Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 7, No. 1, pp. 139–162, January 2011

AN ENUMERATION SEQUENTIAL LINEAR PROGRAMMING ALGORITHM FOR BILEVEL PROGRAMMING WITH LINEAR CONSTRAINTS*

Jean Bosco Etoa Etoa

Abstract: In this paper, we present a new method to solve nonlinear bilevel programming problems with linear constraints. The enumeration sequential linear programming algorithm uses monotonicity properties of the lower level problem within an enumeration framework. At each step, the algorithm attempts to compute an induced descent direction within the tightness of lower level constraints and monotonicity networks. Computational results are reported and compared to those of some previous methods.

Key words: *bilevel programming, enumeration sequential linear programming, induced descent direction, monotonicity network, linear constraints*

Mathematics Subject Classification: *90C27, 90C33, 90C57*

1 Introduction

Multilevel optimization problems are mathematical programs that have a subset of their variables constrained to be an optimal solution of another programs parameterized by their remaining variables. When these parameterized programs are pure mathematical programs, we are dealing with bilevel programming problems (BPP). Early formulations of BPP can be found in [32, 46] while Candler and Norton [14, 15] were the first to use the term 'bilevel' or 'multilevel'. Comprehensive overview of BPP can be found in [18, 20, 49]. Properties as optimality conditions of BPP have been studied in some papers: [2, 6, 9, 12, 13, 16, 19, 44]. Jeroslow [29], Ben-Ayed and Blair [8] proved that the linear BPP is NP-hard while Hansen et al. [27] strengthened this result by showing its strongly NP-hardness. Marcotte and Savard [39] present the relationship between two specific classes of bilevel programs to well-known combinatorial problems. Recently, an alternative definition of linear bilevel programming solution has been proposed in [33, 34, 35] pointing out a deficiency on well known definition of this problem. The proposed new definition is equivalent to moving the upper-level constraints involving the second variable into the lower level, which change the nature of the problem [40]. The number of papers presenting applications is constantly growing. Interesting applications include the investigation of network of oligopolies [1], as well as the determination of optimal prices, as road tolls or prices for electricity [11, 44]. Related is the determination of optimal tax credits for biofuel production [8]. An overview on applications of bilevel programming can be found in [20, 22, 38, 49].

*This research was completed during a PhD thesis research of the author [21].

Algorithms usually used to solve bilevel programming problems can be subdivided into three classes: (i) algorithms solving BPP globally, (ii) methods for computing locally optimal solutions namely stationary points and (iii) heuristics. For linear bilevel BPP, it is well known that if the induced region is nonempty, at least an optimal solution is obtained at an extreme point of the feasible set. This motivated Jdice and Faustino [30] to propose an algorithm for computing an ε-approximative global optimum for a BPP. The other class of methods contains branch-and-bound algorithms which exploit the disjunctive nature of the complementarity constraints obtained from the optimality conditions of the lower level problem (see [7, 21, 27]). Local optimization procedures have also been developed (see [5, 23, 25, 26, 31, 37, 42, 45, 47]).

The *HJS* algorithm (see [27]) exploits necessary conditions for subsets of the lower level problem to contain at least one active constraint. That algorithm also investigates branching rules, based on logical relation expressing the conditions of the tightness of constraints which have been detected. In this paper, we use a similar framework and we introduce the new concept of monotonicity networks. A monotonicity path is used to express the tightness of subsets of the lower level constraints, depending on the current rational solution. We propose an enumeration algorithm to solve a BPP with linear constraints and nonlinear differentiable upper level objective function. The method performs a sequential search of partitions of indexes of lower level variables, exploiting monotonicity properties. For a given partition, a sequential linear approximation algorithm is used to compute an improving rational solution. The algorithm proposed in this paper is a descent method that produces rational iterates converging to a local optimal of the problem solved.

The rest of the paper is divided into three sections. The next section is devoted to the conceptual framework of our algorithm. We introduce sufficient optimality conditions for a bilevel programming problem. Using monotonicity principles within an enumeration framework, we introduce the concept of monotonicity networks to express the tightness of the lower level problem. The algorithm is stated in Section 3 and illustrated by an example in Section 4. Computational experiences are reported in Section 5 on nonlinear and linear problems.

2 Conceptual Framework of Enumeration Sequential Linear Programming Algorithm (ESLP)

This section presents two notions. The first one defines the concept of rational solutions which characterize the feasible set of a bilevel program. The second one presents the concept of monotonicity analysis that allows deducing necessary optimality conditions on the tightness of the constraints. The *ESLP* algorithm presented in Section 3 consists in a sequential algorithm, driven by an efficient walk along these monotonicity conditions that iterates from a rational solution to better one, by solving at any iteration a linear program.

2.1 Theoretical Considerations

The bilevel program considered in this paper is formulated as:

$$BPPL: \quad \begin{array}{l} \min_{x,y} f^1(x,y) \\ s.t. \begin{cases} x \in P, \\ \max_y f_2^T y, \\ s.t. \begin{cases} Ax + By = b, \\ x, y \geq 0, \end{cases} \end{cases} \end{array} \tag{2.1}$$

where $A \in \mathbb{R}^{m \times n_x}$, $b \in \mathbb{R}^m$, $f_2 \in \mathbb{R}^{m+n_y}$, $P = \{x \in \mathbb{R}_+^{n_x} : A_1 x \leq b_1\}$ is a polyhedron in $\mathbb{R}_+^{n_x}$, with $A_1 \in \mathbb{R}^{m_1 * n_x}$, $b \in \mathbb{R}^{m_1}$ x and $y \in \mathbb{R}^{m+n_y}$ represent respectively the upper level and lower level variables, $f^1 : \mathbb{R}^{n_x} \times \mathbb{R}^{m+n_y} \to \mathbb{R}$ is assumed to be continuously differentiable. $x \in P$ represents the upper level constraints, while the constraints defined by $Ax + By = b$ represent the lower level constraints. The matrix B is such that $B = [B^e \ I_m]$ where $B^e \in \mathbb{R}^{m \times n_y}$ and $y = \left(y_1, y_2, ..., y_{n_y}, y_{\bar{1}}, ..., y_{\bar{m}}\right)^T$, where $y_{\bar{1}}, ..., y_{\bar{m}}$ are slack variables and I_m is the identity matrix.

The polyhedron $\Omega = \left\{(x,y) \in P \times \mathbb{R}_+^{n_y+m} : Ax + By = b\right\}$ defined by both upper and lower level is called the feasible region of the problem BPPL. Given a value x of the upper level variable, $M(x)$ is the set of optimal solutions of the lower level problem, while an element of the induced region defined by $IR = \{(x,y) \in \Omega : y \in M(x)\}$ is called a rational solution. The induced region is usually nonconvex and, in the presence of upper level constraints involving lower level variables can be disconnect (see [44]). In the absence of lower level variables in upper level constraints, it has been shown that the induced region is the union of finite connected faces of the feasible region Ω (see [9, 10, 12]).

The lower level variable includes slack variables; we assume that $n_y \geq m$ and the rank of the matrix B^e is equal to m (as usually in linear programming). The following assumption is made in order to assure that problem (2.1) is well posed.

H1. We assume that the polyhedron Ω of the feasible region is nonempty as well as the induced region IR.

From optimality conditions of the lower level linear program, we obtain the equivalent KKT formulation of the problem (2.1):

$$LKS: \quad \begin{array}{l} \min_{x,y,\lambda} f^1(x,y) \\ s.t \begin{cases} Ax + By = b, \\ \lambda^T B \geq f_2, \\ (\lambda^T B - f_2) y = 0, \\ x \in P, y \geq 0, \end{cases} \end{array} \tag{2.2}$$

where $\lambda \in \mathbb{R}^m$ corresponds to the dual variables of the lower level constraints. LKS is a mathematical program with equilibrium constraints. If we assume that the solution of the problem LKS is nondegenerate, the stationarity conditions at a nondegenerate feasible solution of (LKS) can be easily derived from corollary 6.1.3 in [36]. Let $\mathfrak{F}^{LKS}$ be the nonempty set of feasible solutions of the problem (2.2).

