligned} (\mathrm{BLP}) \quad & \max_{x,y} \quad c^{1t}x + d^{1t}y \\ & \text{s.t.} \quad A^1x + B^1y \le b^1, \quad x \ge \mathbb{0}^{n_1}, \\ & \qquad y \in \arg\max_{w} \{d^{2t}w : A^2x + B^2w \le b^2,\ w \ge \mathbb{0}^{n_2}\}, \end{aligned}$$

where: $x, c^1 \in \mathbb{R}^{n_1}$, $y, w, d^i \in \mathbb{R}^{n_2}$, $b^i \in \mathbb{R}^{m_i}$, $A^i \in \mathbb{R}^{m_i \times n_1}$, $B^i \in \mathbb{R}^{m_i \times n_2}$ for $i = 1, 2$. The upper level problem (ULP) and the follower subproblem (FSP(x)) are

$$\begin{aligned} (\mathrm{ULP}) \quad & \max_{x,y} \{c^{1t}x + d^{1t}y : A^1x + B^1y \le b^1,\ x \ge \mathbb{0}^{n_1},\ y \ge \mathbb{0}^{n_2}\}, \\ (\mathrm{FSP}(x)) \quad & \max_{w} \{d^{2t}w : B^2w \le b^2 - A^2x,\ w \ge \mathbb{0}^{n_2}\}. \end{aligned}$$

The polyhedron defined by both upper and lower level linear constraints is called the relaxed feasible region:

$$\Omega = \{(x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} : x \ge \mathbb{0}^{n_1},\ y \ge \mathbb{0}^{n_2},\ A^ix + B^iy \le b^i,\ i = 1, 2\}.$$

[1] We use $\mathbb{1}^p$ and $\mathbb{0}^p$ to respectively denote a vector of 1 and a vector of 0 with dimension $p \in \mathbb{N}$.

We assume throughout this work that the relaxed feasible region is nonempty and bounded. Boundedness is necessary to the proof of Theorem 4.1. The *leader's relaxation problem* LRBLP is obtained from BLP by omitting the second-level objective function:

$$\text{(LRBLP)} \qquad \max_{(x,y)\in\Omega} c^{1t}x + d^{1t}y.$$

A solution $(x, y) \in \Omega$ in which y is optimal for FSP(x) is called *rational*. The set of rational solutions is called the *induced region*. It is a polyhedral set, usually nonconvex which may be disconnected or even empty [21] in presence of upper-level constraints. When the induced region is nonempty, then an optimal solution of BLP exists and at least one is attained at an extreme point of the polytope Ω [21].

We next consider the linear mixed 0–1 programming problem (MIP$_{0-1}$), which may be stated in its general form as

$$\text{(MIP}_{0-1}\text{)} \quad \max_{x,u}\{c^t x + e^t u : Ax + Eu \le b,\ x \ge \mathbb{0}^{n_1},\ u \in \{0,1\}^{n_u}\}.$$

Here, $x, c \in \mathbb{R}^{n_x}$; $e, u \in \mathbb{R}^{n_u}$; $b \in \mathbb{R}^m$; $A \in \mathbb{R}^{m\times n_x}$; $E \in \mathbb{R}^{m\times n_u}$.

This problem may be reformulated as a bilevel program [22]. Conversely, BLP may be reformulated into a MIP$_{0-1}$ [23]. The reformulation is obtained through two transformations. The first one converts the BLP into a one-level nonlinear program by replacing the second-level problem by its KKT optimality conditions,

$$\begin{aligned} &A^2x + B^2y \le b^2, && \lambda^t(b^2 - A^2x - B^2y) = 0, && y \ge \mathbb{0}^{n_2},\\ &B^{2t}\lambda \ge d^2, && y^t(B^{2t}\lambda - d^2) = 0, && \lambda \ge \mathbb{0}^{m_2}, \end{aligned}$$

where $\lambda \in \mathbb{R}^{m_2}$. The second transformation consists in linearizing the nonlinear complementarity constraints by introducing two binary vectors $u \in \mathbb{R}^{m_2}$ and $v \in \mathbb{R}^{n_2}$ and a sufficiently large finite constant $L > 0$ (existence of a finite value for L, under the assumption of a bounded optimal solution, is discussed in [23]) to obtain

$$b^2 - A^2x - B^2y \le L(\mathbb{1}^{m_2} - u), \qquad \lambda \le Lu,\ y \le L(\mathbb{1}^{n_2} - v), \qquad B^{2t}\lambda - d^2 \le Lv.$$

This leads to the following equivalent MIP$_{0-1}$ reformulation of BLP:

$$\begin{aligned} \text{(BLPr)} \quad \max_{x,y,\lambda,u,v} \quad & c^{1t}x + d^{1t}y \\ \text{s.t.} \quad & A^1x + B^1y \le b^1, \\ & A^2x + B^2y \le b^2, \qquad -B^{2t}\lambda \le -d^2, \\ & -A^2x - B^2y + Lu \le L\mathbb{1}^{m_2} - b^2, \qquad \lambda - Lu \le \mathbb{0}^{m_2}, \\ & y + Lv \le L\mathbb{1}^{n_2}, \qquad B^{2t}\lambda - Lv \le d^2, \\ & x \ge \mathbb{0}^{n_1}, \qquad y \ge \mathbb{0}^{n_2}, \qquad \lambda \ge \mathbb{0}^{m_2}, \\ & u \in \{0,1\}^{m_2}, \qquad v \in \{0,1\}^{n_2}. \end{aligned}$$

The interest in these reformulations transcends the simple links between these two problems. Indeed, these equivalences may be extended to algorithms designed for them. More precisely, Audet et al. [22] show that solving any linear mixed 0–1

problem by a classical branch-and-bound algorithm [24] is equivalent to solving its bilevel reformulation with the HJS algorithm [5] as they generate sequences of subproblems which are identical via the reformulation. Various solution techniques used for solving mixed 0–1 programs may be specialized for the bilevel linear programs.

