JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 93, No. 2, pp. 273-300, MAY 1997

# Links Between Linear Bilevel and Mixed 0-1 Programming Problems[1]

C. AUDET,[2] P. HANSEN,[3] B. JAUMARD,[4] AND G. SAVARD[5]

Communicated by J. Abadie

**Abstract.** We study links between the linear bilevel and linear mixed 0-1 programming problems. A new reformulation of the linear mixed 0-1 programming problem into a linear bilevel programming one, which does not require the introduction of a large finite constant, is presented. We show that solving a linear mixed 0-1 problem by a classical branch-and-bound algorithm is equivalent in a strong sense to solving its bilevel reformulation by a bilevel branch-and-bound algorithm. The mixed 0-1 algorithm is embedded in the bilevel algorithm through the aforementioned reformulation; i.e., when applied to any mixed 0-1 instance and its bilevel reformulation, they generate sequences of subproblems which are identical via the reformulation.

**Key Words.** Bilevel programming, mixed 0-1 programming, embedded algorithms, branch-and-bound methods.

## 1. Introduction

It is well known that any two NP-complete problems $P_A$ and $P_B$ are equivalent, in the sense that any instance $A$ of $P_A$ can be reformulated as

[1]This research was done during visits of the first and third authors to the Université de Fribourg, and of the second author to the École Polytechnique Fédérale de Lausanne, Switzerland.

[2]PhD Student, École Polytechnique de Montréal, Département de Mathématiques et de Génie Industriel, Montréal, Québec, Canada. Work of this author was supported by a NSERC fellowship.

[3]Professor, École des Hautes Études Commerciales and GERAD, Département MQ, Montréal, Québec, Canada. Work of this author was supported by FCAR Grant 95ER1048, NSERC Grant to HEC, and NSERC Grant GP0105574.

[4]Associate Professor, École Polytechnique de Montréal and GERAD, Département de Mathématiques et de Génie Industriel, Montréal, Québec, Canada. Work of this author was supported by FCAR Grant 95ER1048, NSERC Grant GP0036426, and a FRSQ fellowship.

[5]Associate Professor, École Polytechnique de Montréal and GERAD, Département de Mathématiques et de Génie Industriel, Montréal, Québec, Canada. Work of this author was supported by NSERC Grant OGP0046405 and FCAR Grant 93ER0141.

an instance $B(A)$ of $P_B$, with a polynomial increase in size. Moreover, solving both $P_A$ and $P_B$ requires exponential time in the worst case, unless $P = NP$.

However, we observe in practice that, within the class of NP-complete and NP-hard problems, some problems are much more difficult to solve than others (this is also true for the average case, but such a case is often difficult to define in a realistic way, and to analyze). Also, reformulating an instance $A$ as $B(A)$, one may increase considerably the time needed to solve it with an algorithm $\mathrm{AL}(P_B)$ for $P_B$, or may leave it the same as with an algorithm $\mathrm{AL}(P_A)$ for $P_A$. This suggests that further insight into the practical difficulty of solving NP-complete and NP-hard problems might be obtained by examining their reformulations as well as the links between the algorithms used to solve them. A central question is then: when does an algorithm $\mathrm{AL}(P_B)$ contain an algorithm $\mathrm{AL}(P_A)$ in the sense that, when applied to $B(A)$, it performs the same or equivalent steps as algorithm $\mathrm{AL}(P_A)$ for $A$? This question can be made precise with the concept of embedded algorithm, later defined precisely in this paper. If $\mathrm{AL}(P_A)$ is embedded in $\mathrm{AL}(P_B)$ through the mapping $B(\cdot): P_A \rightarrow P_B$, then $P_B$ appears to be at least as difficult as $P_A$. In this paper, we study reformulation and embedding of algorithms for mixed 0–1 linear programming (MIP) and linear bilevel programming (BLP) with or without some 0–1 variables at the first level.

Vicente et al. (Ref. 1) reformulate different classes of mixed 0–1 bilevel linear programs into linear multilevel programming problems. We present new reformulations along the same lines, which do not require the introduction of a large finite constant. Conversely, Fortuny-Amat and McCarl (Ref. 2) implicitly propose a reformulation of BLP as a generalized linear complementary problem (GLCP), which is reformulated in turn as a MIP. This two-stage reformulation has its converse: Júdice and Mitra (Ref. 3) reformulate MIP as a GLCP, and we present a reformulation of this last problem as a BLP.

Many algorithms have been proposed to solve BLP. Among the most efficient are those of Hansen et al. (Ref. 4), Júdice and Faustino (Ref. 5), and Bard and Moore (Ref. 6). The two last references address the GLCP reformulation. Reference 5 solves a sequence of linear complementarity problems which converges to an $\epsilon$-optimal solution and uses a branch-and-bound scheme. Hansen et al. designed a branch-and-bound algorithm for the BLP formulations, which is more compact. It extends the Bard and Moore algorithm by using penalties and necessary optimality conditions expressed as logical relations.

The main result of this paper is the definition, illustration, and discussion of the potential use of the concept of embedded algorithm. In particular, we show that the classical branch-and-bound algorithm of Beale and Small for MIP (Ref. 7) is embedded in the Hansen, Jaumard, and Savard algorithm for BLP through a given reformulation. The study of embedding between algorithms can also lead to generalization or specialization of tests from one algorithm to another.

The paper is organized as follows. The BLP and MIP are presented in Section 2. Reformulations are studied in Section 3. In Section 4, algorithms for BLP and MIP are presented and compared. Results are discussed and illustrated on a small example.

## 2. Formulations

**2.1. Bilevel Programming Problem.** Bilevel programming problems model situations in which two decision makers, the leader and the follower, each controls part of the variables. Both have their own objective function and constraints. As in Stackelberg games, the leader makes his decisions first, anticipating those of the follower. The follower decisions depend on the leader decisions, but are not restrained by the leader constraints. It is prohibited for the leader to make decisions that would violate his constraints, when combined with the follower decisions. In the case where the follower is indifferent about multiple optimal decisions, cooperation between both levels is modeled by letting the leader choose among them. The reader can refer to Ben-Ayed (Ref. 8) for a recent survey of BLP and to Vicente and Calamai (Ref. 9) for a bibliographical review of bilevel programming. In its general form, the bilevel linear programming problem can be formally stated as follows:

$$\begin{aligned}
\text{(BLP)}\quad \max_{x,y}\quad & c^{1\prime}x + d^{1\prime}y,\\
\text{s.t.}\quad & A^1x + B^1y \leq b^1,\\
& x \geq 0,\\
& y \in \arg\max_{y}\ c^{2\prime}x + d^{2\prime}y,\\
& \qquad\quad \text{s.t.}\quad A^2x + B^2y \leq b^2,\\
& \qquad\qquad\qquad y \geq 0,
\end{aligned}$$

where

$$\begin{aligned}
&c^1, c^2, x \in \mathbb{R}^{n_x}, && A^1 \in \mathbb{R}^{m_1\times n_x}, && A^2 \in \mathbb{R}^{m_2\times n_x}, && b^1 \in \mathbb{R}^{m_1},\\
&d^1, d^2, y \in \mathbb{R}^{n_y}, && B^1 \in \mathbb{R}^{m_1\times n_y}, && B^2 \in \mathbb{R}^{m_2\times n_y}, && b^2 \in \mathbb{R}^{m_2}.
\end{aligned}$$

An important particular case of BLP is the linear maxmin problem (LMM) [see, e.g., Falk (Ref. 10), Audet et al. (Ref. 11)], in which the objective function of the follower is the opposite of that of the leader, and in which the second-level variable $y$ does not appear in the first-level constraints. Jeroslow (Ref. 12), Ben-Ayed and Blair (Ref. 13), and Bard (Ref. 14) show that a BLP is NP-hard. Hansen, Jaumard, and Savard (Ref. 4) show that LMM, and hence BLP, are strongly NP-hard. This implies that no fully polynomial approximation scheme exists for a BLP unless $P = NP$.

The polyhedron defined by the first-level and second-level constraints, called the relaxed feasible region, is assumed to be nonempty and bounded, i.e., a nonempty polytope. A point $(\hat{x}, \hat{y})$ of the relaxed feasible region such that $\hat{y}$ is an optimal solution of the second-level problem is called rational.

Several authors consider a BLP without first-level constraints. It has been shown by Bialas and Karwan (Ref. 15), Candler and Townsley (Ref. 16), and Benson (Ref. 17) that the rational set is then a connected union of faces of the second-level polytope.

Savard (Ref. 18) observes that this result does not hold anymore when first-level constraints contain second-level variables. Consider the instance in which $y = 0 \in \mathbb{R}$ appears in the first-level constraints, and where the second-level problem is

$$\begin{aligned} y \in \arg\max_{y} \quad & y, \\ \text{s.t.} \quad & y \leq x, \\ & y \leq 1 - x. \end{aligned}$$

A solution $(\hat{x}, \hat{y})$ is rational if and only if $\hat{x}$ belongs to the discrete set $\{0, 1\}$ and $\hat{y} = 0$.