We define the **support** of $y \in \mathbb{R}_+^{n_y+m}$ as

$$sp(y) = \{j \in H : y_j = 0\}, \quad \text{where} \quad H = \{1, 2, ..., n_y\} \cup \{\bar{1}, \bar{2}, .., \bar{m}\}$$

By the assumption (H1), the set $\mathfrak{F}^{LKS}$ is nonempty, i.e., there exists at least $(x_e, y_e, \lambda_e) \in \mathfrak{F}^{LKS}$. Each point $(x_e, y_e, \lambda_e) \in \mathfrak{F}^{LKS}$ is such that the complementary relation $(\lambda_e^T B - f_2) y_e = 0$ is satisfied; hence there exists $T \subset H : T \neq \emptyset$ such that $T \subset sp(y_e)$.

The *ESLP* algorithm is closed to an active set procedure (see [24]): it consists in computing a series of points contained in the induced region by means of exploring the support of the lower level variable as well as the tightness of the lower level dual constraint $\lambda^T B \geq f_2$. A rational solution $(\bar{x}, \bar{y})$ computed by the *ESLP* algorithm is defined as an **improving rational solution** if we have $f^1(\bar{x}, \bar{y}) < f^1(x^*, y^*)$. From a rational solution $z^* = (x^*, y^*)$, within an enumeration framework, the main idea of the *ESLP* algorithm is to find a new partition (T, T^C) of indexes of lower level variables with $T \neq \emptyset$ such that $T \subseteq sp(\bar{y})$, $T \cup T^C = H$, $(\bar{x}, \bar{y})$ being an improving rational solution. The optimal solution computed by *ESLP* algorithm is a local optimal solution.

We use a sequential linear programming algorithm similar to the method described in [41, 43] for nonlinear programming to compute an induced descent direction $d = (dx, dy)$ defined as follow:

Definition 2.1. We consider the bilevel programming problem (2.1) and let $z^k = (x^k, y^k) \in \Omega$; $d = (dx, dy) \in \mathbb{R}^{n_x+m+n_y}$ is called an **induced descent direction** computed from z^k if there exists $\sigma_0 > 0$ such that:

$$(x^k + \sigma dx, y^k + \sigma dy) \in IR \quad \text{forall} \quad \sigma \in [0, \sigma_0] \quad \text{and} \nabla_{(x,y)} f^1(x^k, y^k) \begin{pmatrix} dx \\ dy \end{pmatrix} < 0.$$

From a given feasible solution, our algorithm computes some descent direction or some induce descent direction that improves the value of the upper level function. Let $(x^k, y^k) \in \Omega$, and $\lambda^k \in \mathbb{R}^m$ be the corresponding dual variables of the lower level constraints. From a feasible solution $w^k = (x^k, y^k, \lambda^k) \in \mathfrak{F}^{LKS}$, our algorithm attempts to compute a next feasible solution $w^{k+1} = (x^k + \sigma dx, y^k + \sigma dy, \lambda^k + \sigma d\lambda)$ of problem *LKS* with $\sigma > 0$ such that $(f_2^k - \sigma d\lambda^T B)_j (y^k + \sigma dy)_j = 0$, $j \in H$ where $f_2^k = f_2 - (\lambda^k)^T B$. Let (T, T^C) be a partition of the set of indexes Hsuch that:

$$\begin{cases} (f_2^k - \sigma d\lambda^T B)_T = 0 \ \text{ if } \ (y^k + \sigma dy)_T > 0, \\ (f_2^k - \sigma d\lambda^T B)_{T^C} < 0 \ \text{ if } \ (y^k + \sigma dy)_{T^C} = 0. \end{cases} \tag{2.3}$$

To simplify, we assume that $\sigma = 1$. A descent direction $(dx, dy, d\lambda)$ can be computed by solving the linearized program:

$$LLP(w^k, T): \quad \max_{dx,dy,d\lambda,\xi} \xi \quad s.t. \begin{cases} \nabla_{(x,y)} f^1(x^k, y^k) \begin{pmatrix} dx \\ dy \end{pmatrix} + \xi \leq -\varepsilon, \\ Adx + Bdy = 0, \\ (f_2^k - d\lambda^T B)_T \leq 0, \\ (f_2^k - d\lambda^T B)_{T^C} = 0 \\ dy_T = -y_T^k, \ dy_{T^C} \geq -y_{T^C}^k, x^k + dx \in P, \ \xi \geq 0, \end{cases} \tag{2.4}$$

where $\varepsilon > 0$ being very small and V_L represents the components of a vector V with indexes in L. In the formulation (2.4), if the first constraint in (2.4) is removed as well as the variable ξ , and the objective function is replaced by

$$\min_{dx,dy,d\lambda} \nabla_x f^1(x^k, y^k) dx + \nabla_y f^1(x^k, y^k) dy \leq -\varepsilon, \tag{2.5}$$

the result is an equivalent formulation of $LLP(w^k, T)$.

The linearization method above is based on the approach described in [41, 48] to solve a convex nonlinear programming problem. In the following definition, we introduce the concept of optimal solution computed by our algorithm.

Definition 2.2. We consider the bilevel programming problem (2.1) and let $z^k = (x^k, y^k) \in \Omega$ be a rational solution; if for all partitions (T, T^C) in H, no induced descent direction from z^k exists, then z^k is an optimal solution of the problem (2.1). The best rational solution computed by the *ESLP* algorithm is called an ***ESLP* optimal solution**.

Remark 2.3. Assume that the linear program $LLP(w^k, T)$ has an optimal solution $(dw, \xi*) = (dx, dy, d\lambda, \xi*)$. The following situation may occur if one uses the program $LLP(w^k, T)$ to compute a descent direction or an induced descent direction from w^k:

i) If $LLP(w^k, T)$ does not have a solution, another partition (T, T^C) has to be considered.

ii) If $\xi^* > 0$, we have $\nabla_{(x,y)} f^1(x^k, y^k)^T (dx, dy) \leq -\xi < 0$; then (dx, dy) is a descent direction , see [41, Theorem 7]. Let $\sigma > 0$ such that $f^1(x^k + \sigma dx, y^k + \sigma dy) \leq f^1(x^k, y^k)$ and let $w^{k+1} = (x^k + \sigma dx, y^k + \sigma dy, \lambda^k + \sigma d\lambda)$. According to (2.4), we have :

$$\begin{cases} Ax^{k+1} + By^{k+1} = b, \\ y_T^{k+1} = 0 \ \text{ and } \ ((\lambda^{k+1})^T B - f_2^{k+1})_T \geq 0, \\ y_{T^C}^{k+1} \geq 0 \ \text{ and } \ ((\lambda^{k+1})^T B - f_2^{k+1})_{T^C} = 0, \\ x^{k+1} \in P, \end{cases} \tag{2.6}$$

i.e. w^{k+1} is a feasible solution of the problem *LKS* and (dx, dy) is an induced descent direction.

iii) $(\bar{x}, \bar{y}, \bar{\lambda}) = (x^k, y^k, \lambda^k)$ is the best computable solution of the problem *LKS*, i.e.$(\bar{x}, \bar{y})$ is an *ESLP* optimal solution of the BPPL problem (2.1) if for the partition the given (T, T^C), we have:

(a) $\xi^* = 0$ when $\varepsilon \cong 0$; then the current solution $w^k = (x^k, y^k, \lambda^k)$ is a stationary point of *LKS*: the KKT multipliers of the mathematical program with equilibrium constraints defined by (2.2) are derived from the multipliers of the linear programming problem $LLP(w^k, T)$(use [41, Theorem 7] and corollary 6.1.3 in [36] for the proof);

(b) for all partitions $(T_1, T_1^C) \neq (T, T^C)$ in H, the linearized program $LLP(w^k, T_1)$ is such that:

- $LLP(w^k, T_1)$ does not have a solution;
- the solution $(dw, \xi*) = (dx, dy, d\lambda, \xi*)$ of $LLP(w^k, T_1)$ is such that (dx, dy) is not an induced descent direction from z^k.