3 Valid Cuts for Bilevel Linear Programming

In this section, we introduce three different types of valid inequalities for the BLP. The first one is based on MIP_{0-1} Gomory cuts and is simply called *Gomory cut*. The two other types are respectively called *simple cuts* and *extended cuts*.

First, recall that Gomory cuts [25, 26] were originally defined for MIP_{0-1}. They may be generated from any valid equality involving integer variables. Consider the following generic equality $\sum_{j\in N} a_j p_j + \sum_{j\in J} g_j r_j = b$, where $b \in \mathbb{R}$, $p_j \in \mathbb{Z}_+$ for $j \in N = \{1, 2, \ldots, n_p\}$ and $r_j \in \mathbb{R}_+$ for $j \in J = \{1, 2, \ldots, n_r\}$, $a_j \in \mathbb{R}$ and $g_j \in \mathbb{R}$ denote respectively coefficients associated to the variables p_j and r_j. Define the sets of indices $J^+ = \{j \in J : g_j > 0\}$ and $J^- = \{j \in J : g_j < 0\}$, and define $f_j = a_j - \lfloor a_j \rfloor, \forall j \in N$ and $f_0 = b - \lfloor b \rfloor$ to be the fractional parts of a_j and b. The corresponding Gomory cut may be expressed as

$$\sum_{j\in N:\, f_j \le f_0} f_j p_j + \frac{f_0}{1-f_0} \sum_{j\in N:\, f_j > f_0} (1-f_j) p_j + \sum_{j\in J^+} g_j r_j - \frac{f_0}{1-f_0} \sum_{j\in J^-} g_j r_j \ge f_0. \tag{3}$$

3.1 Gomory Cuts (GC)

We exploit the mixed reformulation BLPr in order to derive valid cuts for BLP. Consider the following linear relaxation of BLPr where the integrality constraints (corresponding to the complementarity constraints) and binary variables are removed:

$$\begin{array}{lll} \text{(LRP)} & \max\limits_{x,y,\lambda} & c^{1t}x + d^{1t}y \\ & \text{s.t.} & A^1x + B^1y \le b^1, \qquad A^2x + B^2y \le b^2, \qquad -B^{2t}\lambda \le -d^2, \\ & & x \ge \mathbb{0}^{n_1}, \qquad y \ge \mathbb{0}^{n_2}, \qquad \lambda \ge \mathbb{0}^{m_2}. \end{array}$$

Note that the variables can be partitioned into two disjoint sets involving (x, y) and λ. Moreover, the disjoint variable λ does not appear in the objective function. One might be tempted to reduce the size of LRP and simply discard the constraints involving λ. But, we will not do so, since the main result of this paper is to generate valid inequalities that involve primal variables x and y and dual variable λ. Let (x, y, λ) be an optimal solution of the linear program LRP obtained by applying the simplex algorithm. We define two vectors $s, \mu \in \mathbb{R}^{n_2+m_2}$ as following: $s = [b^2 - A^2x - B^2y,\, y]^T$, $\mu = [\lambda,\, B^{2t}\lambda - d^2]^T$. Hence, complementary constraints may be expressed as: $s_i\mu_i = 0, \forall i \in \{1, 2, \ldots, n_2 + m_2\}$. In the unlikely event that

every complementarity constraint is satisfied, the relaxed solution is rational and therefore optimal for BLP. Now suppose that a complementarity constraint is violated, i.e. $\exists i \in \{1, 2, \ldots, n_2 + m_2\}$ such that $s_i \mu_i \neq 0$. The choice of such constraint is discussed in Sect. 4.1. A basic solution of the linear program (LRP) provides an expression for s_i and μ_i,

$$s_i = s_{i0} - \sum_{j \in NB} g_j z_j, \tag{4}$$

$$\mu_i = \mu_{i0} - \sum_{j \in NB} h_j z_j, \tag{5}$$

where z_j represents a generic non-basic variable, NB is the set of indices of non-basic variables, s_{i0} and μ_{i0} are respectively the value of s_i and μ_i and finally g_j and h_j denote respectively the coefficients associated to the variables z_j in (4) and (5). Let ${J_1}^+ = \{j \in NB : g_j > 0\}$ and ${J_2}^+ = \{j \in NB : h_j > 0\}$. The next theorem derives a valid cut for the BLP from the complementarity constraint i.

Theorem 3.1 *If the ith complementarity condition is violated by an optimal basic solution (x, y, λ) of (LRP), then both inequalities*

$$\sum_{j \in {J_1}^+} g_j z_j \geq u_i s_{i0}, \tag{6}$$

$$\sum_{j \in {J_2}^+} h_j z_j \geq (1 - u_i)\mu_{i0}, \tag{7}$$

where $u_i \in \{0, 1\}$, are valid for BLP.