**2.2. Mixed 0–1 Programming Problem.** The previous example suggests that there are similarities between bilevel programming and mixed integer programming. The linear mixed 0–1 programming problem is formulated as follows:

$$\begin{aligned} \text{(MIP)} \quad \max_{x,u} \quad & c^t x + e^t u, \\ \text{s.t.} \quad & Ax + Eu \leq b, \\ & u \in \{0, 1\}^{n_u}, \\ & x \geq 0, \end{aligned}$$

where

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

A natural generalization of both BLP and MIP is the bilevel linear programming problem with mixed 0-1 variables at the first level:

$$
\begin{aligned}
\text{(MIBLP)}\quad \max_{x,y,u}\quad & c^{1t}x+d^{1t}y+e^{1t}u,\\
\text{s.t.}\quad & A^1x+B^1y+E^1u\le b^1,\\
& u\in\{0,1\}^{n_u},\\
& x\ge 0,\\
& y\in\arg\max_y\quad c^{2t}x+d^{2t}y+e^{2t}u,\\
& \qquad\quad \text{s.t.}\quad A^2x+B^2y+E^2u\le b^2,\\
& \qquad\qquad\quad\ y\ge 0,
\end{aligned}
$$

where

$$
\begin{array}{llll}
x,c^1,c^2\in\mathbb{R}^{n_x}, & A^1\in\mathbb{R}^{m_1\times n_x}, & A^2\in\mathbb{R}^{m_2\times n_x}, & b^1\in\mathbb{R}^{m_1},\\
y,d^1,d^2\in\mathbb{R}^{n_y}, & B^1\in\mathbb{R}^{m_1\times n_y}, & B^2\in\mathbb{R}^{m_2\times n_y}, & b^2\in\mathbb{R}^{m_2},\\
u,e^1,e^2\in\mathbb{R}^{n_u}, & E^1\in\mathbb{R}^{m_1\times n_u}, & E^2\in\mathbb{R}^{m_2\times n_u}. &
\end{array}
$$

Wen and Yang (Ref. 19) propose a branch-and-bound algorithm to solve MIBLP when there are no continuous first-level variables and the set of first-level constraints is restricted to $u\in\{0,1\}^{n_u}$. Audet et al. (Ref. 20) show how to sharpen their upper bound in the original version of this paper.

## 3. Reformulations

Recall that any NP-complete problem can be polynomially reduced to any other NP-complete problem. NP-hard problems define a wider class. An NP-hard problem is at least as difficult as any NP-complete problem, since by definition, any NP-complete problem can be polynomially reduced to an NP-hard problem. However, an NP-hard problem might not reduce to an NP-complete problem.

A polynomial time Turing reduction [see, e.g., Garey and Johnson (Ref. 21)] from problem $P_A$ to problem $P_B$ is an algorithm AL($P_A$) that solves P$_A$ in polynomial time, by consulting a polynomial time oracle for $P_B$. Such an algorithm is divided into three phases. Given an instance $A$ of $P_A$, it first reformulates $A$ into $B(A)$, an instance of $P_B$. The second phase consists in consulting the oracle in order to solve $B(A)$. Finally, this solution is transformed into a solution of the instance $A$. A key element in the polynomial time Turing reduction, aside from the oracle, is the reformulation $B(\cdot)$, which allows one to transfer in polynomial time an optimal solution of $B(A)$ to an optimal solution of $A$. The following definition emphasizes this third phase.

**Definition 3.1.** Let $P_A$ and $P_B$ be two optimization problems. A reformulation $B(\cdot)$ of $P_A$ as $P_B$ is a mapping from $P_A$ to $P_B$ such that, given any instance $A$ of $P_A$ and an optimal solution of $B(A)$, an optimal solution of $A$ can be obtained within a polynomial amount of time.

The reformulation involved in a polynomial time Turing reduction is a polynomial time mapping. It may occur that, even though the optimal solution of $A$ can be obtained within polynomial time from that of $B(A)$, the reformulation is not a polynomial mapping. For example, Balas et al. (Ref. 22), Lovász and Schrijver (Ref. 23), and Sherali and Adams (Ref. 24) present methods to reformulate integer problems into equivalent linear problems by generating all the facets.

A reformulation is completely characterized by $P_A$ and $B(P_A)$. If no large finite constant appear in the instance $B(P_A)$, we denote it as reformulation $P_A \rightarrow P_B$, and otherwise as reformulation $P_A \Rightarrow P_B$.

Links between the bilevel and mixed integer programming problem are studied in this section. Reformulations from one problem to another are not always straightforward i.e., they may require the introduction of intermediary problems. All reformulations considered are schematized in Fig. 1.

**3.1. From MIP Directly to BLP.** We present two reformulations of problems MIBLP and MIP as bilevel problems. The first pair involves a large

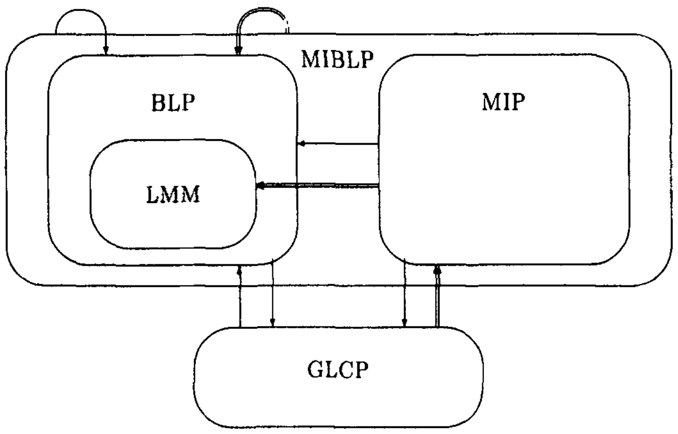

Fig. 1. Reformulations of the problems.

finite constant $L$ which depends on the parameters defining the instance. The second pair does not involve any such constant.

The following proposition slightly generalizes a result of Vicente et al. (Ref. 1) for the case where first-level variables of MIBLP are all constrained to be 0–1. The proof, being similar, is omitted here.

**Proposition 3.1.** Reformulation MIBLP $\Rightarrow$ BLP. There exists a large finite constant $L>0$ such that, if $(x^*, y^*, u^*)$ is an optimal solution of MIBLP, then $(x^*, y^*, u^*, 0)$ is an optimal solution of

$$
\begin{aligned}
\text{(BLP)}\quad \max_{x,y,u,v}\quad & c^{1\prime}x + d^{1\prime}y + e^{1\prime}u - L\mathbf{1}'v,\\
\text{s.t.}\quad & A^1x + B^1y + E^1u \leq b^1,\\
& 0 \leq u \leq 1,\\
& x \geq 0,\\
& (y, v) \in \arg\max_{y,v}\quad c^{2\prime}x + d^{2\prime}y + e^{2\prime}u + \mathbf{1}'v,\\
& \qquad\qquad \text{s.t.}\quad A^2x + B^2y + E^2u \leq b^2,\\
& \qquad\qquad\qquad y \geq 0,\\
& \qquad\qquad\qquad v \leq u,\\
& \qquad\qquad\qquad v \leq 1 - u,
\end{aligned}
$$

where $\mathbf{1}' = (1, 1, \ldots, 1)$.

Moreover, for such an $L$, if $(x^*, y^*, u^*, v^*)$ is an optimal solution of this last problem, then $(x^*, y^*, u^*)$ is an optimal solution of MIBLP, and $v^* = 0$.

The constant $L$ must be large enough in order that the objective function values, evaluated at extreme points of the relaxed feasible region, where $v \neq 0$, are smaller than those where $v = 0$. The former values depend on $L$ as opposed to the latter. Therefore, finding $L$ may in principle be done by an extreme point enumeration technique. In practice, increasing values of $L$ are tried until a rational solution $(\hat{x}, \hat{y}, \hat{u}, \hat{v})$ with $\hat{v} = 0$ is obtained. Moreover, since $L$ could be very large, computational difficulties may arise.

Specializing this result from MIBLP to MIP yields the bilevel reformulation MIP $\Rightarrow$ BLP. It can be seen as a LMM, since the second-level variable $v$ does not appear in the first-level constraints (there are no other second-level variables), and the first-level and second-level objective functions are opposite (the second-level objective function is simply $\mathbf{1}'v$).

The second pair of reformulations is as follows.

**Proposition 3.2.** Reformulation MIBLP→BLP. If $(x^*, y^*, u^*)$ is an optimal solution of MIBLP, then $(x^*, y^*, u^*, 0)$ is an optimal solution of

$$\begin{aligned}
\text{(BLP)}\quad \max_{x,y,u,v}\ & c^{1t}x+d^{1t}y+e^{1t}u,\\
\text{s.t.}\ & A^1x+B^1y+E^1u\leq b^1,\\
& 0\leq u\leq 1,\\
& x\geq 0,\\
& v=0,\\
& (y,v)\in\arg\max_{y,v}\ c^{2t}x+d^{2t}y+e^{2t}u+\mathbf{1}^tv,\\
& \qquad\qquad \text{s.t.}\ A^2x+B^2y+E^2u\leq b^2,\\
& \qquad\qquad\qquad y\geq 0,\\
& \qquad\qquad\qquad v\leq u,\\
& \qquad\qquad\qquad v\leq 1-u.
\end{aligned}$$

Moreover, if $(x^*, y^*, u^*, v^*)$ is an optimal solution of this last problem, then $(x^*, y^*, u^*)$ is an optimal solution of MIBLP, and $v^*=0$.