The rate of convergence of such method is generally linear (see [21, 43]). Instead of solving problem $LLP(w^k, T)$, and following the approach taken by *HJS* [27], we solve two problems. The first problem $LPR(z^k, T)$ defined in the primal space, computes an improving solution that is then checked for rationality by evaluating the lower level problem. We have:

$$LPR(\ z^k, T): \quad \max_{dx,dy,\xi} \xi \quad s.t. \begin{cases} \nabla_z f^1(x^k, y^k) \begin{pmatrix} dx \\ dy \end{pmatrix} + \xi \leq -\varepsilon, \\ Adx + Bdy = 0, \\ y_j^k + dy_j = 0, \ j \in T, \\ y_j^k + dy_j \geq 0, \ j \in T^C, \\ x^k + dx \in P, \xi \geq 0, \end{cases} \tag{2.7}$$

where $z^k = (x^k, y^k) \in \Omega$ and $\varepsilon > 0$ being very small; the presence of $-\varepsilon$ in the first constraint of $LPR(z^k, T)$ is to prevent to have $(\xi*, dx, dy) = (0, 0, 0)$ as optimal solution. An optimal

solution $(\xi*, dx, dy)$ of the problem $LPR(z^k, T)$corresponds to a rational solution if for some $\sigma > 0$, $y^{k+1} = y^k + \sigma dy$ is solution of the lower level linear problem for $x^{k+1} = x^k + \sigma dx$ fixed:

$$LP_2\ (x^k): \quad \begin{matrix} \max_y d_2^t y \\ s.t. \end{matrix} \begin{cases} By = b - A(x^k + \sigma dx), \\ y \geq 0. \end{cases} \tag{2.8}$$

This last linear program insures that the solution is a rational one, that is $(x^{k+1}, y^{k+1}) \in IR$. The dual variables λ associated to the constraints of (2.8) and any solution of $LPR(z^k, T)$ is a solution of $LLP(w^k, T)$. At optimality of LP (2.7), the computed value σ_0 of σ can lead to some different partition (T', T'^C) of H rather than (T, T^C).

In the following theorem, we introduce sufficient optimality conditions for the problem (2.1) to be used in our algorithm. The proof is deduced from Definition 2.2 and Remark 2.3.

Theorem 2.4. *Under assumption (H1), let IR be the induced region of the bilevel program defined by (2.1). For a given integer k, we consider a rational solution* $(x^k, y^k) = (x^*, y^*) \in IR$*, the set of indexes* $H = \{1, 2, ..., n_y\} \cup \{\bar{1}, \bar{2}, ..., \bar{m}\}$ *and the family of LP problems* $LPR(z^{k_0}, T)$*,* $T \subset H$ *where* $k_0 \geq k$ *and* $z^{k_0} \in \Omega$*. For* (x^*, y^*) *to be an optimal solution of the problem (2.1), it is sufficient to have, for all* $T \subset H$*, one of the following situations:*

1. *The program* $LPR(z^{k_0}, T)$ *does not have a solution.*
2. *If* $(\xi*, dx, dy)$ *is an optimal solution of the program* $LPR(z^{k_0}, T)$*, then* (dx, dy) *is not an induced descent direction.*

2.2 Necessary Optimality Conditions: the M-Principles

This subsection presents the monotonicity principles that are used to exploit efficiently the partition of indexes in H at each step of our algorithm, in order to compute an induced descent direction. These principles where first introduced by Wilde [51] and generalized by Hansen et al. [27]. The *HJS* algorithm [27] takes advantage of these principles within a branch and bound framework by exploiting the condition of tightness of the lower level constraints. For the rest of the paper, a monotonicity property will be denoted by an M-principle.

Let $LP_2(\bar{x})$ be the follower parametric LP problem in standard form:

$$LP_2(\bar{x}): \quad \begin{matrix} \max_y \sum_{j=1}^{n_y} f_{2j} y_j \\ s.t. \end{matrix} \begin{cases} \sum_{j=1}^{n_y} B_{ij}^e y_j + y_{\bar{i}} = b_i - \sum_{t=1}^{n_x} A_{it} \bar{x}_t, \\ y_j \geq 0, j = 1, 2, ..., n_y;\ y_{\bar{i}} \geq 0, i = 1, 2, ..., m. \end{cases}$$

For each structural variable y_j, with $j \in \{1, 2, ..., n_y\}$, we define the following set of indexes:

$$I_j^+ = \left\{ i \in \{1, 2, ..., m\} : B_{ij}^e > 0 \right\},$$
$$I_j^- = \left\{ i \in \{1, 2, ..., m\} : B_{ij}^e < 0 \right\}.$$

The following properties summarized the two M-principles, with respect to $PL_2(\bar{x})$.

Property 2.5 (first M-principle)**.** Let (x^*, y^*) be an optimal solution of problem (2.4). Then, for any $j \in \{1, 2, ..., n_y\}$ such that $f_{2j} < 0$, there exists at least one index $i \in I_j^- \cup \{j\}$

such that: a) $\bar{i} \in sp(y^*)$ ($dy_{\bar{i}} = -y^*_{\bar{i}}$) if $i \in I_j^-$, b) $j \in sp(y^*)$ ($dy_j = -y^*_j$) if $i = j$. For any $j \in \{1, 2, ..., n_y\}$ such that $f_{2j} > 0$, there exists at least one index $i \in I_j^+$ such that $\bar{i} \in sp(y^*)$.

The above property is a consequence of corollary metric converter Corollary 4.2 in [27]. Another formulation of theorem metric converter Corollary 3.2 in [44] is contained in the following property, an expression of the second M-principles, applied when the objective function does not depend on a variable. It is only valid when the set $M(x)$ has at most one element.

Property 2.6 (second M-principle)**.** Under the assumption (H1), we assume that the set $M(x)$ has at most one element. Let (x^*, y^*) be an optimal solution of the problem (2.1) according to Definition 2.2. Then, for all $j \in \{1, 2, ..., n_y\}$ such that $f_{2j} = 0$, there exists at least an index $i \in I_j^+ \cup I_j^-$ such that $\bar{i} \in sp(y^*)$.

The following property is another formulation of Property 2.5 and represents necessary optimality conditions, formulated as the tightness of the lower level constraints. Each of such constraint is associated to a boolean variable α_i equal to 1 if the constraint is active ($\bar{i} \in sp(y^*)$), and equal to 0 otherwise.

Property 2.7 (Theorem 4.1 and Corollary 4.2, [27])**.** For any rational solution of problem (2.1), the tightness of the constraints in the lower level problem is such that

$$\sum_{i:B^e_{ij}>0} \alpha_i \geq 1 \ \text{ if } \ f_{2j} > 0, \qquad \text{(m1)}$$

$$\sum_{i:B^e_{ij}<0} \alpha_i + \alpha_{m+j} \geq 1 \ \text{ if } \ f_{2j} < 0, \qquad \text{(m2)}$$

$$\sum_{i:B^e_{ij}\neq 0} \alpha_i \geq 1 \ \text{ if } \ f_{2j} = 0, \qquad \text{(m3)}$$

for $j = 1, ..., n_y$ and $i = 1, ..., m$.

For any rational solution (x^*, y^*), if a boolean variable α_i (respectively α_{m+j}) is equal to 1, necessarily, $\bar{i} \in sp(y^*)$ (respectively $j \in sp(y^*)$). If each element of the set of boolean variables α_i with indexes included in $T \subset H$ is equal to one, then $T \subset sp(y^*)$. From the above logical relations and, for any value f_{2j} of the lower level objective function, we introduce an M-indexes subset defined as follows for $j \in \{1, 2, ..., n_y\}$ and $i \in \{1, 2, ..., m\}$:

$$J_j = \begin{cases} \{\bar{i} : B^e_{ij} > 0\} \ \text{ if } \ f_{2j} > 0, \\ \{\bar{i} : B^e_{ij} < 0\} \cup \{j\} \ \text{ if } \ f_{2j} < 0, \\ \{\bar{i} : B^e_{ij} \neq 0\} \ \text{ if } \ f_{2j} = 0. \end{cases}$$

We mention that, when a subset J_j is such that $|J_j| = 1$, then necessarily at optimality, we have, $t \in sp(y^*)$ for $\{t\} = J_j$.

3 Conceptual Framework of the ESLP Algorithm

We now introduce the notion of M-networks, constructed from the M-indexes subsets for a given problem (2.1). M-networks are based on the support of the lower level variable as well as on some logical relations expressing the tightness of lower level constraints at the optimality; these M-networks are used in our algorithm in a branch and bound framework can compute improving rational solution from a given feasible solution of problem (2.2).