Proof Since the ith complementarity condition is violated, we consider its corresponding constraints

$$s_i \leq L(1 - u_i), \qquad \mu_i \leq L u_i, \quad u_i \text{ binary}.$$

By introducing the nonnegative slack variables ω and ω', one obtains

$$s_i + \omega = L(1 - u_i), \qquad \mu_i + \omega' = L u_i, \quad u_i \text{ binary}.$$

Combining the above equations with the expression of s_i and μ_i in (4) and (5) yields

$$L u_i + \omega - \sum_{j \in NB} g_j z_j = L - s_{i0}, \tag{8}$$

$$L u_i + \sum_{j \in NB} h_j z_j - \omega' = \mu_{i0}. \tag{9}$$

Dividing (8) by $L > 0$ yields

$$u_i + \frac{\omega}{L} - \sum_{j \in NB} \frac{g_j}{L} z_j = 1 - \frac{s_{i0}}{L}. \tag{10}$$

Deriving the Gomory cut (3) on equation (10), we obtain

$$\frac{\omega}{L} - \sum_{j \in J_1^-} \frac{g_j}{L} z_j + \frac{1 - s_{i0}/L}{s_{i0}/L} \sum_{j \in J_1^+} \frac{g_j}{L} z_j \geq 1 - \frac{s_{i0}}{L}, \tag{11}$$

where $J_1^+ = \{j \in NB : g_j > 0\}$ and $J_1^- = \{j \in NB : g_j < 0\}$. Combining (10) and (11), we obtain the first inequality (6). Dividing the equation (9) by $-L < 0$ yields

$$-u_i + \frac{\omega'}{L} - \sum_{j \in NB} \frac{h_j}{L} z_j = -\frac{\mu_{i0}}{L}. \tag{12}$$

Deriving the Gomory cut (3) on equation (12), we obtain

$$\frac{\omega'}{L} - \sum_{j \in J_2^-} \frac{h_j}{L} z_j + \frac{1 - \mu_{i0}/L}{\mu_{i0}/L} \sum_{j \in J_2^+} \frac{h_j}{L} z_j \geq 1 - \frac{\mu_{i0}}{L}, \tag{13}$$

where $J_2^+ = \{j \in NB : h_j > 0\}$ and $J_2^- = \{j \in NB : h_j < 0\}$. Combining (12) and (13), we obtain the inequality (7). □

We will refer to inequalities (6) and (7) as Gomory cuts (GC) for BLP. An important feature of these cuts is that they do not involve the large finite constant L. This constant canceled out when equations (10) and (11) and equations (12) and (13) were combined.

The same Gomory cuts (6) and (7) may be generated directly from the initial form of BLP. This gives a different insight to their constructions. To give an alternative proof of Theorem 3.1, we can consider the general class of linear programming problems

$$\text{(P)} \quad \max_z \{c^t z : Az = b,\ z \geq \mathbb{0}^n\},$$

where $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, c \in \mathbb{R}^n, z \in \mathbb{R}^n$. Let z_i be a nonzero basic variable,

$$z_i = z_{i0} - \sum_{j \in NB} g_j z_j,$$

where g_j is the coefficient associated to z_j. Imposing to z_i to be null yields

$$z_{i0} - \sum_{j \in J^- \cap NB} g_j z_j = \sum_{j \in J^+ \cap NB} g_j z_j.$$

Consequently the following cut:

$$\sum_{j \in J^+ \cap NB} g_j z_j \geq z_{i0}, \tag{14}$$

is a necessary condition to get $z_i = 0$. We present an alternative proof of Theorem 3.1 that does not require the mixed reformulation.

Alternative Proof of Theorem 3.1 LRP is a linear program of type P. Substituting $z_{i0} = s_{i0}$ in (14) gives (15) and $z_{i0} = \mu_{i0}$ gives (16):

$$\sum_{j \in {J_1}^+} g_j z_j \geq s_{i0}, \tag{15}$$

$$\sum_{j \in {J_2}^+} h_j z_j \geq \mu_{i0}. \tag{16}$$

At least one of the inequalities (15) and (16) is valid for the BLP. Hence, by introducing the binary variable u_i the same valid inequalities (6) and (7) are derived. □

3.2 Simple Cuts (SC)

The next theorem generates valid cuts that do not involve binary variables when a complementarity constraint i is violated, i.e. $\mu_{i0} s_{i0} \neq 0$.

Theorem 3.2 *The inequality*

$$\sum_{j \in ({J_1}^+ \cup {J_2}^+)} z_j \max\left\{\frac{g_j}{s_{i0}}, \frac{h_j}{\mu_{i0}}\right\} \geq 1 \tag{17}$$

is valid for BLP. We refer to inequality (17) *as simple cuts* (*SC*) *for BLP.*

Proof As stated in Sect. 3.1, either (15) or (16) is valid. Dividing the first inequality by s_{i0} and the second by μ_{i0}, one obtains the following inequalities:

$$\sum_{j \in {J_1}^+} \frac{g_j}{s_{i0}} z_j \geq 1, \tag{18}$$

$$\sum_{j \in {J_2}^+} \frac{h_j}{\mu_{io}} z_j \geq 1. \tag{19}$$

Equation (17) follows by applying the disjunctive approach [27] to inequalities (18) and (19). □

3.3 Extended Cuts (EC)

Binary variables introduced while generating Gomory cuts may appear in a basic representation of s_i and μ_i, i.e. (4) and (5). These variables could be taken into account in the equation defining the cut. Let J and N denote respectively the sets of indices of the continuous variables z_j and binary variables w_j. At any basic solution, the non-basic variables are equal to zero, and therefore, s_i and μ_i may be expressed

as follows:

$$s_i = s_{i0} - \sum_{j \in NB \cap J} g_j z_j - \sum_{j \in NB \cap N} g'_j w_j, \tag{20}$$

$$\mu_i = \mu_{i0} - \sum_{j \in NB \cap J} h_j z_j - \sum_{j \in NB \cap N} h'_j w_j, \tag{21}$$

where g'_j and h'_j denote the coefficients associated to the variables w_j in (20) and (21). The next theorem gives a valid cut for BLP from the complementarity constraint i.