**Proof.** The first-level constraints of this reformulation ensure that any rational solution $(\hat{x}, \hat{y}, \hat{u}, \hat{v})$ is such that $\hat{v}=0$. Moreover, rationality of the solution implies that

$$\hat{v}=\min\{\hat{u}, 1-\hat{u}\},$$

and therefore $\hat{u}\in\{0, 1\}^{n_u}$. It follows that the solution $(\hat{x}, \hat{y}, \hat{u})$ is rational for MIBLP.

Conversely, to any rational solution $(\hat{x}, \hat{y}, \hat{u})$ of MIBLP, there corresponds a rational solution $(\hat{x}, \hat{y}, \hat{u}, \hat{v})$ of reformulation MIBLP→BLP where $\hat{v}=0$. The result follows from the fact that an optimal solution is rational. □

Once again, this reformulation can be specialized to MIP. It is written in full details as it is extensively referred to in Section 4.3.

**Corollary 3.1.** Reformulation MIP $\rightarrow$ BLP. If $(x^*, u^*)$ is an optimal solution of MIP, then $(x^*, u^*, 0)$ is an optimal solution of

$$
\begin{aligned}
\text{(BLP)} \quad \max_{x,u,v} \quad & c'x + e'u, \\
\text{s.t.} \quad & Ax + Eu \leq b, \\
& 0 \leq u \leq 1, \\
& x \geq 0, \\
& v = 0, \\
& v \in \arg\max_{v} \quad \mathbf{1}'\mathrm{v}, \\
& \qquad \text{s.t.} \quad v \leq u, \\
& \qquad\qquad\quad v \leq 1 - u.
\end{aligned}
$$

Moreover, if $(x^*, u^*, v^*)$ is an optimal solution of this last problem, then $(x^*, u^*)$ is an optimal solution of MIP, and $v^* = 0$.

Vicente et al. (Ref. 1) show that the bilevel linear problem with mixed variables at the second level cannot be reformulated as a trilevel problem using a reformulation similar to reformulation MIBLP $\Rightarrow$ BLP, as the existence of a finite constant $L$ cannot be guaranteed anymore.

Reformulation MIBLP $\rightarrow$ BLP does not involve any particular large constant, and so similar arguments show that the mixed 0–1 linear $n$-level programming problem, with mixed 0–1 variables on each level, can be reformulated as a linear $(n+1)$-level programming problem. The integrality of each set of 0–1 variables is assured by imposing via the next level that the minimum of $u$ and $1-u$ is 0. Trilevel linear programming is studied by e.g., Bard (Ref. 25) and Wen and Bialas (Ref. 26); and multilevel linear programming is studied by e.g., Blair (Ref. 27), Bard and Falk (Ref. 28), and Benson (Ref. 17).

**3.2. From BLP to MIP via GLCP.** Fortuny-Amat and McCarl (Ref. 2) propose a two-stage reformulation of the bilevel programming problem with bilinear constraints into a MIP. We present the implicit reformulation of BLP as a MIP. The first stage consists in reformulating a BLP as a generalized linear complementarity problem. This last problem is defined as follows:

$$\text{(GLCP)}\quad \max_{\tilde{x}} \ \tilde{c}'\tilde{x},$$
$$\text{s.t.}\quad \tilde{A}\tilde{x} \le \tilde{b},$$
$$\tilde{x} \ge 0,$$
$$M\tilde{x} + q \ge 0,$$
$$\tilde{x}'(M\tilde{x} + q) = 0,$$

where $\tilde{x}, \tilde{c}, q \in \mathbb{R}^{\tilde{n}}$, $M \in \mathbb{R}^{\tilde{n}\times\tilde{n}}$, $\tilde{A} \in \mathbb{R}^{\tilde{m}\times\tilde{n}}$, $\tilde{b} \in \mathbb{R}^{\tilde{m}}$.

The three constraints

$$\tilde{x} \ge 0, \qquad M\tilde{x} + q \ge 0, \qquad \tilde{x}'(M\tilde{x} + q) = 0$$

define the classical linear complementarity problem; see, e.g., Júdice and Mitra (Ref. 3) for further discussion.

**Proposition 3.3.** Reformulation BLP $\rightarrow$ GLCP. If $(x^*, y^*)$ is an optimal solution of BLP, then there exists $\lambda^* \in \mathbb{R}^{m_2}$ such that $(x^*, y^*, \lambda^*)$ is an optimal solution of

$$\text{(GLCP)}\quad \max_{x,y,\lambda} \ c^{1\prime}x + d^{1\prime}y,$$
$$\begin{aligned} \text{s.t.}\quad & A^1x + B^1y \le b^1, && \lambda' B^2 \ge d^{2\prime},\\ & x \ge 0, && \lambda \ge 0,\\ & A^2x + B^2y \le b^2, && \lambda'(b^2 - A^2x - B^2y) = 0,\\ & y \ge 0, && y'(B^{2\prime}\lambda - d^2) = 0. \end{aligned}$$

Moreover, if $(x^*, y^*, \lambda^*)$ is an optimal solution of this last problem, then $(x^*, y^*)$ is an optimal solution of BLP.

**Proof.** For a given $x$, the second-level problem of BLP can be written in primal and dual form, respectively,

$$\begin{aligned} &\max_y \ d^{2\prime}y, && \min_\lambda \ \lambda'(b^2 - A^2x),\\ &\text{s.t.}\quad B^2y \le b^2 - A^2x, && \text{s.t.}\quad \lambda' B^2 \ge d^{2\prime},\\ &\qquad y \ge 0, && \qquad \lambda \ge 0. \end{aligned}$$

The complementary slackness conditions are

$$\lambda'(b^2 - A^2x - B^2y) = 0 \quad \text{and} \quad y'(B^{2\prime}\lambda - d^2) = 0.$$

The result follows by replacing the second-level problem by these conditions as well as the primal and dual constraints. □

The linear complementarity structure becomes apparent after performing the substitutions

$$\tilde{A}=[A^1, B^1, 0], \qquad \tilde{b}=b^1,$$

and

$$\tilde{x}=\begin{bmatrix} x \\ y \\ \lambda \end{bmatrix}, \qquad \tilde{c}=\begin{bmatrix} c^1 \\ d^1 \\ 0 \end{bmatrix},$$

$$M=\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & B^{2t} \\ -A^2 & -B^2 & 0 \end{bmatrix}, \qquad q=\begin{bmatrix} 0 \\ -d^2 \\ b^2 \end{bmatrix}.$$

The second stage consists in reformulating the GLCP into a MIP.

**Proposition 3.4.** Reformulation GLCP ⇒ MIP. Suppose that the optimal value of GLCP is bounded, and let $\tilde{x}^*$ be a finite optimal solution. There exists a large finite constant $L>0$ and $\tilde{u}^* \in \mathbb{R}^n$ such that $(\tilde{x}^*, \tilde{u}^*)$ is an optimal solution of

$$\begin{aligned}
\text{(MIP)} \quad & \max_{\tilde{x},u} \ \tilde{c}^t\tilde{x}, \\
& \text{s.t.} \quad \tilde{A}\tilde{x}\leq\tilde{b}, \\
& \tilde{x}\geq 0, \qquad M\tilde{x}+q\geq 0, \\
& \tilde{x}\leq Lu, \qquad M\tilde{x}+q\leq L(1-u), \\
& u\in\{0,1\}^n.
\end{aligned}$$

Moreover, for such an $L$, if $(x^*, u^*)$ is an optimal solution of this last problem, then $x^*$ is an optimal solution of GLCP.

**Proof.** Let $\tilde{x}^*$ be a finite optimal solution of GLCP, and let $(x^*, u^*)$ be an optimal solution of the reformulated problem for

$$L=\max\{\|\tilde{x}^*\|_\infty, \|M\tilde{x}^*+q\|_\infty\},$$

where $\|\cdot\|_\infty$ denotes the infinity norm. For $i \in \{1, 2, \dots, n_u\}$, set

$$\tilde{u}_i^* = 1, \qquad \text{if } \tilde{x}_i^* > 0,$$

$$\tilde{u}_i^* = 0, \qquad \text{otherwise.}$$

Therefore,

$$\tilde{x}^* \le L\tilde{u}^*, \qquad M\tilde{x}^* + q \le L(1 - \tilde{u}^*),$$

and so

$$\tilde{c}^t x^* \ge \tilde{c}^t \tilde{x}^*.$$

Conversely, $(x^*, u^*)$ satisfies

$$x^*(Mx^* + q) = 0;$$

therefore,

$$\tilde{c}^t x^* = \tilde{c}^t \tilde{x}^*. \qquad \square$$

Combination of reformulations BLP $\to$ GLCP and GLCP $\Rightarrow$ MIP yields the following corollary.