3.1 The M-Networks

The M-principles are used in the *ESLP* algorithm as follows: let J_{j_0} be an M-indexes subset with the smallest cardinality, i.e. $|J_{j_0}| = \min_{1\leq j\leq n_y} |J_j|$. We introduce the **M-network** G_M^0 as follows: consider an initial ordered sequence of M-indexes subsets $(J_{j_0}, J_{j_1}, J_{j_2}, ..., J_{j_{Ro}}, J_{j_{Ro+1}}, ..., J_{j_{n_y-1}})$; each element of J_{j_t} constitutes a node of level $t \in \{0,\ 1,\ 2\ , ..., n_y - 1\}$ in the network G_M^0. Within an initial enumeration of M-indexes subsets, let J_{j_t} and $J_{j_{t+1}}$ be two successive subsets of indexes. An edge $(i_1, i_2) \in G_M^0$ is such that $i_1 \in J_{j_t},\ i_2 \in J_{j_{t+1}}$ and $i_1 \neq i_2$. We now introduce the concept of M-tree and M-path as follows:

Definition 3.1. Let G_M^0 be an M-network; we define an **M-tree** with a root $i_0 \in J_{j_0}$ as a subset of vertices $i_n \in G_M^0$ such that there exists a path called **M-path**, connecting i_0 to i_n.

Walk on M-trees: Let A_M^0 be the set of M-trees, and let $T = \{i_0, i_1, ..., i_r\}$ be an M-path such that $i_0 \in J_{j_0}$; for $r > 0$, we have $i_r \in J_{j_r}$. An M-path T is such that $\alpha_i = 1$ for all $i \in T$and $\alpha_i \geq 0$ if $i \notin T$; this means that, at a given step k of the *ESLP* algorithm, the lower level variables are such that $i \in sp(y^k)$ $(dy_i = -y_i^{k-1})$ for all $i \in T$, and $y_i^k \geq 0$ $(dy_i \geq -y_i^{k-1})$ if $i \notin T$. In other words, the lower level constraints with indexes of slack variables $i \in T$are active at z^k. Moreover, by construction of the M-network, if we have for all $r \geq 0$ $i_r \in J_{j_r} \cap T$, then $\alpha_{i_r} = 1$ and $\alpha_i \geq 0$ for all $i \in J_{j_r}$ such that $i \neq i_r$.

The *ESLP* algorithm consists in applying a branch and bound method on sequential M-trees, using the link between each M-variable y_i and the corresponding binary variable α_i. Consider problem (2.1) and the corresponding subsets of M-indexes $J_j,\ \ j \in \{1, 2, ..., n_y\}$, as well as the M-property $\sum\limits_{i\in J_j} \alpha_i \geq 1$ attached to each M-indexes subsetJ_j. The enumeration uses M-indexes subsets according to a **dichotomous branching** rule: sequentially, for each M-indexes subset J_j, a single index $i_0 \in J_j$ is chosen and the corresponding boolean variable α_{i_0} is fixed to be equal to 1, whereas $\alpha_i \geq 0$ for all $i \in J_j,\ i \neq i_0$. This means that, after setting $i_0 \in sp(y)$, we check if this additional constraint leads to an improving rational solution. If y_{i_0} is a slack variable, the corresponding constraint is active. At optimality, active constraints of the lower level problem correspond to an M-indexes subset $T \subseteq \{i_1, i_2, ..., i_{n_y}\}$, where $i_j \in J_j,\ j \in \{1, 2, ..., n_y\}$. From M-constraints, the *ESLP* algorithm finds such subset T corresponding to an optimal solution.

Let $z^k = (x^k, y^k) \in \Omega$ be a feasible solution at iteration k. The *ESLP* algorithm uses a branch-and-bound with depth-first search technique within M-trees. Let A_m be an M-tree and T an M-path with nodes included in a subset P_a such that $P_a = \{i_0, i_1, i_2...\}$ with $i_0 \in J_{j_0}$ and $i_r \in J_{j_r},\ r \geq 1$. The *ESLP* algorithm solves sequentially $LPR(z^k, \{i_0\})$, $LPR(z^k, \{i_0, i_1\}), ..., LPR(z^k, P_a)$ where $z^k \in \Omega$. Let (dx, dy) be a solution of one of these problems. The backtracking is performed if we have one of the following conditions:

- i) an improving rational solution $(x^{k+1}, y^{k+1}) = (x^k + \sigma dx, y^k + \sigma dy)$ is computed and we have $T \subset sp(y^{k+1})$;
- ii) one of the above LP has no feasible solution;
- iii) (x^{k+1}, y^{k+1}) is a rational solution obtained from one of the above LP such that $f^1(x^{k+1}, y^{k+1}) > f^1(x^k, y^k)$.

Let n_{is} be the level of a node i_s in the network G_M^0, and A_M^0 the set of M-trees. If a backtracking condition is verified at a node i_s contained in the M-path T, then the walk along the corresponding M-tree is stopped. From the node i_s, we trace back indexes in

T. Let $T_B = \{i_s, i_{s-1}, ..., i_b\}$ be a subset of T containing backtracking nodes. Then, for the next LP to be solved in *ESLP* algorithm, the constraints $i \in sp(y)$ for $i \in T_B$ (i.e. $y_i = 0$ or $\alpha_i = 1$ for all $i \in T_B$) are relaxed and transformed into $y_i \geq 0$ for $i \in T_B$ (i.e. $\alpha_i \geq 0$ for all $i \in T_B$). From the node $i_{b-1} \in T$ including at least one successor which has not been investigated, other investigable M-paths are constructed. The walk procedure is stopped for an M-tree if there is no more investigable M-path from the root of such M-tree. Let

$$Ro = \min_{A_m \in A_M^0} \{n_{is} : i_s \in A_m\}.$$

Ro represents the smallest value on the set of levels of nodes corresponding to backtracking nodes i_s in the network G_M^0.

Remark 3.2. Let $\sum_{i \in J_j} \alpha_i \geq 1$ be an M-constraint and let J_j be an M-indexes subset. With a dichotomous branching rule, a single index $i_0 \in J_j$ is chosen and we set $\alpha_{i_0} = 1$. According to *Ro* and the backtracking rule, it is easy to show that, walks on the network consisting of the sequence of subsets $J_{j_0}, J_{j_1}, J_{j_2}, ..., J_{j_{Ro}}$ can not improve the value of the current solution. Let $J_{j_{Ro+1}}, J_{j_{Ro+2}}, ..., J_{j_{n_y-1}}$ be subsets for which some indexes have not been investigated by *ESLP* algorithm, due to backtracking or dichotomous branching rules; an investigation on some of the indexes in these subsets may improve the solution.

According to Remark 3.2, new relevant M-paths follow the construction of a sequence of M-networks $G_M^t,\ t = 1, 2, ..., n_y - Ro$as follows: we derive an M-network G_M^1 from the following ordered sequence of M-indexes subsets $J_{j_{Ro+1}}, J_{j_0}, J_{j_1}, J_{j_2}, ..., J_{j_{Ro}}, J_{j_{Ro+2}}, ..., J_{j_{n_y-1}}$; the numeration of levels of nodes in subsets $J_{j_i},\ i = 0, 1, 2, ..., n_y - 1$ is therefore updated, all the nodes of level $Ro + 1$ in the M-network G_M^0 become nodes of level metricconverterProductID1 in1 in the M-network G_M^1, and the nodes of level $2, 3, ..., Ro$ in G_M^1 are the nodes of level $1, 2, 3, ..., Ro - 1$ in G_M^0. In the same way, from the network G_M^1, we derive M-network G_M^2 from the following ordered sequence of M-indexes subsets $J_{Ro+2}, J_{Ro+1}, J_{j_0}, J_{j_1}, J_{j_2}, ..., J_{j_{Ro}}, J_{j_{Ro+3}}, ..., J_{j_{n_y-1}}$, and so on for the sequence of networks $G_M^t,\ t = 3, 4, ..., n_y - Ro$.