Theorem 3.3 *The inequalities*

$$\sum_{j \in J_1{}^+ \cap J} g_j z_j + \sum_{\substack{j \in N \\ g'_j > 0,\, g'_j < s_{i0}}} g'_j w_j + \sum_{\substack{j \in N \\ g'_j > 0,\, g'_j \geq s_{i0}}} s_{i0} w_j \geq u_i s_{i0}, \tag{22}$$

$$\sum_{j \in J_2{}^+ \cap J} h_j z_j + \sum_{\substack{j \in N \\ h'_j > 0,\, h'_j < \mu_{i0}}} h'_j w_j + \sum_{\substack{j \in N \\ h'_j > 0,\, h'_j \geq \mu_{i0}}} \mu_{i0} w_j \geq (1 - u_i)\mu_{i0}, \tag{23}$$

where $u_i \in \{0, 1\}$, *are valid for BLP.*

Proof By deriving the primal cut of GC (6), we obtain

$$\sum_{j \in J_1{}^+ \cap J} g_j z_j + \sum_{j \in N : g'_j > 0} g'_j w_j \geq u_i s_{i0},$$

with $w_j = 0$ or $w_j = 1$. Let $j \in N$ such that $g'_j > s_{i0}$. Observe that if $w_j = 0$, the inequality obtained by replacing g'_j by s_{i0} in (6) remains valid; and if $w_j = 1$, then $w_j \geq u_i$, hence $s_{i0} w_j \geq s_{i0} u_i$ and the inequality obtained by replacing g'_j by s_{i0} in (6) is also valid. In conclusion, the inequality (22) is valid. In the same way, we prove the validity of (23) by deriving the dual cut of GC (7). □

We will refer to inequalities (22) and (23) as extended cuts (EC) for BLP.

3.4 Relative Depths of Cuts

We propose to compare the depths of the three cuts: the Gomory cuts, the simple cuts and the extended cuts. The comparison is based on the next definition.

Definition 3.1 Let Π_1: $\sum_i {\pi_i}^1 z_i \geq {\pi_0}^1$ and Π_2: $\sum_i {\pi_i}^2 z_i \geq {\pi_0}^2$ be two valid cuts. Π_1 is deeper than Π_2 if ${\pi_i}^1 \leq {\pi_i}^2, \forall i$ and ${\pi_0}^1 \geq {\pi_0}^2$.

Using the previous definition, the following proposition compares the depth of an extended cut to a Gomory cut.

Proposition 3.1 *The extended cuts (EC) are deeper than the Gomory cuts (GC): more precisely (22) is deeper than (6) and (23) is deeper than (7).*

Proof Let π_j and π'_j denote respectively the coefficients associated to the binary variable w_j in (22) and (23). The result follows by observing that $\pi_j = \min\{g'_j, s_{io}\} \leq g'_j$ and $\pi'_j = \min\{h'_j, \mu_{io}\} \leq h'_j$. □

Simple examples [20] show that there is no systematic dominance for the other pairs of cuts. Nevertheless, the simple cut has the advantage of not introducing additional variables.

4 Branch-and-Cut Algorithm for BLP

In this section, we propose a branch-and-cut algorithm for BLP using the cuts described in Sect. 3. The algorithm is composed of two phases. The first one (Sect. 4.1) is a preprocessing phase whose goal is to improve the value of the upper bound on the objective function value. It is an iterative step where at each iteration a valid inequality is generated and added to the linear relaxation. After a maximal number of iterations, the second phase—an enumeration phase (Sect. 4.2)—is triggered in order to ensure finite and exact convergence of the algorithm. A full description of the algorithm is given in Sect. 4.3. An academic example is provided in [20] to illustrate our approach, and to compare the three types of cuts.

4.1 Cutting-Plane Phase

We first propose to perform an iterative preprocessing phase which generates valid cuts of one of the three types presented in Sect. 3. Each iteration is divided into four steps: The application of tests [28] to conclude optimality and to terminate the algorithm, the selection of a complementarity constraint, the introduction of cuts, and finally the selection of the active ones (through a cleaning procedure).

Application of Tests

A test consists in verifying some properties of the current problem in hopes of concluding optimality or of fixing some variables. We use the terminology presented in Hansen et al. [28] for classifying tests in branch-and-bound framework.

The procedure first examines the set of complementarity constraints by applying a rationality and a conditional test. The *rationality test* checks the rationality of the LRP optimal solution. In the event that all complementary conditions are satisfied, then the test concludes that the LRP solution is optimal for the current problem. *The conditional test* is applied to all violated complementarity constraints. This test relies on the following proposition:

Proposition 4.1 *Let i be the index of a violated complementarity constraint. Consider the expression of s_i (4) and μ_i (5) obtained after solving LRP with the*

simplex algorithm:

(i) *If* $g_j \leq 0, \forall j \in NB$, *then* $\mu_i = 0$ *at optimality.*
(ii) *If* $h_j \leq 0, \forall j \in NB$, *then* $s_i = 0$ *at optimality.*

Proof Since $s_i \mu_i \neq 0$ then $s_{i0} > 0$ and $\mu_{i0} > 0$. As stated in Sect. 3.1, the condition $\sum_{j \in J_1^+ \cap NB} g_j z_j \geq s_{i0}$ is necessary to get $s_i = 0$. In particular, if $g_j \leq 0, \forall j \in NB$, then $s_i \neq 0$ at optimality and hence, $\mu_i = 0$ at optimality.

The second part of the proposition can be shown in the same way. □

In summary, the conditional test examines the expressions of s_i and μ_i. Whenever all coefficients associated to nonbasic variables in the expression of s_i (respectively μ_i) are non positive then μ_i (respectively s_i) is fixed at 0.