**Corollary 3.2.** Reformulation BLP $\Rightarrow$ MIP. Suppose that the optimal value of BLP is bounded, and let $(\tilde{x}^*, \tilde{y}^*)$ be a finite optimal solution. There exists a large finite constant $L > 0$ and $\lambda^* \in \mathbb{R}^{m_2}$, $u^* \in \mathbb{R}^{n_u}$, $v^* \in \mathbb{R}^{n_u}$, such that $(\tilde{x}^*, \tilde{y}^*, \lambda^*, u^*, v^*)$ is an optimal solution of

$$\text{(MIP)} \quad \max_{x,y,\lambda,u,v} \; c^{1t}x + d^{1t}y,$$

$$\begin{array}{lll} \text{s.t.} & A^1x + B^1y \le b^1, & \\ & x \ge 0, & \\ & A^2x + B^2y \le b^2, & -B^{2t}\lambda \le -d^2, \\ & y \ge 0, & \lambda \ge 0, \\ & -A^2x - B^2y + Lu \le L\mathbb{1} - b^2, & \lambda - Lu \le 0, \\ & y + Lv \le L\mathbb{1}, & B^{2t}\lambda - Lv \le d^2, \\ & u \in \{0, 1\}^{m_2}, & v \in \{0, 1\}^{n_y}. \end{array}$$

Moreover, for such an $L$, if $(x^*, y^*, \lambda^*, u^*, v^*)$ is an optimal solution of this last problem, then $(x^*, y^*)$ is an optimal solution of BLP.

**Proof.** The condition of bounded optimal solution in reformulation GLCP $\Rightarrow$ MIP is verified, since we assumed that BLP has a bounded optimal solution. □

**3.3. From MIP to BLP via GLCP.** There are reciprocals to these reformulations. Júdice and Mitra (Ref. 3) show that the MIP can be reformulated as a GLCP.

**Proposition 3.5.** Reformulation MIP $\rightarrow$ GLCP. A solution $(x^*, u^*)$ is optimal for MIP if and only if it is an optimal solution of

$$
\begin{aligned}
\text{(GLCP)}\quad \max_{x,u}\quad & c'x + e'u,\\
\text{s.t.}\quad & Ax + Eu \le b,\\
& x \ge 0,\\
& 0 \le u \le 1,\\
& u'(1-u) = 0.
\end{aligned}
$$

**Proof.** The result follows from the fact that $u \in \{0, 1\}^{n_u}$ if and only if

$$0 \le u \le 1 \quad \text{and} \quad u'(1-u) = 0. \qquad \square$$

The GLCP can in turn be reformulated as a BLP.

**Proposition 3.6.** Reformulation GLCP $\rightarrow$ BLP. If $\tilde{x}^*$ is an optimal solution of GLCP, then $(\tilde{x}^*, 0)$ is an optimal solution of

$$
\begin{aligned}
\text{(BLP)}\quad \max_{\tilde{x},v}\quad & \tilde{c}'\tilde{x},\\
\text{s.t.}\quad & \tilde{A}\tilde{x} \le \tilde{b},\\
& \tilde{x} \ge 0,\\
& M\tilde{x} + q \ge 0,\\
& v = 0,\\
& v \in \arg\max_{v}\ \mathbf{1}'v,\\
& \qquad \text{s.t.}\quad v \le \tilde{x},\\
& \qquad\qquad\ \ v \le M\tilde{x} + q.
\end{aligned}
$$

Moreover, if $(\tilde{x}^*, v^*)$ is an optimal solution of this last problem, then $\tilde{x}^*$ is an optimal solution of GLCP, and $v^* = 0$.

**Proof.** The result follows from the fact that any rational solution $(\hat{x}, \hat{v})$ must satisfy

$$\hat{v} = \min\{\hat{x}, M\hat{x} + q\} = 0. \qquad \square$$

Note that, if reformulations MIP $\rightarrow$ GLCP and GLCP $\rightarrow$ BLP are successively applied, they yield the same bilevel problem as reformulation MIP $\rightarrow$ BLP.

A reformulation similar to reformulation MIBLP $\Rightarrow$ BLP (in which the first-level constraint $v = 0$ is penalized in the objective function) is not valid for GLCP. The following counterexample is such that the relaxed feasible region of the suggested bilevel reformulation is unbounded:

$$\begin{aligned} \max_{\tilde{x} \in \mathbb{R}} \quad & \tilde{x}, \\ \text{s.t.} \quad & \tilde{x} \geq 0, \\ & 0\tilde{x} + 1 \geq 0, \\ & \tilde{x}'(0\tilde{x} + 1) = 0. \end{aligned}$$

One can see that $\tilde{x}^* = 0$ is the only optimal solution of this instance of GLCP.

$$\begin{aligned} \max_{\tilde{x}, v \in \mathbb{R}} \quad & \tilde{x} - Lv, \\ \text{s.t.} \quad & \tilde{x} \geq 0, \\ & 0\tilde{x} + 1 \geq 0, \\ & v \in \arg\max_{v \in \mathbb{R}} \quad Lv, \\ & \qquad \text{s.t.} \quad v \leq \tilde{x}, \\ & \qquad \qquad v \leq 0\tilde{x} + 1. \end{aligned}$$

The objective value of this reformulation is unbounded at $(\tilde{x}, v) = (\infty, 1)$. There is no finite constant $L$ for which this reformulation is valid.

## 4. Embedded Algorithms

For a given optimization problem $P_A$, the notation AL$(P_A)$ refers to an algorithm which can be applied to any instance $A_0$ of $P_A$. Such an algorithm applied to $A_0$ is seen as a method which generates and processes the sequence of instances $\{A_l\}_{l \geq 0}$ until an optimal solution of $A_0$ is found, or until it is shown to be infeasible [if AL$(P_A)$ is a branch-and-bound scheme, the $A_l$ usually refer to subproblems; this concept of algorithm applies to cutting-planes methods, column generation, etc.]. The exact nature of the algorithm is defined by the way it generates and processes the instances. In order to compare alogrithms which process instances of different but related problems (related via a mapping), we introduce the concept of embedded algorithm, which compares the two generated sequences.

**Definition 4.1.** Let $P_A$ and $P_B$ be two optimization problems. Algorithm AL($P_A$) is embedded in AL($P_B$) through the mapping $B(\cdot)$ from $P_A$ to $P_B$, if for any instance $A_0$ of $P_A$, the sequences $\{A_l\}_{l\geq 0}$ generated by AL($P_A$) applied to $A_0$, and $\{B_l\}_{l\geq 0}$ generated by AL($P_B$) applied to $B(A_0)$ are such that, for any $l\geq 0$, $B_l = B(A_l)$.

This definition is intended to be as general as possible. The mapping $B(\cdot)$ is not required to be a reformulation, as the link between both algorithms is characterized by the way the sequences of instances are generated. This allows comparison of algorithms that process problems which are not related by a polynomial time Turing reduction. Moreover, it may happen that different kinds of algorithms can be compared. For example, some cutting-planes algorithms can be viewed, through a transformation to the dual, as column generation methods.

In this section, we present and compare algorithms which solve BLP and MIP. As a close relationship between these algorithms will be exhibited, it is necessary to state them in full detail.

**4.1. Bilevel Programming Problem.** We first describe the branch-and-bound algorithm proposed by Hansen et al. (Ref. 4) to solve BLP, denoted AL(BLP). It uses two relaxations of the current subproblem: the leader relaxation $\mathrm{LR}_{\mathrm{BLP}}$, obtained by omitting the second-level objective function, and the follower relaxation $\mathrm{FR}_{\mathrm{BLP}}(\hat{x})$, which consists of the second-level problem for $x$ fixed to $\hat{x}$. Algorithm AL(BLP) also considers the follower subproblem $\mathrm{FS}_{\mathrm{BLP}}(\hat{x})$, which consists of the second-level problem (of the initial problem) for $x$ fixed to $\hat{x}$.

We associate to the $i$th second-level constraint (including the nonnegativity ones), the Boolean variable $\alpha_i$, equal to 1 if the constraint is tight and equal to 0 otherwise, Using these variables, we derive the following monotonicity relations, i.e., necessary conditions for rationality. In any rational solution to BLP, the tightness of the constraints in the follower subproblem is such that

$$\sum_{i \mid B^2_{ij} > 0} \alpha_i \geq 1, \qquad \text{if } d^2_j > 0,$$

$$\sum_{i \mid B^2_{ij} < 0} \alpha_i + \alpha_{m_2+j} \geq 1, \qquad \text{if } d^2_j < 0,$$

for $j=1,2,\dots,n_y$. The set of monotonicity relations is denoted by $R$. A monotonicity relation is denoted by $r_k$, and is of the form

$$\sum_{i\in I_k} \alpha_i \geq 1,$$

where $I_k$ is a set of indices.