3.2 Principles of an Implicit Enumeration Algorithm

ESLP algorithm consists of walks through M-paths as defined in the previous subsection. More precisely, let P_a be a path in an M-tree A_m, and for some $\sigma > 0$ let $(x^{k+1}, y^{k+1}) = (x^k + \sigma dx, y^k + \sigma dy)$ be a rational solution such that (dx, dy) is an optimal solution of a linear programming problem $LPR(z^k, T),\ T \subset P_a$. The subset T is constructed gradually within the indexes of the set P_a. The evaluation carrying on the variables $y_i,\ i \in T$ consists of comparing the solution of the lower level problem $LP_2(x^{k+1})$ with (x^{k+1}, y^{k+1}). If the solution (x^{k+1}, y^{k+1}) is rational and improves the value of the upper level objective function f^1, then the variables $y_i,\ i \in T$ keep temporally the state related to the descent direction $dy_i,\ i \in T$, then we have $T \subset sp(y^{k+1})$. In this case, we say that the **evaluation** ($dy_t^k = -y_t^k,\ t \in T$) **corresponding to the set of variables** $y_i,\ i \in T$ **is successful**. Next, we select another path if either a backtracking condition is satisfied, or there exists at least one M-tree that has not been explored. In each stage of *ESLP* algorithm, the objective consists in the computation of an improving rational solution (x^{k+1}, y^{k+1}) such that $T \subset sp(y^{k+1})$. We write beside a node of an M-network, the index of the lower level variable related, as well as the triplet $(T, z^k, f^1(z^k))$ where T represents the subset used to solve the problem $LPR(z^k, T)$. An edge (i, j) has the letter S if, because of backtracking, one can no longer walk through the M-tree from the node i.

Remark 3.3. We consider the bilevel programming problem (2.1).

1) A node in an M-network corresponds to some partition of indexes (T, T^C) used solve the problem $LPR(z^k, T)$. Let N be the number of partitions (T, T^C) in the subset of indexes H.

2) There exist n_y M-indexes subsets, and each M-indexes subset has at most $m+1$ elements.

3) By construction, an M-network has at most N nodes. A walk on an M-tree is an elementary path of length p $(p = 1, 2, ..., n_y)$. Let N be the order of a network; according to its properties, the number of paths of length p is equal to $A_N^{p+1} = \frac{N!}{(N-p-1)!}$. Consequently, we may guess that this number of paths is equal to $O(N!)$.

4) Let A_I be an M-tree. The number of children of a node $i \in A_I$ written as $d_I(i)$, is the degree of that node. From the construction an M-network, we have $d_I(i) \leq m$. The number N_{AM} of M-trees with nodes i such that $d_I(i) = k$, where $i \in \{1, 2, ..., n_y\} \cup \{\bar{1}, \bar{2}, ..., \bar{m}\}$ is given by $N_{AM} = (N-1)^{(N-k-1)} C_{k-1}^{N-2}$(See [17]).

3.3 *ESLP* Algorithm

We consider the BPP problem (2.1); *valf* represents the current value of the upper level objective function, while a rational solution is $z^* = (x^*, y^*)$. Using notations introduced in the previous subsection, we present first the procedure $WALK(G_M^t, z^*)$ used in *ESLP* algorithm to investigate an M-network G_M^t; let A_M^t be the number of M-trees. The procedure $WALK(G_M^t, z^*)$ follows:

Procedure $WALK(G_M^t, z^*)$
If $t = 0$, set $Ro = n_y$. Set $M_T = A_M^t$(set of trees). While $M_T \neq \emptyset$, execute A-procedure.
A-Procedure:
Select an M-tree $A_m \in M_T$, then set $M_T = A_M^t \backslash A_m$ and
$CH_T = \{Ch \in A_m : Ch$ is an elementary path$\}$. While $CH_T \neq \emptyset$, execute B-procedure.
B-Procedure:
Select a path $Ch \in CH_T$, then set $P_a = Ch$,$CH_T = CH_T \backslash \{Ch\}$and $T = \emptyset$.
While (i) $P_a \neq \emptyset$ or (ii) a backtracking condition is not satisfied, execute C-procedure.
C-Procedure:
Select an index $i \in P_a$of level n_i in G_M^t, then set $P_a = P_a \backslash \{i\}$ and
$T = T \cup \{i\}$.
Let (dx, dy) be a solution of the program $LPR(z^k, T)$ (if it exists)
where $z^k \in \Omega$.
For some $\sigma > 0$, set $(x^{k+1}, y^{k+1}) = (x^k + \sigma dx, y^k + \sigma dy)$.
If $f^1(x^{k+1}, y^{k+1}) < f^1(x^k, y^k)$, solve
$LP(x^{k+1})$ to check if $(x^{k+1}, y^{k+1}) \in IR$. Set $k = k+1$.
Backtracking is performed on a node i_0 within the path Ch if:
(i) the program $LPR(z^k, T)$ does not have a solution.
(ii) $(x^{k+1}, y^{k+1}) \in IR$ with $f^1(x^{k+1}, y^{k+1}) < valf$. Set
$z^* = (x^*, y^*) = (x^{k+1}, y^{k+1})$,
and $valf = f^1(x^*, y^*)$.
(iii) $f^1(x^{k+1}, y^{k+1}) > f^1(x^k, y^k)$.
Remove from Ch the node i_0 as well as its children, then go out of C-procedure.
End of C-procedure.
End B-procedure.

If $t = 0$, set $Ro = \min(Ro, n_{i_0})$.

End of A-procedure. (Logout with a rational solution $z^* = (x^*, y^*)$ and the number Ro if $t = 0$).

ESLP algorithm now follows:

***ESLP* algorithm** (solving the BPP problem (2.1))

Step 0 (Initialization).

Set $k = 0$, compute a rational solution $\bar{z} = (\bar{x}, \bar{y})$ and set $valf = f^1(\bar{x}, \bar{y})$. For $j = 1, 2, ..., n_y$, compute the list L_M of all M-indexes subsets J_j.

For any subset J_j such that $|J_j| = 1$, set $y_t = 0$,with $\{t\} = J_j$; remove the subset J_j from the list L_M, then construct the M-network G^0_M.

Execute procedure $WALK(G^0_M, z^*)$ to compute an improving rational solution $z^* = (x^*, y^*)$ and the number Ro.

For $t = 1, 2, ..., n_y - Ro$, construct each of the M-network G^t_M.

Step 1 (Looking for subsets of lower level indexes leading to an improving rational solution).

For $t = 1, 2, ..., n_y - Ro$, execute the procedure $WALK(G^t_M, z^*)$.

Step 2 (Optimality). **Stop,** when Step 1 is completed, $z^* = (x^*, y^*)$ is an *ESLP* optimal solution of the problem (2.1).

Let z^* be a rational solution of the problem (2.1). The following proposition shows that each descent direction is computed by *ESLP* algorithm in a finite time.

Proposition 3.4. *We consider the BPP problem (2.1) such that assumption (H1) is satisfied. Let $(x^*, y^*) \in IR$ be a rational solution of a given step kof ESLP algorithm. We assume that there exists an improving rational solution $(\bar{x}, \bar{y})$. Hence, ESLP algorithm can compute an induced descent direction by means of solving a finite number of LP.*

Proof. Let $(x^*, y^*) \in IR$ be the current rational solution of a step k of *ESLP* algorithm and let $k_0 \geq k$. Given an M-path $T \subset H$, let (G^t_M), $t = 0, 1, 2, ...$ be a sequence of M-networks and let us consider the LP $PLR(z^{k_0}, T)$ defined by (2.7). Let $(\bar{x}, \bar{y})$ be an improving rational solution of the problem (2.1) computed from a given $z^{k_0} = (x^{k_0}, y^{k_0}) \in \Omega$. By hypothesis, there exists a path T such that $(\xi*, dx, dy)$ is an optimal solution of the LP problem $LPR(z^{k_0}, T)$ with $(x^{k_0+1}, y^{k_0+1}) = (x^{k_0} + \sigma dx, y^{k_0} + \sigma dy)$, for some $\sigma > 0$ and $f^1(x^{k_0+1}, y^{k_0+1}) < f^1(x^{k_0}, y^{k_0})$. According to Remarks 3.2, each M-network G^t_M has a finite number of trees, hence a finite number of M-paths. On each of such paths, we solve a finite number of LP $LPR(z^{k_0}, T)$ in order to compute an induced descent direction. When this problem has a solution, then one solves the LP$LP_2(x^{k_0})$ in order to verify that the new solution $(x^{k_0+1}, y^{k_0+1}) = (x^{k_0} + \sigma dx, y^{k_0} + \sigma dy)$ is rational. The conclusion follows. □

When the problem (2.4) has a solution, *ESLP* algorithm computes a rational solution by means of walks on all M-trees. From the induced solution (x^*, y^*) of the step k, *ESLP* algorithm leads to an *ESLP* optimal solution of the problem (2.1) in a finite number of steps. However, according to the proposition above, the solution $(\xi*, dx, dy)$ is computed by solving a finite number of LP $LPR(z^{k_0}, T)$ where $k_0 \geq k$. Moreover, we have for some $\sigma > 0$, $(\bar{x}, \bar{y}) = (x^{k_0+1}, y^{k_0+1}) = (x^{k_0} + \sigma dx, y^{k_0} + \sigma dy)$, (dx, dy) being an induced descent direction. A given set of indexes H has a finite number of partition (T, T^C). Any step of *ESLP* algorithm consists in examining partitions of indexes (T, T^C) such that $(\bar{x}, \bar{y}) \in IR$ and $T \subset sp(\bar{y})$. Problem (2.1) can then be solved in a finite number of steps.