Selection of a Complementarity Constraint

Let V denotes the set of indices of violated complementarity constraints, and S denotes the set of indices of complementarity constraints from which cuts were previously generated. The efficiency of the cuts procedure relies mainly on the selection of the constraint $i \in V$ from which the cut is generated. We now enumerate different selection rules used in our algorithm:

(S1) Select an index $i \in \arg\max_{j \in V} (\mu_j s_j)$ (same rule as in [16]).
(S2) If $\min_{j \in S} \{|\frac{1}{2} - u_j|\} = \frac{1}{2}$ then use the rule S1, otherwise select $i \in \arg\min_{j \in S} \{|\frac{1}{2} - u_j|\}$.
(S3) Select the first violated complementarity constraint: $i = \min_j \{j \in V\}$.
(S4) Select an index $i \in \arg\max_{j \in V} \{\min(\mu_j, s_j)\}$.
(S5) Select $i = \min_j \{j \in V \setminus S\}$. If there are none, use S3.
(S6) Select the index i with largest penalty see [20] for details.

Introduction of Cuts

The introduction of cuts consists in adding a cut corresponding to the complementarity constraint i selected by a pre-defined rule. The introduction of both Gomory and extended cuts also relies on the value of the binary variable u_i and on the constant L appearing the reformulation (BLPr). The scheme for generating the cuts is:

1. If no cuts were previously generated from the complementarity constraint i, we introduce a variable u_i and generate the corresponding Gomory (or extended) cut.
2. Otherwise, the value of u_i determines the type of cut: If $0 < u_i < 1$, we derive the corresponding Gomory (or extended) cut. If $u_i = 1$, we reintroduce the constraint: $s_i \leq L(1 - u_i)$. If $u_i = 0$, we reintroduce the constraint: $\mu_i \leq L u_i$.
   The purpose of reintroducing constraints is to add strong cuts when the Gomory ones become inefficient.

Cleaning Procedure

This procedure is necessary when the cutting-plane phase of the algorithm fails to conclude optimality, and thus, the enumeration phase must be invoked. After generating cuts, the size of the relaxed problem LRP is reduced by removing every cut whose corresponding slack variables are basic. This procedure aims at keeping the LRP size small, thus reducing the computational effort in each node of the enumeration tree.

4.2 Branch-and-Bound Phase

The second phase consists in the branch-and-bound algorithm (called HJS) proposed by Hansen, Jaumard and Savard [5] without computing penalties and without deriving monotony relations. This method uses two relaxations of the current subproblem: the *leader relaxation* LRP which includes the valid cuts generated in preprocessing phase and the *follower relaxation* FRP($\bar{x}$), which corresponds to the second level problem with x fixed to $\bar{x}$. HJS also considers the *follower's subproblem* of the initial problem FSP($\bar{x}$) described in Sect. 1.

Branching is done on the tightness of constraints in the follower's subproblem by associating a boolean variable α_i to the ith complementarity condition. Setting α_i to 1 translates in fixing the ith second level primal constraint to equality in both LRP and FRP. Setting α_i to 0 implies fixing the corresponding dual constraint to equality. If LRP contains a binary variable u_i, then setting α_i to 1 (respectively 0) implies setting u_i to 1 (respectively 0). Note that the α variables are not explicitly added to the problem but are used as flags by the algorithm to manage the enumeration tree. Any of the following conditions is sufficient to prune the current node:

1. Feasibility Test: Either LRP or the FRP has no feasible solution.
2. Optimality Test: The LRP optimal value is less than or equal to the incumbent value.
3. Resolution Test: The optimal solution $(\bar{x}, \bar{y})$ of LRP is rational.

As in the preprocessing phase, numerous branching rules are used:

(B1) Product Rule: Select α_i in such a way that $i \in \arg\max_j \{s_j \mu_j\}$ where the values of s_j and μ_j are obtained after solving LRP. This rule does not use the information provided by FRP($\bar{x}$), such as the objective function of the follower's problem. This is the rule used by Bard and Moore [16].

(B2) HJS Rule: Same as the product rule, except that the value of μ_j is obtained by the solution of the dual problem of FRP($\bar{x}$). This rule exploits informations available in both LRP and FRP problems.

(B3) Max-Min Rule: Select α_i in such a way that $i \in \arg\max_j \{\min\{s_j, \mu_j\}\}$ where the value of s_j is obtained after solving LRP and μ_j is given by the solution of the dual problem of FRP($\bar{x}$).

4.3 General Scheme of the Algorithm

Based on the two procedures detailed in Sects. 4.1 and 4.2, we propose a branch-and-cut algorithm (CBB) to solve BLP under the assumption that the relaxed feasible

region is nonempty and bounded. All linear programs are solved by the Simplex method, and therefore the solutions are vertices. The algorithm is stated as follows:

Phase I: Cutting-Plane Generation
For $it = 1, 2, \ldots,$ Max (where Max is the maximum number of iterations allowed)

Step 1. *Rationality Test* Let (x, y, λ) be an optimal solution of LRP.
If all complementarity constraints are satisfied then end: (x, y) is optimal.
Step 2. *Conditional Test* Apply the conditional test to all complementarity constraints. If any variable was fixed then go to step 1.
Step 3. *Selection of a Complementarity Constraint* Select a complementarity constraint i using a specific selection criterion.
Step 4. *Introduction of Cuts* Add the cut corresponding to the complementarity constraint i.

Phase II: Enumeration (Simplification of HJS [5])
Consider the LRP with the valid cuts generated in phase I and apply the HJS algorithm [5] without computing penalties and without deriving monotony relations. Further details for PHASE II are given in [20].

Note that in the first iteration of the cutting-plane phase, primal and dual variables are not coupled in LRP. Therefore, in the first iteration, the rationality test uses $\lambda = \lambda_0$ where λ_0 is the dual optimal solution of FSP(0).