The effect of fixing a variable $\alpha_i$ at 1, i.e., satisfying a constraint as an equality, can be anticipated to some extent by computing a penalty. Consider the equations corresponding to an optimal tableau of $\mathrm{LR}_{\mathrm{BLP}}$ for the current subproblem,

$$z = z_L^* - \sum_{j\in N} c_j^* s_j,$$

$$s_i = b_i^* - \sum_{j\in N} A_{ij}^* s_j, \qquad i\in B,$$

where $s_i$ denotes either a variable $x_i$, $y_i$ or a slack variable, $B$ denotes the index set of basic variables, and $N$ the index set of nonbasic variables. Values with an asterisk correspond to coefficients of the optimal tableau. Then, if $s_i$ is the slack variable of the $i$th constraint, the down penalty for setting $s_i$ at 0 is

$$p_i = b_i^* \min\{c_j^*/A_{ij}^* : j\in N \mid A_{ij}^* > 0\}.$$

It corresponds to the decrease in value of $z_L$ during the first dual-simplex iteration after adding the constraint $s_i \leq 0$.

**Algorithm AL(BLP).**

Step a. Initialization. Initialize the incumbent solution $(x_{\mathrm{opt}}, y_{\mathrm{opt}})$ to an arbitrary value and the incumbent value $z_{\mathrm{opt}}$ to $-\infty$. Consider all variables $\alpha_i$, $i=1,2,\dots,m_2+n_y$, as free. Set $R=\emptyset$.

Step b. First Direct Optimality Test. Solve $\mathrm{LR}_{\mathrm{BLP}}$; let $(x_L^*, y_L^*)$ denote an optimal solution, and let $z_L^*$ be the optimal value. If $z_L^* \leq z_{\mathrm{opt}}$, go to Step m (backtracking).

Step c. First Direct Feasibility Test. Solve the dual of $\mathrm{FR}_{\mathrm{BLP}}(x_L^*)$. If it has no feasible solution, go to Step m.

Step d. Direct Resolution Test, First Part. Consider again the optimal solution $(x_L^*, y_L^*)$ of $\mathrm{LR}_{\mathrm{BLP}}$. Check if $(x_L^*, y_L^*)$ is rational for the current subproblem: solve $\mathrm{FR}_{\mathrm{BLP}}(x_L^*)$, and let $y_F^*$ be an optimal solution. If $d^{2t}y_F^*$, then $(x_L^*, y_L^*)$ is reational; otherwise, go to Step f.

Step e. Direct Resolution Test, Second Part. Consider again the optimal solution $(x_L^*, y_L^*)$ of $\mathrm{LR}_{\mathrm{BLP}}$. Check if $(x_L^*, y_L^*)$ is rational for the initial problem: solve $FS_{BLP}(x_L^*)$, and let $y_{FS}^*$ be an optimal solution. If $d^{2'}y_L^* = d^{2'}y_{FS}^*$, then $(x_L^*, y_L^*)$ is rational: update $z_{opt}$ and $(x_{opt}, y_{opt})$, then go to Step m. Otherwise, go to Step f.

Step f. Second Direct Optimality Test. Compute all penalties $p_i$ associated with strictly positive slack variables in the optimal tableau of $LR_{BLP}$. Set the other $p_i$ equal to 0. Then, for all $k$ such that $r_k \in R$, compute $\pi_k = \min\{p_i: i \in I_k\}$, and set $\Pi = \max\{\pi_k : r_k \in R\}$. If $z_{opt} \geq z_L^* - \Pi$, go to Step m.

Step h. First Conditional Optimality Test. Consider again $LR_{BLP}$ with the penalties $p_i$. For all $i$ such that $z_{opt} \geq z_L^* - p_i$, fix $\alpha_i$ at 0 and update the set of monotonicity relations $R$.

Step i. Third Direct Optimality Test. If $R$ contains a relation $r_k$ such that $\alpha_i = 0$ for all $i \in I_k$, i.e., the monotonicity relation $r_k$ cannot be satisfied, go to Step m.

Step j. Relational Optimality Test. For all remaining $y_j$ appearing in $d^{2'}y$, add to $R$ the logical relations on the $\alpha_i$, if they are nonredundant. Eliminate from $R$ those relations which have become redundant.

Step k. Second Conditional Optimality Test. If $R$ contains a relation $r_k$ such that $\alpha_j = 0$ for all $j \in I_k$, except for one index $i$, set the corresponding $\alpha_i$ to 1. Eliminate from the subproblem a variable $y_j$ remaining in the $i$th constraint, and return to Step b.

Step 1. Branching. Apply the selected branching rule (for the purpose of this paper, we select the Boolean variable $\alpha_i$ associated with the largest penalty $p_i$, but alternative selections are discussed in Section 4.5) to choose either a free variable $\alpha_i$ or a relation $r_k \in R$ for which all variables $\alpha_i$ with $i \in I_k$ are free. In the former case, branch by fixing $\alpha_i$ at 1. In the latter case, branch by fixing the first variable $\alpha_i$ in $r_k$ equal to 1. Eliminate a variable $y_j$ remaining in the $i$th constraint. Return to Step b.

Step m. Backtracking. If branching took place on a variable, find the last $\alpha_i$ branched upon and equal to 1, set this $\alpha_i = 0$, and free the $\alpha_j$ fixed at 0 after $\alpha_i$ was fixed at 1. Otherwise, consider the last logical relation $r_k$ for which less than $|I_k|$ branches have been explored; consider the next branch. If there is no such variable or relation, stop. Otherwise, update the current subproblem and return to Step b.

The original version of AL(BLP) contains a Step g called Second Direct Feasibility Test, which states: If $LR_{BLP}$ is infeasible, go to Step m. If one considers an infeasible solution to have value $-\infty$, then Step b covers this case.

**4.2. Mixed 0-1 Programming Problem.** Algorithm AL(MIP) uses the continuous relaxation $CR_{MIP}$ of MIP, which consists in replacing the integrality constraints $u \in \{0, 1\}^{n_u}$ by $u \in [0, 1]^{n_u}$.

The effect of fixing a variable $u_i$ at 0 or 1 can be anticipated to some extent by computing penalties. As before, let $p_i$ be the down penalty for setting $s_i$ at 0, where $s_i$ denotes either a variable $x_i$, $u_i$ or a slack variable. Algorithm AL(MIP) considers the penalties $p_i^0$ for fixing $u_i$ at 0, and $p_i^1$ for fixing the slack variable associated to the constraint $u_i \leq 1$ at 0 (i.e., for fixing $u_i$ at 1).

The Beale and Small (Ref. 7) branch-and-bound algorithm with penalties for MIP can be stated as follows [letters referring to the identification of the steps are not consecutive to facilitate comparison with algorithm AL(BLP)].

**Algorithm AL(MIP).**

Step a. Initialization. Initialize $(x_{opt}, u_{opt})$ to an arbitrary value and set $z_{opt} = -\infty$.

Step b. First Direct Optimality Test. Solve $CR_{MIP}$; let $(x_C^*, u_C^*)$ denote an optimal solution, and let $z_C^*$ be the optimal value. If $z_C^* \leq z_{opt}$, go to Step m (backtracking).

Step d. Direct Resolution Test. Consider again the optimal solution $(x_C^*, u_C^*)$ of $CR_{MIP}$. If the solution is feasible (i.e., if $u_C^* \in \{0, 1\}^{n_u}$), then update $z_{opt}$ and $(x_{opt}, u_{opt})$, and then go to Step m. Otherwise, go to Step f.

Step f. Second Direct Optimality Test. Compute the penalties $p_i^0$ associated with strictly positive variables $u_i^*$ and $p_i^1$ associated with strictly positive slack variables corresponding to the constraints $u_i \leq 1$, in the optimal tableau of $CR_{MIP}$. Set the other $p_i^h$ equal to 0, $h = 0, 1$. Then for all $i \in \{1, 2, \ldots, n_u\}$, compute $\pi_i = \min\{p_i^0, p_i^1\}$, and set $\Pi = \max\{\pi_i : i = 1, 2, \ldots, n_u\}$. If $z_{opt} \geq z_C^* - \Pi$, go to Step m.

Step k. Conditional Optimality Test. If there are indices $i$ and $h$ such that $z_C^* - p_i^h \leq z_{opt}$, then fix $u_i = 1 - h$ and go to Step b.

Step l. Branching. Let $h$ and $i$ be indices such that $p_i^h$ is maximum. Branch by fixing $u_i = \mathrm{h}$. Return to Step b.

Step m. Backtracking. Find the last variable $u_i$ branched upon and equal to $h$, set this $u_i = 1 - \mathrm{h}$, free the variables fixed after $u_i$ was fixed at $h$, and return to Step b. If there is no such variable, stop.

**4.3. Embedding of Algorithm AL(MIP) in AL(BLP).** In this section, we prove our main result, i.e., that AL(MIP) is embedded in AL(BLP) through reformulation MIP $\rightarrow$ BLP. Throughout the section, $A$ refers to an instance of MIP and $B(A)$ is the bilevel reformulation of $A$ defined by reformulation MIP $\rightarrow$ BLP.