Remark 3.5. If the functional f^1 is linear in (2.1), we have a linear bilevel programming problem. In this case, instead of solving $LPR(z^k, T)$ to compute a descent direction inside

the procedure $WALK(G_M^t, z^*)$, an improving rational solution is computed by solving the following LP:

$$\min_{x,y} f^1(x,y)$$
$$s.t. \begin{cases} Ax + By = b, \\ y_i \geq 0 \ \forall i \notin T, \\ y_i = 0 \ \forall i \in T, \\ x \in P, \end{cases}$$

where the subset of indexes T is built according to the previous M-analysis. Any partition (T, T^C) used by our algorithm corresponds in this case to an extreme point of the set of feasible solution problem (2.1).

4 An Example

Consider the following linear bilevel programming problem:

Example 1 (Data randomly generated according to Audet et al. [4])**.**

$$\min \ -10x_1 - 10x_2 - 7x_3 - 10x_4 + 5x_5 + 3x_6 + 2y_1 + 3y_2 + 4y_3 - 11y_4 - 3y_5 + 9y_6$$
$$s.t. \begin{cases} \min \ -y_1 + y_2 - 6y_3 + 5y_4 - 12y_5 - 10y_6 \\ s.t. \begin{cases} 6x_4 - y_5 + 15y_6 \leq 9, \\ -9x_1 + 3y_4 + 6y_6 \leq 11, \\ 18x_3 - 2x_4 + 5y_3 \leq 7, \\ -3x_3 \leq 14, \\ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + y_1 + y_2 + y_3 + y_4 + y_5 + y_6 \leq 6, \\ x_i, y_i \geq 0, i = 1, .., 6. \end{cases} \end{cases}$$

The resulting constraints when the lower level problem is in standard form are represented below.

$$\begin{cases} 6x_4 - y_5 + 15y_6 + y_{\bar{1}} = 9, \\ -9x_1 + 3y_4 + 6y_6 + y_{\bar{2}} = 11, \\ 18x_3 - 2x_4 + 5y_3 + y_{\bar{3}} = 7, \\ -3x_3 + y_{\bar{4}} = 14, \\ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_{\bar{5}} = 6, \\ -y_1 \leq 0, \\ -y_2 \leq 0, \\ -y_3 \leq 0, \\ -y_4 \leq 0, \\ -y_5 \leq 0, \\ -y_6 \leq 0, \\ x_i \geq 0, i = 1, .., 6. \end{cases}$$

A Boolean variable $\alpha_i, i = 1, 2, ..., 11$ corresponds to each equation in above system of constraints. The M-constraints follow:

$$\begin{cases} \alpha_5 \geq 1 \ ; \alpha_7 \geq 1; \ \alpha_3 + \alpha_5 \geq 1; \\ \alpha_1 + \alpha_9 \geq 1; \ \alpha_5 \geq 1; \ \alpha_1 + \alpha_2 + \alpha_3 \geq 1. \end{cases}$$

Then, the M-indexes subsets are:

$$J_1 = \{\bar{5}\}, \ J_2 = \{2\}, \ J_3 = \{\bar{3}, \bar{5}\}, \ J_4 = \{4, \bar{1}\}, \ J_5 = J_1, \ J_6 = \{\bar{1}, \bar{2}, \bar{5}\}.$$

At the optimality, we have$y_2 = y_{\bar{5}} = 0$. But, in order to illustrate how *ESLP* algorithm works, we do not take into account this implication. The initial solution is $z^0 = (0,0,0,0,0,0,0,0,0,0,6,0)$ and $valf^0 = -18$. The values of the slack variables are not indicated in this solution. The M-network G_M^0 is constructed form the ordered sequence of M-indexes subsets $(J_1, J_2, J_3, J_4, J_6)$. The adjacency-matrix representation of G_M^0 is given on Table 4.1, while Figure 4.1 represents the corresponding tree.

1) Walks on the first M-tree: the resolution of the program $LPR(z^0, T_2)$ on path $T_2 = \{\bar{5}, 2, \bar{3}, 4\}$, gives the following rational solution:

$$z^1 = (1.9918, 3.6194, 0.3889, 0, 0, 0, 0, 0, 0, 0, 0, 0) \text{ and } valf^1 = -58.8333.$$

A backtracking is then performed on the node 4. On path $T_3 = \{\bar{5}, 2, \bar{3}, \bar{1}\}$, the resolution of the program $LPR(z^1, T_3)$ gives the following rational solution:

$$z^2 = (0, 0, 0.6172,\ 2.0547, 0, 0, 0, 0, 0, 3.3281, 0, 0) \text{ and } valf^2 = -61.4766.$$

On paths $T_4 = \{\bar{5}, 2, 4\}$, $T_5 = \{\bar{5}, 2, \bar{1}\}$, the programs $LPR(z^2, T_4)$ and $LPR(z^2, T_5)$ have no feasible solution. Walks on the first M-tree are completed, and $Ro = 4$.

	$\bar{5}$	2	$\bar{3}$	$\bar{5}$	4	$\bar{1}$	$\bar{1}$	$\bar{2}$	$\bar{5}$
$\bar{5}$	0	1	0	0	0	0	0	0	0
2	0	0	1	1	0	0	0	0	0
$\bar{3}$	0	0	0	0	1	1	0	0	0
$\bar{5}$	0	0	0	0	1	1	0	0	0
4	0	0	0	0	0	0	1	1	1
$\bar{1}$	0	0	0	0	0	0	0	1	1
$\bar{1}$	0	0	0	0	0	0	0	0	0
$\bar{2}$	0	0	0	0	0	0	0	0	0
$\bar{5}$	0	0	0	0	0	0	0	0	0

Table 4.1: Exemple2 – adjacency matrix representation of the initial M-network

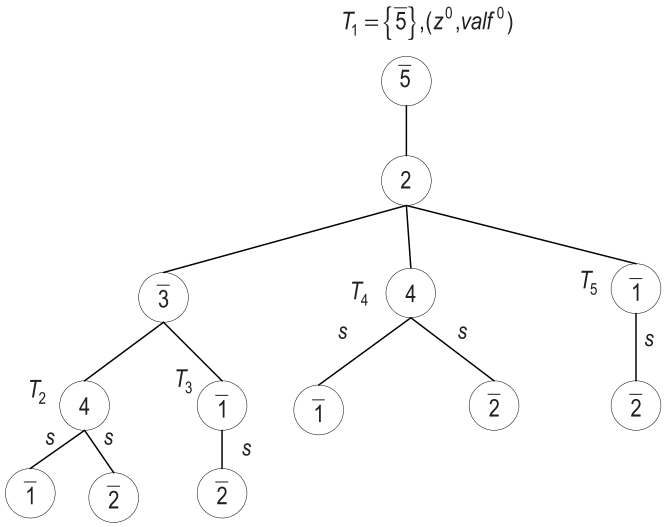

Figure 4.1: Example2 – first M-tree

As $Ro = 4$, the M-network G_M^1, is constructed from the ordered sequence subsets of indexes $(J_6, J_1, J_2, J_3, J_4)$ as well as paths including combinations of indexes from these subsets. Table 4.2 is the adjacency-matrix representation of G_M^1. Figure 4.2 includes the corresponding M-trees.