Theorem 4.1 *Under the assumption of a bounded and nonempty relaxed feasible region, the CBB algorithm solves the BLP in finite time.*

Proof The CBB adds a preprocessing step that terminates in finite time to the HJS algorithm, which was shown to be finite [5]. □

5 Numerical Results

The algorithm CBB was coded in C and uses the CPLEX 8.1 library to solve linear programs. Computational experiments are carried out on an ULTRA 60 station under SOLARIS 2.7-05. In order to test the different algorithmic criteria and its effectiveness, a series of problems was randomly generated with the same sizes as in Hansen et al. [5] with a density of 8% (the proportion of nonzero matrix entries). The generating method is inspired by the one detailed in Audet et al. [29] for bilinear programming. All coefficients are randomly chosen with a uniform distribution. Each component of the vectors b^1 and b^2 is chosen between -40 and 40. Those of A^1 and B^2 are chosen between -20 and 20. The elements of the matrices A^2 and B^1 are taken from the interval $[0, 20]$. The coefficients of the vector c^1 belong to the interval $[-10, 10]$ and those of d^1 belong to $[10, 20]$. The elements of the vector d^2 are chosen between 0 and 10. Finally, the additional second level constraint $\sum_{j\in n_1} x_j + \sum_{j\in n_2} y_j \leq n_1 + n_2$ is added to obtain a bounded relaxed problem.

The constant L required by the Gomory and Extended cuts was fixed to 100. The entries in the following tables are mean values μ and standard deviation σ for 10 randomly generated problems.

A first class of tests aimed at streamlining the cutting-plane phase without the cleaning procedure. In each series of experiments, the quality of the upper bound is assessed by computing the relative decrease of the objective function $\delta = (F_n - F_{\text{opt}})/(F_{\text{ini}} - F_{\text{opt}})$ where F_n is the value of the objective function after n cuts iterations, F_{ini} is the initial value and F_{opt} is the optimal value. Results are reported in [20]. Note that the results obtained by applying extended cuts are the same as those obtained with Gomory cuts. Results show that the quality of the upper bound is sensitive to the selection criterion. The best results are obtained by applying S1, S2 or S6. The first rule S1 was adopted in the remaining experiments since it requires less computational effort. In all series of tests the simple cuts outperforms the Gomory cuts. This observation is illustrated in the top part of Fig. 1 which reports the mean decrease of the objective function δ versus the number of cutting-plane iterations. There is a strict monotonic decrease in the average upper bound as the number of cuts increases. The decrease in the objective function value reaches 24% using Gomory cuts and 36% using simple cuts. This decrease leads to a reduction of the number of nodes in branch-and-bound trees [20]. As expected, a significant reduction of the number of nodes in branch-and-bound tree is obtained when adding cuts. The bottom part of Fig. 1 presents the mean size of branch-and-bound tree in function of the cut generation iteration number. The average reduction in the number of nodes reaches 43% using simple cuts and 31% using Gomory cuts. These results are the first premises of the effectiveness of our algorithm and introduce the second class of experiments in which we study the entire branch-and-cut algorithm by computing the number of the branching nodes (nds) and the branching time (T(s), in second) since the time of cutting-plane phase never exceeds 0.1 s. In these series of experiments, the cleaning procedure is activated in order to reduce the branching time. Results show an extreme sensitivity of the number of nodes and cpu time to the selected branching rule. As in [5], the most efficient branching rule is HJS rule (see [20] for numerical results). The last series of experiments aims at evaluating the effectiveness of our algorithm. Table 1 compares the results of the pure branch-and-bound algorithm B&B with two versions of our branch-and-cut algorithm (CBB) using HJS rule with $\lfloor \frac{m_2}{3} \rfloor$ Simple cuts iterations. The first version (CBB1) does not call the cleaning procedure but the second one does. In addition to the number of nodes, the branching time and the objective function decrease, the table displays improvements rate (%) for comparison with pure branch-and-bound. These numerical experiments suggest that the branch-and-cut algorithm CBB(2) outperforms pure branch-and-bound in terms of number of nodes explored and computing times on all series of problems excepted the series 7 and 8 where B&B takes slightly less time than CBB(2). Moreover, the cleaning procedure appears to enhance the computational results. This procedure reduces considerably the computational efforts: The mean improvement of computational time shifts from 26% to 31% in comparison to pure branch-and-bound.