**Lemma 4.1.** $\mathrm{CR}_A$ and $\mathrm{LR}_{B(A)}$ are identical.

**Proof.** The result follows by discarding the variable $v$ from $\mathrm{LR}_{B(A)}$, since $v=0$). □

We now derive properties regarding the algorithms themselves. The nature of the reformulation $B(A)$ allows simplification of the monotonicity relations used by AL(BLP). For $k\in\{1, 2, \ldots, n_u\}$, the monotonicity relation $r_k$ is $\alpha_i^0+\alpha_i^1\geq 1$, where $i=k$ and the binary variables are associated to the second-level constraints as follows:

$$(\alpha_i^0) \quad v_i\leq u_i, \qquad (\alpha_i^1) \quad v_i\leq 1-u_i, \qquad (\alpha_i) \quad v_i\geq 0.$$

These relations can be stated without referring to the set $I_k$, and by using the index $i$ instead of $k$ [this will facilitate comparison with AL(MIP)].

**Proposition 4.1.** Let $A'$ be obtained from $A$ by fixing $u_i$ at $1-h$, for $h\in\{0, 1\}$ and $i\in\{1, 2, \ldots, n_u\}$, and let $B'$ be obtained from $B(A)$ by fixing $\alpha_i^h$ at 0 and $\alpha_i^{1-h}$ at 1. Then, $B'=B(A')$.

**Proof.** If $h=0$, then $B'$ will be such that $\alpha_i^0=0$ and $\alpha_i^1=1$; therefore, $v_i=1-u_i$, and the first-level constraint $v_i=0$ ensures that $u_i=1$. Otherwise, if $h=1$, then $B'$ will be such that $\alpha_i^1=0$ and $\alpha_i^0=1$; therefore, $v_i=u_i$, and the first level constraint $v_i=0$ ensures that $u_i=0$. In both cases, $B'$ is precisely $B(A')$. □

**Lemma 4.2.** For any $\hat{u}\in[0, 1]^{n_u}$, the dual of $\mathrm{FS}_{B(A)}(\cdot, \hat{u})$ is always feasible and bounded.

**Proof.** The optimal solution of $\mathrm{FS}_{B(A)}(\cdot, \hat{u})$ is $\hat{v}=\min\{\hat{u}, 1-\hat{u}\}$, which is clearly feasible and bounded. The result follows from duality theory. □

Both algorithms AL(MIP) and AL(BLP) compute penalties. The following lemma shows the equivalence between them. Penalties are often useless for a degenerate problem. We suppose here that $\mathrm{LR}_{B(A)}$ is stripped of the obvious redundant constraints, i.e., all those which involve the variable

$v$. Thus, when referring to the second-level slack variables, we actually refer to the variable $u$ or the slack variable associated to $u \leq 1$,

**Proposition 4.2.** The penalties used by AL(MIP) applied to $A$ are the same as those used by AL(BIP) applied to $B(A)$.

**Proof.** The penalties $p_i^0$ and $p_i^1$ used by AL(MIP) respectively measure the change in objective function value in the first dual-simplex iteration of $\mathrm{CR}_A$ when $u_i$ is fixed at 0 and 1, for $i \in \{1, 2, \dots, n_u\}$.

The penalties used by AL(BIP) are evaluated for strictly positive second-level slack variables. Recall that the optimal solution of $\mathrm{LR}_{B(A)}$ is such that $v_L^* = 0$; therefore, the penalties are evaluated for the slack variables of the constraints

$$\begin{aligned} &(p_i^0) \quad v_i \leq u_i, &&\text{if } u_i > 0,\\ &(p_i^1) \quad v_i \leq 1 - u_i, &&\text{if } u_i < 1. \end{aligned}$$

The first-level constraint $v = 0$ implies that both penalties are respectively $p_i^0$, the change in the objective function value in the first dual-simplex iteration when the constraint $u_i = 0$ is added, and $p_i^1$, the change in the objective function value in the first dual-simplex iteration when the constraint $1 - u_i = 0$ is added. Lemma 4.1 implies that these are exactly the same penalties as in AL(MIP). □

These preliminaries lead to our main result.

**Theorem 4.1.** Algorithm AL(MIP) is embedded in AL(BLP) through reformulation MIP → BLP.

**Proof.** Let $A_0 = \text{MIP}$ and $B_0 = B(A_0)$, which is in fact reformulation MIP → BLP. Let $\{A_l\}_{l \geq 0}$ and $\{B_l\}_{l \geq 0}$ be the sequences generated by the algorithms applied to $A_0$ and $B_0$.

The proof is done by induction on $l$. For $l = 0$, $B_l = B(A_l)$ by definition.

Suppose that, for a given $l \geq 0$, $B_l = B(A_l)$ and that both incumbent values $z_{\text{opt}}$ coincide. We show that $B_{l+1} = B(A_{l+1})$ and that both updated incumbent values coincide by comparing the algorithms step by step.

Steps a (which occurs only when $l = 0$) of both algorithms are the same; the incumbent value is set at $-\infty$.

Lemma 4.1 ensures that Steps b are the same in both algorithms. Lemma 4.2 states that Step c of AL(BLP) is useless. Moreover, Steps d and e of

AL(BLP) can be combined into checking if $0=\min\{u_L^*, 1-u_L^*\}$. Therefore, these two steps are the same as Step d of AL(MIP); hence, the incumbent $z_{\text{opt}}$ is updated in the same way.

The monotonicity relation for the bilevel reformulation $r_i$, for $i\in\{1, 2, \ldots, n_u\}$, is $\alpha_i^0+\alpha_i^1\geq 1$. Therefore, in AL(BLP), $\pi_i=\min\{p_i^0, p_i^1\}$, as in AL(MIP). Proposition 4.2 implies that Steps f are the same in both algorithms. Step i of AL(BLP) is redundant: if it is reached, then $z_{\text{opt}}<z_L^*-\pi_i$ for each index $i$; therefore, step h will never set both $\alpha_i^0$ and $\alpha_i^1$ at 0.

Steps h, j, k of algorithm AL(BLP) are the same as step k of AL(MIP). After processing Steps h, j, k, algorithm AL(BLP) will go to Step b if and only if there is an index $i$ such that $\alpha_i^h=0$. Consider the case where such an index $i$ exists. The corresponding variable $\alpha_i^{1-h}$ is fixed at 1. Proposition 4.1 implies that these three steps are the same as Step k of AL(MIP), which sets $u_i$ at $1-h$. Therefore, $B_{l+1}=B(A_{l+1})$. Otherwise, if no such index $i$ exists, both algorithms go to Step l.

Step l of AL(BLP) creates two subproblems, one with $\alpha_i^h=0$ $(B')$ and the other with $\alpha_i^h=1$ $(B'')$. Step l of AL(MIP) also creates two subproblems, one $(A')$ with $u_j=0$ and the other $(A'')$ with $u_j=1$. Proposition 4.2 implies that the branching rules are the same in both algorithms, i.e., $i=j$. Proposition 4.1 ensures that $B'=B(A')$ and $B''=B(A'')$.

In Steps m of both algorithms, the current node in the enumeration tree is deleted until an unexplored branch is reached (if one remains), and the induction hypothesis ensures that they go back to Step b with $B_{l+1}=B(A_{l+1})$. □

Algorithm AL(BLP) applied to reformulation MIP $\Rightarrow$ BLP will not yield exactly the same steps. It may develop branches in which a rational solution does not satisfy $\hat{u}\in\{0, 1\}^{n_u}$, since there might be rational solutions for which $\hat{v}\neq 0$. Instead of eliminating rapidly a branch by branching upon $u_i=0$ or $u_i=1$ as with reformulation MIP $\rightarrow$ BLP, it will develop the tree further down by branching upon $v_i=u_i$ or $v_i=1-u_i$.

In order to show the converse of Theorem 4.1, i.e., that AL(BLP) is embedded in AL(MIP), we would have to find a mapping from BLP to MIP and compare the generated sequences of instances. Audet et al. (Ref. 20) present a counterexample which shows that reformulation BLP $\Rightarrow$ MIP does not allow us to reach such a conclusion. This suggests but is not sufficient to conclude that AL(BLP) is not embedded in AL(MIP), since there might exist another mapping from BLP to MIP that would satisfy the condition of Definition 4.1.

**4.4. Example.** In this section, we first solve the following MIP instance:

$$\begin{aligned}\max_{x,u}\quad & 12x+8u_1+3u_2,\\ \text{s.t.}\quad & 4x-u_1-4u_2\le 1,\\ & 5x+4u_1+2u_2\le 3,\\ & x\ge 0,\\ & u\in\{0,1\}^2,\end{aligned}$$

using AL(MIP) and then its bilevel reformulation using AL(BLP).