2) Figure 4.2 represents paths on the second M-network: on tree (1), the resolution of the program $LPR(z^2, \{\bar{1}\})$ gives the following rational solution

$$z^3 = (0.0494, 0, 0, 2.1358, 0, 0, 0, 0, 0, 3.8148, 0, 0) \text{ and } valf^3 = -63.8148.$$

	$\bar{1}$	$\bar{2}$	$\bar{5}$	$\bar{5}$	2	$\bar{3}$	$\bar{5}$	4	$\bar{1}$
$\bar{1}$	0	0	0	1	0	0	0	0	0
$\bar{2}$	0	0	0	1	0	0	0	0	0
$\bar{5}$	0	0	0	0	0	0	0	0	0
$\bar{5}$	0	0	0	0	1	0	0	0	1
2	0	0	0	0	0	1	1	0	0
$\bar{3}$	0	0	0	0	0	0	0	1	1
$\bar{5}$	0	0	0	0	0	0	0	1	1
4	0	0	0	0	0	0	0	0	0
$\bar{1}$	0	0	0	0	0	0	0	0	0

Table 4.2: Example2**,** adjacency matrix representation of the second M-network.

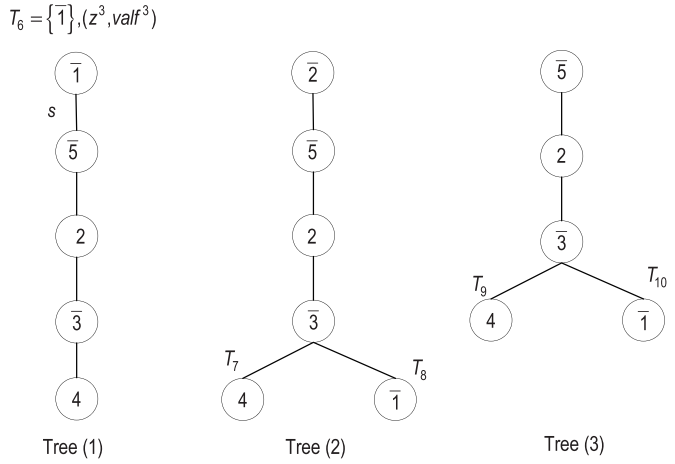

Figure 4.2: Example2 – M-trees of the second M-network.

The walk of this tree is stopped as a rational solution is computed. On tree (2), the programs $LPR(z^3, T_7)$ and $LPR(z^3, T_8)$ where $T_7 = \{\bar{2}, \bar{5}, 2, \bar{3}, 4\}$ and $T_8 = \{\bar{2}, \bar{5}, 2, \bar{3}, \bar{1}\}$ have no feasible solution. The exploration of the tree (2) is then stopped. On the tree (3), let $T_9 = \{\bar{5}, 2, \bar{3}, 4\}$ and $T_{10} = \{\bar{5}, 2, \bar{3}, \bar{1}\}$; the programs $LPR(z^3, T_9)$ and $LPR(z^3, T_{10})$ have no feasible solution. The exploration of the tree (3) is also stopped and the exploration on the network G_M^1 is completed. Hence, z^3 is an *ESLP* optimal solution (z^3 is an optimal solution).

5 Computational Experience with *ESLP* Algorithm

We recall that *ESLP* algorithm solves two types of linear programs at each step:

- First, the relaxed linear program $LPR(z^k, T)$, $T \subseteq P$ ($P \subset \bigcup_{1 \le j \le n_y} J_j$) computes a direction of displacement (dx, dy) of the current step k. For a given problem, the total number of descent directions calculated is designated by *Noded*; this quantity represents the number of relaxed LP $LPR(z^k, T)$ solved by the *BM* algorithm.

- Secondly, if (dx, dy) is a solution of the relaxed LP problem, that is $LPR(z^k, T)$, it is necessary to verify that the solution $(x^k + \sigma dx, y^k + \sigma dy)$ is rational by solving the lower level problem $LP_2(x^{k+1})$. For a given problem, *Node* represents the number of times this verification occurs for a given problem, whereas *Nodea* represents the number of the rational solutions contributing to improve the value of the upper level objective function. The columns of the table representing the values of *Node* and *Noded* include two numbers: the smaller value corresponds to the stage of the algorithm computing the optimal solution, whereas the higher value designates the number of programs solved. The performances measure include *CPU* time (seconds) are represented in the same way.

ESLP algorithm was coded in MATLAB and all computations were performed on a PC Pentium 4 (processor 3.2 GHZ, 1.24 GB of stocktickerRAM). We solved some small size and medium size problems with data randomly generated: i) as [4], and ii) by the method introduced by Vicente and Calamai [50]. We compare the results computed by *ESLP* algorithm with those obtained by the algorithm of Bard and Moore [7] (*BM*) and the *CBB* algorithm developed by Audet et al. [3].

5.1 Solving Small Size Problems

First, we illustrated *ESLP* algorithm on the following problems with quadratic upper level objective function.

Problem 1.

$$\min_{x,y\geq 0} -8x_1^2 - 4x_2^2 + 8y_1^2 - 10y_2^2 - 20y_3^2$$
$$s.t.\begin{cases} x_1 + 2x_2 \leq 2, \\ 2x_2 \leq 1, \\ \min_y 2y_1 - y_2 + 2y_3 \\ s.t.\begin{cases} -y_1 + y_2 + y_3 \leq 1, \\ 2x_1 - y_1 + 2y_2 - 0.5y_3 \leq 1, \\ 2x_2 + 2y_1 - y_2 - 0.5y_3 \leq 1. \end{cases} \end{cases}$$

Problem 2.

$$\min_{x,y\geq 0} 2x^2 + 12y^2$$
$$s.t.\begin{cases} \max_y y \\ s.t.\begin{cases} x + y \geq 8, \\ -3x + 2y \leq 6, \\ 3x + 4y \leq 48, \\ 2x - 5y \leq 9. \end{cases} \end{cases}$$

Problem 3.

$$\min_{x,y\geq 0} -4x_1^2 - 2x_2^2 + 2y_1^2 - 20y_2^2 - 2y_3^2$$
$$s.t.\begin{cases} x_1 + x_2 - y_3 \leq 1.3, \\ \min_y 2y_1 + y_2 + y_3 \\ s.t.\begin{cases} -y_1 + y_2 + y_3 \leq 1, \\ 4x_1 - 2y_1 + 4y_2 - y_3 \leq 2, \\ 4x_2 + 4y_1 - 2y_2 - y_3 \leq 2. \end{cases} \end{cases}$$

Problems	(x*,y*)	F*	f*	Ro	Node	Nodea	Noded
Problem 1	(1.5,0,1,0,2)	-90	6	1	11	4	15
					11		22
Problem 2	(12,3)	396	3	1	3	3	3
					3		3
Problem 3	(1.5,0,1,0,2)	-15	6	1	3	3	3
					3		30

Table 5.1: *ESLP* algorithm – results on small size problems.

Problems	(x*,y*)	F*	f*	Node	Nodea	Noded
Problem 1	(1.5,0,1,0,2)	-90	6	2	2	2
				2		3
Problem 2	(12,3)	396	3	10	2	17
				12		23
Problem 3	(1.5,0,1,0,2)	-15	6	5	2	6
				21		41

Table 5.2: *Bard and Moore's* algorithm – results on small size problems

Table 5.1 and Table 5.2 contain the computational results of small size problems solved respectively with *ESLP* and *BM* algorithms (the solutions are the same); (x^*, y^*) represents the optimal solution, while F^* and f^* represent respectively the optimal values of the upper level and the lower level objective functions. *ESLP* optimal solutions computed by our algorithm are optimal solutions for the above small size problems.