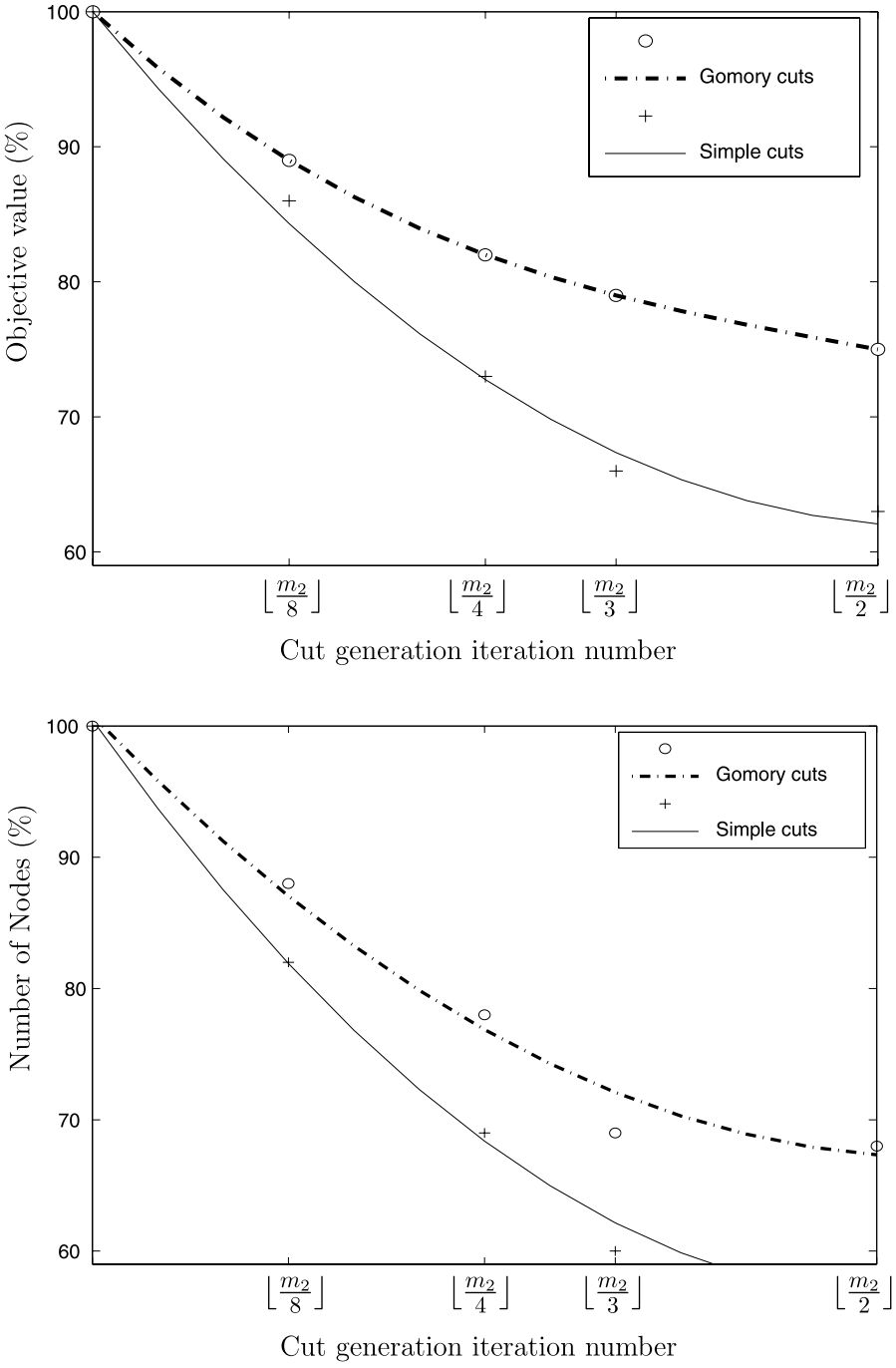

Fig. 1 Average decrease in the objective function value of LRP and of the number of nodes generated by the enumeration scheme

Table 1 Comparison between B&B, CBB1 and CBB2

Algorithm			B&B		CBB(1)			CBB(2)		
Problem			Nodes	t(s)	$\delta(\%)$	Nodes	t(s)	$\delta(\%)$	Nodes	t(s)
1	$n_1 = 35, n_2 = 35$	$\mu\|\sigma$	2851|3588	3.8|5.0	30.5|23.9	1868|3093	2.9|5.02	21.6|13.1	2033|3027	3.0|4.7
	$m_1 = 7, m_2 = 21$	%	–	–	–	34.4%	23.6%	–	28.6%	21.0%
2	$n_1 = 60, n_2 = 30$	$\mu\|\sigma$	3422|7789	6.3|14.4	43.8|20.8	765|738	1.8|1.8	43.8|20.8	766|738	1.7|1.7
	$m_1 = 9, m_2 = 27$	%	–	–	–	77.6%	71.8%	–	77.6%	73.4%
3	$n_1 = 45, n_2 = 45$	$\mu\|\sigma$	16513|15168	33.0|31.7	25.9|12.7	10049|5010	22.9|11.8	25.9|12.7	10043|5002	21.8|11.3
	$m_1 = 9, m_2 = 27$	%	–	–	–	39.1%	30.6%	–	39.1%	33.9%
4	$n_1 = 70, n_2 = 30$	$\mu\|\sigma$	2860|2617	6.4|5.9	47.1|0.26	1266|1722	3.5|4.7	36.5|0.3	1865|1982	4.9|5.2
	$m_1 = 10, m_2 = 30$	%	–	–	–	55.7%	45.3%	–	34.7%	23.4%
5	$n_1 = 60, n_2 = 40$	$\mu\|\sigma$	17289|25508	40.3|61.2	46.2|18.3	6955|9946	18.5|24.4	46.2|18.3	6812|9537	16.7|21.0
	$m_1 = 10, m_2 = 30$	%	–	–	–	59.7%	54.0%	–	60.6%	58.5%
6	$n_1 = 50, n_2 = 50$	$\mu\|\sigma$	23770|19001	55.1|45.5	28.4|17.1	10808|10798	30.3|31.6	25.4|19.3	10668|10812	27.6|29.5
	$m_1 = 10, m_2 = 30$	%	–	–	–	54.5%	45.0%	–	55.1%	49.9%
7	$n_1 = 70, n_2 = 50$	$\mu\|\sigma$	77503|77199	234|226	22.6|14.9	66857|44816	252|163	22.6|15.0	67653|44645	234|154
	$m_1 = 12, m_2 = 36$	%	–	–	–	13.7%	−7.7%	–	12.7%	−0.3%
8	$n_1 = 60, n_2 = 60$	$\mu\|\sigma$	146353|103268	467|337.4	18|16.5	135063|97953	550|411	18|16.5	135015|97949	488|369
	$m_1 = 12, m_2 = 36$	%	–	–	–	7.7%	−17.7%	–	7%	−4.5%
9	$n_1 = 70, n_2 = 60$	$\mu\|\sigma$	269460|434465	1002|1641	41|17	227126|311061	1072|1444	35|19	186167|302049	752|1204
	$m_1 = 13, m_2 = 39$	%	–	–	–	15.7%	−7.0%	–	30.9%	24.9%

6 Conclusions

Based on the links that unify the bilevel linear programming BLP and mixed linear programming MIP_{0-1}, this paper presents different valid inequalities for the BLP. A branch-and-bound algorithm (CBB) based on the aforementioned cuts in a preprocessing phase is proposed. Tested in a series of randomly generated problems, the CBB outperforms the pure branch-and-bound algorithm in terms of computing times, reduction of gap between upper and lower bounds on the objectives and in terms of number of nodes explored by the B&B.