Solving $\mathrm{CR}_{\mathrm{MIP}}$ leads to $(x_C^*, u_C^*)=(1/2, 0, 1/4)$ and $z_C^*=27/4>z_{\mathrm{opt}}=-\infty$ (Step b). This solution is not feasible since $u_{C2}^*\notin\{0,1\}$ (Step d). The following penalties are found (Step f):

$$p_1^0=0,\quad p_1^1=1/4,\quad \pi_1=0,$$
$$p_2^0=1/12,\quad p_2^1=27/20,\quad \pi_1=1/12,\qquad \Pi=1/12.$$

This branch is not fathomed, since $27/4-\Pi>z_{\mathrm{opt}}=-\infty$. The largest penalty is $p_2^1$, and so the branch $u_2=1$ is explored (Step l). Solving $\mathrm{CR}_{\mathrm{MIP}}$ leads to $(x_C^*, u_C^*)=(1/5, 0, 1)$ and $z_C^*=27/5>z_{\mathrm{opt}}=-\infty$ (Step b). This solution is feasible, and so the incumbent is set to $(x_{\mathrm{opt}}, u_{\mathrm{opt}})=(1/5, 0, 1)$ and $z_{\mathrm{opt}}=27/5$ (Step d). Backtracking (Step m) brings us to explore the branch in which $u_2=0$.

Solving $\mathrm{CR}_{\mathrm{MIP}}$ leads to $(x_C^*, u_C^*)=(1/3, 1/3, 0)$ and $z_C^*=20/3>z_{\mathrm{opt}}=27/5$ (Step b). This solution is not feasible since $u_{C1}^*\notin\{0,1\}$ (Step d). The following penalties are found (Step f):

$$p_1^0=11/3,\quad p_1^1=16/15,\quad \pi_1=16/15,\quad \Pi=16/15.$$

This branch is not fathomed since $20/3-\Pi=28/5>z_{\mathrm{opt}}=27/5$. The variable $u_1$ is fixed at 1 since $z_C^*-p_1^0=10/3\le z_{\mathrm{opt}}=27/5$ (Step k). $\mathrm{CR}_{\mathrm{MIP}}$ is unfeasible (Step b). Backtracking brings the algorithm to stop with optimal solution $(x_{\mathrm{opt}}, u_{\mathrm{opt}})=(1/5, 0, 1)$.

We now use AL(BLP) to solve the bilevel reformulation:

$$\begin{aligned}\max_{x,u,v}\quad & 12x+8u_1+3u_2,\\ \text{s.t.}\quad & 4x-u_1-4u_2\le 1,\\ & 5x+4u_1+2u_2\le 3,\\ & x\ge 0,\\ & 0\le u\le 1,\end{aligned}$$

$$\begin{aligned}
&v=0,\\
&v\in\arg\max_v \quad v_1+v_2,\\
&\text{s.t.}\quad (\alpha_1^0)\quad -u_1+v_1\leq 0,\\
&\qquad\ \ (\alpha_1^1)\quad u_1+v_1\leq 1,\\
&\qquad\ \ (\alpha_2^0)\quad -u_2+v_2\leq 0,\\
&\qquad\ \ (\alpha_2^1)\quad u_2+v_2\leq 1.
\end{aligned}$$

The monotonicity relations are $\alpha_1^0+\alpha_1^1\geq 1$ and $\alpha_2^0+\alpha_2^1\geq 1$. Solving $LR_{BLP}$ leads to $(x_L^*, u_L^*, v_L^*)=(1/2, 0, 1/4, 0, 0)$ and $z_L^*=27/4>z_{opt}=-\infty$ (Step b). This solution is not rational, since the optimal value of $FR_{BLP}$ $(x_L^*, u_L^*)$ is $1/4\neq 0$ (Steps d and e). The following penalties are found (Step f):

$$\begin{array}{llll}
p_1^0=0, & p_1^1=1/4, & \pi_1=0, & \\
p_2^0=1/12, & p_2^1=27/20, & \pi_1=1/12, & \Pi=1/12.
\end{array}$$

This branch is not fathomed, since $27/4-\Pi>z_{opt}=-\infty$. The largest penalty is $p_2^1$, and so the branch $\alpha_2^1=1$, hence $u_2=1-v_2$, is explored (Step l). Solving $LR_{BLP}$ leads to $(x_L^*, u_L^*, v_L^*)=(1/5, 0, 1, 0, 0)$ and $z_L^*=27/5>z_{opt}=-\infty$ (Step b). This solution is rational, and so the incumbent is set to $(x_{opt}, u_{opt}, v_{opt})=(1/5, 0, 1, 0, 0)$ and $z_{opt}=27/5$ (Step d). Backtracking (Step m) brings us to explore the branch in which $\alpha_2^1=0$.

The monotonicity relations imply that $\alpha_2^0=1$, hence $u_2=v_2$. Solving $LR_{BLP}$ leads to $(x_L^*, u_L^*, v_L^*)=(1/3, 1/3, 0, 0, 0)$ and $z_L^*=20/3>z_{opt}=27/5$ (Step b). This solution is not rational, since the optimal value of $FR_{BLP}$ $(x_L^*, u_L^*)$ is $1/3\neq 0$ (Steps d and e). The following penalties are found (Step f):

$$p_1^0=11/3,\qquad p_1^1=16/15,\qquad \pi_1=16/15,\qquad \Pi=16/15.$$

This branch is not fathomed since $20/3-\Pi=28/5>z_{opt}=27/5$. The variable $\alpha_1^0$ is fixed at 0, since $z_L^*-p_1^0=10/3\leq z_{opt}=27/5$ (Step h). The monotonicity relations imply that $\alpha_1^1=1$, hence $u_1=1-v_1$ (Step k). $LR_{BLP}$ is unfeasible (Step b). Backtracking brings the algorithm to stop with optimal solution $(x_{opt}, u_{opt})=(1/5, 0, 1)$.

**4.5. Branching Rules.** The following branching rules are proposed in Ref. 4 for AL(BLP). Denote by $s_i$ the slack variable associated with the second-level constraint $i$, and denote by $\lambda_i$ the corresponding dual variable for $i=1, 2, \ldots, m_2+n_y$.

(BR1) Multiple Branching.

(i) Select the logical relation $r_k \in R$ with smallest cardinality.
(ii) Break ties by choosing a relation with the largest number of slack variables $s_i$ in basis.
(iii) Order the variables $\alpha_i$ in $r_k$ by decreasing values of the penalties $p_i$.

(BR2) Binary Branching. Select the Boolean variable $\alpha_i$ associated with the largest penalty $p_i$.

(BR3) Select the $\alpha_i$ associated with the largest product $s_i \lambda_i$ [same rule as in the algorithm of Bard and Moore (Ref. 6)].

(BR4) Multiple or Binary Branching.

(i) Select the relation $r_k \in R$ with two variables $\alpha_i$ and both the corresponding slack variables $s_i$ in basis, such that $\sum_{i \in I_k} \lambda_i s_i$ is maximum.
(ii) If there is no such relation, use BR3.

These various branching rules have counterparts for AL(MIP).

(BR1) This rule is not very useful since in reformulation MIP $\rightarrow$ BLP, all logical relations have cardinality 2. The logical relations are too simple to base a branching rule upon them.

(BR2) This rule was discussed in Theorem 4.1.

(BR3) The following proposition shows that this branching rule corresponds to a well known rule in mixed 0–1 programming.

**Proposition 4.3.** The branching rule BR3 is equivalent to choosing the variable $u_i$, for $i \in \{1, 2, \ldots, n_u\}$, whose value is closest to 1/2.

**Proof.** Let $s^0$, $s^1$ be the slack variables associated with the constraints $v \leq u$, $v \leq 1 - u$, and let $\lambda^0$, $\lambda^1$ be the dual variables of the problem

$$\min_{\lambda^0, \lambda^1} \quad u'\lambda^0 + (1-u)'\lambda^1,$$

$$\text{s.t.} \quad \lambda^0 + \lambda^1 \geq 1,$$

$$\lambda^0 \geq 0, \qquad \lambda^1 \geq 0.$$

At optimality, they will take the values

$$\lambda_i^0 = 1 - \lambda_i^1 = \begin{cases} 1, & \text{if } u_i \leq 1/2, \\ 0, & \text{if } u_i > 1/2; \end{cases}$$

henceforth, the branching rule BR3 will select the variable associated with the largest product (recall that an optimal solution of $\mathrm{LR}_{\mathrm{BLP}}$ will be such that $v = 0$)

$$s_i^0 \lambda_i^0 = \begin{cases} u_i, & \text{if } u_i \leq 1/2, \\ 0, & \text{if } u_i > 1/2, \end{cases}$$

$$s_i^1 \lambda_i^1 = \begin{cases} 0, & \text{if } u_i \leq 1/2, \\ 1 - u_i, & \text{if } u_i > 1/2. \end{cases}$$

Note that, for a given $i$,

$$\max\{s_i^0 \lambda_i^0, s_i^1 \lambda_i^1\} = \begin{cases} u_i, & \text{if } u_i \leq 1/2, \\ 1 - u_i, & \text{if } u_i > 1/2, \end{cases}$$

which yields the result. □

(BR4) The first step of this rule will be satisfied if there is at least one noninteger variable $u_i$. The sum in question will be $s_i^0 \lambda_i^0 + s_i^1 \lambda_i^1$, which in fact is equal to $\max\{s_i^0 \lambda_i^0, s_i^1 \lambda_i^1\}$; therefore, this rule is identical to BR3.