References

1. Van Stackelberg, H.: The Theory of Market Economy. Oxford University Press, Oxford (1952)
2. Luo, Z.Q., Pang, J.S., Ralph, D.: Mathematical Programs with Equilibrium Constraints. Cambridge University Press, Cambridge (1996)
3. Loridan, P., Morgan, J.: ϵ-Regularized two-level optimization problems: approximation and existence results. In: Fifth French-German Conference on Optimization. Lecture Notes in Mathematics, vol. 1405, pp. 99–113. Springer, Berlin (1989)
4. Jeroslow, R.: The polynomial hierarchy and a simple model for competitive analysis. Math. Program. **32**, 146–164 (1985)
5. Hansen, P., Jaumard, B., Savard, G.: New branch-and-bound rules for linear bilevel programming. SIAM J. Sci. Stat. Comput. **13**, 1194–1217 (1992)
6. Vicente, L.N., Savard, G., Júdice, J.J.: Discrete linear bilevel programming problem. J. Optim. Theory Appl. **89**, 597–614 (1996)
7. Outrata, J.V., Kocvara, M., Zowe, J.: Nonsmooth Approach to Optimization Problems with Equilibrium Constraints. Kluwer Academic, Dordrecht (1998)
8. Scholtes, S., Stohr, M.: Exact penalization of mathematical programs with equilibrium constraints. SIAM J. Control Optim. **37**, 617–652 (1999)
9. Fukushima, M., Pang, J.S.: Complementarity constraint qualifications and simplified b-stationarity conditions for mathematical programs with equilibrium constraints. Comput. Optim. Appl. **13**, 111–136 (1999)
10. Dussault, J.-P., Marcotte, P., Roch, S., Savard, G.: A smoothing heuristic for a class of bilinear bilevel programs. Cahier du GERAD, GERAD (2004)
11. Candler, W., Townsley, R.: A linear two-level programming problem. Comput. Oper. Res. **9**, 59–76 (1982)
12. Bialas, W., Karwan, M.: Two-level linear programming. Manag. Sci. **30**, 1004–1020 (1984)
13. Tuy, H., Migdalas, A., Värbrand, P.: A global optimization approach for the linear two-level program. J. Glob. Optim. **3**, 1–23 (1993)
14. Bialas, W., Karwan, M., Shaw, J.: A parametric complementarity pivot approach for two-level linear programming. Operations Research Program Report 80-2, State University of New York at Buffalo (1980)
15. Júdice, J., Faustino, A.: A sequential LCP method for bilevel linear programming. Ann. Oper. Res. **34**, 89–106 (1992)
16. Bard, J.F., Moore, J.: A branch-and-bound algorithm for the bilevel programming problem. SIAM J. Sci. Stat. Comp. **11**, 281–292 (1990)
17. Tuy, H.: Concave programming under linear constraints. Sov. Math. **5**, 1437–1440 (1964)
18. White, D., Anandalingam, G.: A penalty function approach for solving bilevel linear programs. J. Glob. Optim. **3**, 397–419 (1993)
19. Marcotte, P., Shiquan, W., Chen, Y.: A cutting-plane algorithm for the linear bilevel programming problem. CRT Report 925 (1993)
20. Audet, C., Savard, G., Zghal, W.: New branch-and-cut algorithm for bilevel linear programming. GERAD Report G-2004-13, GERAD (2004)
21. Savard, G.: Contributions à la programmation mathématiques à deux niveaux. PhD Thesis, École Polytechnique de Montréal (1988)

22. Audet, C., Hansen, P., Jaumard, B., Savard, G.: Links between linear bilevel and mixed 0–1 programming problems. J. Optim. Theory Appl. **93**, 273–300 (1997)
23. Fortuny-Amat, J., McCarl, B.: A representation and economic interpretation of a two-level programming problem. J. Oper. Res. Soc. **32**, 783–792 (1981)
24. Beale, E.M.L., Small, R.E.: Mixed integer programming by a branch-and-bound technique. Proc. 3rd IFIP Congress **2**, 450–451 (1965)
25. Gomory, R.: An algorithm for the mixed integer problem. Technical Report RM-2537, Rand Corporation (1960)
26. Nemhauser, G.L., Wolsey, L.A.: Integer Programming and Combinatorial Optimization. Wiley–Interscience, New York (1988)
27. Wolsey, L.A.: Integer Programming. Collection Wiley–Interscience Series in Discrete Mathematics and Optimization. Wiley–Interscience, Hoboken (1998)
28. Hansen, P., Jaumard, B., Lu, S.H.: A framework for algorithms in globally optimal design. J. Mech. Trans. Autom. Des. **111**, 353–360 (1989)
29. Audet, C., Hansen, P., Jaumard, B., Savard, G.: A symmetrical linear maxmin approach to disjoint bilinear programming. Math. Program. **85**, 573–592 (1999)