## 5. Discussion

This paper proposes a way to compare algorithms which process different optimalization problems. Such comparison differs from traditional ones used in complexity theory, in the sense that polynomial reformulation from one problem to another is not required. Hence, it allows one at least in theory to study embedding of algorithms defined for any pair of optimization problems.

These concepts are applied to BLP and MIP. Theorem 4.1 shows that AL(MIP) is embedded in AL(BLP) through reformulation MIP $\rightarrow$ BLP. The mapping from MIP to BLP is simple, i.e., it does not involve any large unknown constant, and moreover it is a reformulation (given an optimal solution of a reformulated instance of MIP, one can obtain an optimal solution of the instance within a polynomial amount of time). This suggests that BLP is at least as difficult as MIP.

At first glance, finding that an algorithm is embedded in another algorithm might appear as a negative result, as the former one would then be redundant. Should this be the case, such simplification and unification might also be viewed as positive. But it will not necessarily be so. Indeed, that $\mathrm{AL}(P_A)$ is embedded in $\mathrm{AL}(P_B)$ means that the latter algorithm, when applied to the mapped instance $B(A)$ of the instance $A$ of $P_A$, generates a sequence of problems equivalent through the same mapping to those generated by $\mathrm{AL}(P_A)$. It does not mean that the time to do so is the same. Even if the mapping augments the size of $P_A$ by only a constant factor, the resulting increase in time may not be negligible. It could be large or very large if the mapping involves a polynomial or nonpolynomial increase in size. Moreover, even when the increase in time is moderate, the former algorithm may be worth keeping, for simplicity or pedagogical reasons (we do not surmise that the classical algorithm for MIP will be replaced by that of Ref. 4 in view of the results of this paper).

The interest of the embedded algorithm concept, and its bearing on algorithm development, becomes apparent in less direct ways. First, from the algorithm development viewpoint, finding that $\mathrm{AL}(P_A)$ is embedded in $\mathrm{AL}(P_B)$ shows that the structures of $P_A$ and $P_B$ have much in common. Therefore, tests of various algorithms for $P_B$ can be specialized for $P_A$. When they do not lead to any new conclusions, this shows that their strength is based on addressing difficulties of $P_B$ not present in $P_A$, thus enhancing the knowledge of the difference between these problem structures. Conversely, tests for $P_A$ can sometimes be generalized to $P_B$. When it is not the case, it means that they exploit some properties of $P_A$ not present in $P_B$, again enhancing the knowledge of the difference between these problem structures. Audet et al. (Ref. 20) generalize the improved penalties developed by Tomlin (Ref. 29) for MIP to BLP.

Second, from the complexity viewpoint, finding that $\mathrm{AL}(P_A)$ is embedded in $\mathrm{AL}(P_B)$, but than the converse does not appear to be true, suggests that $P_B$ is more difficult than $P_A$, even if both $P_A$ and $P_B$ belong to the same class of NP-complete or NP-hard problems. This could be corroborated by the study of other types of algorithms for these problems. In particular, it seems worthwhile to study if cutting-planes algorithms for MIP are embedded in existing or potential cutting-planes algorithms for BLP.

Third, systematic study of embeddings of algorithms for several related problems could reveal a structure describing their relative difficulty. Symmetry between two algorithms, i.e., the case where each is embedded in the other through given mappings, implies that there is a very strong similarity between the problems which they address. Moreover, embedding of algorithms is transitive, and so the induced structure between problems would be a preorder (i.e., transitive, reflexive, b ıt not necessarily symmetric or asymmetric). Using this structure, one might study how tests for algorithms for specific problems can be generalized or specialized. In time, this might lead to many new tests and algorithms, and to a better understanding and organization of them.

## References

1. Vicente, L. N., Savard, G., and Júdice, J. J., *Discrete Linear Bilevel Programming Problem*, Journal of Optimization Theory and Application, Vol. 89, pp. 597–614, 1996.
2. Fortuny-Amat, J., and McCarl, B., *A Representation and Economic Interpretation of a Two-Level Programming Problem*, Journal of the Operational Research Society, Vol. 32, pp. 783-792, 1981.
3. Júdice, J. J., and Mitra, G., *Reformulations of Mathematical Programming Problems as Linear Complementary Problems and Investigation of Their Solution Methods*, Journal of Optimization Theory and Applications, Vol. 57, pp. 123-149, 1988.
4. Hansen, P., Jaumard, B., and Savard, G., *New Branch-and-Bound Rules for Linear Bilevel Programming*, SIAM Journal on Scientific and Statistical Computing, Vol. 13, pp. 1194-1217, 1992.
5. Júdice, J. J., and Faustino, A. M., *A Sequential LCP Method for Bilevel Linear Programming*, Annals of Operations Research, Vol. 34, pp. 89–106, 1992.
6. Bard, J. F., and Moore, J. T., *A Branch-and-Bound Algorithm for the Bilevel Linear Programming Problem*, SIAM Journal on Scientific and Statistical Computing, Vol. 11, pp. 281–292, 1990.
7. Beale, E. M. L., and Small, R. E., *Mixed Integer Programming by a Branch-and-Bound Technique*, Proceedings of the 3rd IFIP Congress 1965, Vol. 2, pp. 450-451, 1965.
8. Ben-Ayed, O., *Bilevel Linear Programming*, Computers and Operations Research, Vol. 20, pp. 485-501, 1993.
9. Vicente, L. N., and Calamai, P. H., *Bilevel and Multilevel Programming: A Bibliography Review*, Journal of Global Optimization, Vol. 5, pp. 291-306, 1994.
10. Falk, J. E., *A Linear Max-Min Problem*, Mathematical Programming, Vol. 5, pp. 169–188, 1973.
11. Audet, C., Jaumard, B., and Savard, G., *Concavity Cuts for the Linear Maxmin Problem*, Report G-94-52, Les Cahiers du GERAD, 1994.
12. Jeroslow, R. G., *The Polynomial Hierarchy and a Simple Model for Competitive Analysis*, Mathematical Programming, Vol. 32, pp. 146-164, 1985.
13. Ben-Ayed, O., and Blair, C. E., *Computational Difficulties of Bilevel Linear Programming*, Operations Research, Vol. 38, pp. 556–559, 1990.
14. Bard, J. F., *Some Properties of the Bilevel Programming Problem*, Journal of Optimization Theory and Applications, Vol. 68, pp. 371-378, 1991.

15. Bialas, W., and Karwan, M., *On Two-Level Optimization*, IEEE Transactions on Automatic Control, Vol. 27, pp. 211–214, 1982.
16. Candler, W., and Townsley, R., *A Linear Two-Level Linear Programming Problem*, Computers and Operations Research, Vol. 9, pp. 59–76, 1982.
17. Benson, H. P., *On the Structure and Properties of a Linear Multilevel Programming Problem*, Journal of Optimization Theory and Applications, Vol. 60, pp. 353–373, 1989.
18. Savard, G., *Contributions à la Programmation Mathématique à Deux Niveaux*, Thèse de Doctorat, École Polytechnique de Montréal, 1989.
19. Wen, U. P., and Yang, Y. H., *Algorithms for Solving the Mixed Integer Two-Level Linear Programming Problem*, Computers and Operations Research, Vol. 17, pp. 133–142, 1990.
20. Audet, C., Hansen, P., Jaumard, B., and Savard, G., *Links between the Linear Bilevel and Mixed 0–1 Programming Problems*, Report G-95-20, Les Cahiers du GERAD, 1995.
21. Garey, M. R., and Johnson, D. S., *Computers and Intractability*, W. H. Freeman and Company, New York, New York, 1979.
22. Balas, E., Ceria, S., and Cornuéjols, G., *A Lift-and-Project Cutting Plane Algorithm for Mixed 0–1* Programs, Mathematical Programming, Vol. 58, pp. 295–324, 1993.
23. Lovász, L., and Schrijver, A., *Cones of Matrices and Set-Functions and 0–1 Optimization*, SIAM Journal on Optimization, Vol. 1, pp. 166–190, 1991.
24. Sherali, H. D., and Adams, W. P., *A Hierarchy of Relations between the Continuous and Convex Hull Representations for Zero-One Programming Problems*, SIAM Journal on Discrete Mathematics, Vol. 3, pp. 411–430, 1990.
25. Bard, J. F., *An Investigation of the Linear Three-Level Programming Problem*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 14, pp. 711–717, 1984.
26. Wen, U., and Bialas, W., *The Hybrid Algorithm for Solving the Three-Level Linear Programming Problem*, Computers and Operations Research, Vol. 13, pp. 367–377, 1986.
27. Blair, C. E., *The Computational Complexity of Multi-Level Linear Programs*, Annals of Operations Research, Vol. 34, pp. 13–19, 1992.
28. Bard, J. F., and Falk, J., *An Explicit Solution to the Multi-Level Programming Problem*, Computers and Operations Research, Vol. 9, pp. 77–100, 1982.
29. Tomlin, J. A., *An Improved Branch-and-Bound Method for Integer Programming*, Operations Research, Vol. 19, pp. 1070–1075, 1971.