5.2 Solving Medium Size Problems

We considered three types of problems:

A) Data are randomly generated as in Audet et al. [4] with 8% of density: the elements of B^e are chosen between -20 and 20, while those of A are taken in the interval [0 20]. The coefficients of c^1 belong to the interval $[-10\ 10]$ and those of d^1 belong to [10 20]. The elements of d^2 are chosen between 0 and 10. The elements of the second member of constraints b are randomly chosen between -40 and 40. Finally, the additional lower level constraint $\sum_{j=1}^{n_x} x_j + \sum_{j=1}^{n_y} y_j \leq n_x + n_y$ is added to obtain bounded relaxed problem. The vector (n, n_y, m) characterizes the size of each of the solved problems. The results are presented in Table 5.3 and Table 5.4. We noticed that the solution computed by *ESLP* algorithm was better that the solutions computed by the *BM* algorithm (with CPU= 10000) even when using the stopping criteria $CPU \geq 20000$ for the problem of size $(n, n_y, m) = (70, 35, 30)$.

n	n_y	m	Ro	$Node$	$Nodea$	$Noded$	CPU	F^{ESLP}
70	35	30	7	8534	12	13398	1139	418.92
				11176		17887	1517	
90	45	30	28	48	6	60	6.17	563.78
				2459		6661	459	
100	33	50	8	63	10	8871	682	97.85
				63		10001	10001	
100	30	45	11	118	5	259	26	328.49
				3631		6413	800	
110	40	40	1	1084	10	9983	797	480.66
				7685		62232	5297	
120	33	33	1	3981	9	5128	452	665.25
				14378		19327	1750	
120	60	30	32	9966	8	171515	9341	671.2
				10219		182987	10001	

Table 5.3: *ESLP* algorithm – Problems of Audet et al. [4]

n	n_y	m	Node	Nodea	Noded	CPU	F^{ESLP}
70	35	30	139606	10	279164	14827	411.05
			188599		3777153	20001	
90	45	30	460	5	884	80	558.4
			55698		111360	10001	
100	33	50	76	3	104	18	97.85
			76		149	22	
100	30	45	70	2	95	12	324
			41813		83586	10001	
110	40	40	1120	5	2210	337	480.66
			36240		72442	10001	
120	33	33	7006	5	13974	1084	641.38
			67496		134957	10001	
120	60	30	26630	8	53221	6846	671.20
			40131		80210	10001	

Table 5.4: *Bard and Moore's* algorithm– Problems of Audet et al. [4]

B) For medium size problems generated as in [3], we compared the computational results obtained by *ESLP* algorithm, *BM* algorithm and *CBB* algorithm. The comparison was carried on seven problems of size $(n, n_y, m) = (70, 35, 20)$ where $n = n_y + n_x$. Table 5.5 and Table 5.6 present the results of this study; F^{ESLP}, F^{CBB} and F^{BM} represent respectively the value of the upper level objective function computed by *ESLP*, *CBB* and *BM* algorithms. For all the problems solved, comparative studies of Table 5.5 and Table 5.6 show that *ESLP* algorithm is efficient: the *ESLP* optimal solution on Example 6 is better than the solution computed by the *CBB* algorithm; the same observation was made on the problem with $(n, n_y, m) = (70, 35, 30)$ of Table 5.3 for which we find $F^{ESLP} = 418.92$ and $F^{CBB} = 391.11$. Using stopping criteria $CPU \geq 10000$ in Example 4 of Table 5.5, the solution computed by *ESLP* algorithm is such that $F^{ESLP} = 530.81$ and $F^{ESLP} = F^{ESLP} = 532.23$. While forcing *ESLP* algorithm to calculate an 7th improving rational solution for this problem, we found $F^{ESLP} = 531.31$ with a $CPU = 83794$, $Neud = 100522$ and $Neudd = 438779$.

We mention that *ESLP* algorithm was coded in MATLAB, and the computational experiments carried out on a PC computer, while linear programs are solved by an interior point method. Whereas the *CBB* algorithm coded in C, uses the CPLEX 8.1 library to solve linear programs and computational experiments carried out on a ULTRA 60 station under Solaris 2.7-05. Numerical results attached to each computational environment could constitute one of the explanations attached to these differences, besides the stopping criteria $CPU \geq 10000$ for exemple4.

In general, it could be noted that *ESLP* algorithm is faster to lead to a step corresponding to an optimal solution, compared to the *BM* algorithm, at least 9 times on 14. Besides, the *BM* algorithm was not able to solve the problem of Example 7, even with the stopping criteria $CPU \geq 20000$.

Problems	*Ro*	*Node*	*Nodea*	*Noded*	*CPU*	F^{ESLP}	F^{CBB}
Example1	12	968	3	1423	75	574.68	576.63
		10709		16000	836		
Example2	17	6086	4	19024	970	656.55	656.55
		62181		214452	10001		
Example3	13	27	2	28	20.7	599.93	599.93
		61136		179918	9419		
Example4	21	15949	6	67683	7882	530.83	532.23
		21078		88090	10001		
Example5	12	93	2	112	20.7	479.17	479.17
		65431		77253	10001		
Example6	18	4561	11	7960	1197	492.69	486.27
		448323		76244	10001		
Example7	14	50	2	80	12.5	632.46	632.46
		6154		13704	1933		

Table 5.5: Comparative tests - *ESLP* and *CBB* algorithms

Problems	*Node*	*Nodea*	*Noded*	*CPU*	F^{BM}	F^{ESLP}	F^{CBB}
Example1	25601	6	51173	2615	574.68	574.68	576.63
	66219		232411	10001			
Example2	100	3	177	177	656.55	656.55	656.55
	130977		261936	10001			
Example3	172	4	320	14	599.93	599.93	599.93
	83365		166725	7601			
Example4	4494	4	8964	393	532.23	530.83	532.23
	96329		192639	10001			
Example5	1580	4	3136	136	479.17	479.17	479.17
	77128		154234	10001			
Example6	137	3	251	251	492.69	492.69	486.27
	84142		168264	10001			
Example7	1	1	1	0	613.95	632.46	632.46
	52258		1	10001			

Table 5.6: Comparative tests – *Bard and Moore*, *ESLP* and *CBB* algorithms

C) We consider the linear bilevel programming problems (with minimization criteria) constructed by Vicente and Calamai [50]; we fixed $\rho_i = \text{mod}(i, 5) + 4,\ \ i = 1, 2, ..., \frac{m}{3}$. The solution of this BPP problem is known in advance.

Table 5.7 presents the results computed by *ESLP* algorithm: F^{THEO} represents the theoretical optimal value of the upper level objective function. These problems proved to be the most difficult to solve in terms of the computational time and the number of linear programs solved in each step. In Table 5.7, we present the results computed by the *ESLP* algorithm for $n \geq 50$, using the stopping criterions $CPU \geq 10000$, $CPU \geq 20000$ or $CPU \geq 30000$. We noted that the value of the solution computed by the *ESLP* algorithm using the stopping criteria $CPU \geq 30000$, was improved compared to the solution computed with stopping criterions $CPU \geq 10000$ or $CPU \geq 20000$.

n	n_y	m	Ro	$Node$	$Nodea$	$Noded$	CPU	F^{ESLP}	F^{THEO}
20	10	40	10	46	4	85	4.5	28	28
				48		87	4.7		
30	15	60	15	632	5	1461	83	42	42
				635		1464	84		
40	20	80	20	11192	6	26749	1236	56	56
				11197		26754	1237		
50	24	98	24	159443	5	390754	19024	69	69
				160345		401360	20001		
60	29	118	29	195567	6	473466	26857	84	83
				220543		535772	30001		
70	23	116	23	65917	6	164674	8205	65	65
				65923		164680	8206		
70	35	140	35	11485	5	27187	3193	101	98
				36455		87060	10001		
80	27	134	27	28617	6	69845	8240	76	75
				72330		176141	20001		
80	39	158	39	187668	7	386724	25612	114	111
				186068		455341	30001		

Table 5.7: *ESLP* algorithm - Problems of Vicente and Calamai [50]

For example, we got $F^{ESLP} = 70$ using the stopping criteria $CPU \geq 10000$, with $n = 50$ and $CPU = 1906$. On the other hand with the stopping criteria $CPU \geq 20000$, we found $F^{ESLP} = F^{THEO} = 69$ with $CPU = 19024$.

In linear cases, the *BM* or *ESLP* algorithms can prematurely be stopped before the global optimal solution is computed as we do not have an efficient stopping criterion. Moreover, according to our computational results, we realized that for some problems, a considerable number of iterations are performed, whereas the optimal solution has already been found. Here, we considered a compromise between the computational time and the best solution computed. With the stopping criteria $CPU \geq 10000$, the capacity of the *ESLP* algorithm (i) to compute several rational solutions faster or (ii) to find at least 96% of the value of a global optimum shows that this algorithm is not only worthy of interest, but can also be improved. The improvement concerns the walk strategies on M-trees. The reliability of *ESLP* algorithm depends on the sharpness in the construction and the walks trough the series of M-networks G_M^t. Besides, we realized that, if $\max(n, m) \leq 100$, *ESLP* algorithm computed a global optimum on almost all the problems solved. We also note that *ESLP* algorithm presents some similarities with the *HJS* algorithm of Hansen et al. [27]: in the *ESLP* algorithm, the elimination of some lower level variables is done implicitly thanks to the walks on M-trees. Finally, the size of the linear programs solved by *ESLP* algorithm is lower than the size of those solved by *BM* algorithm.

6 Conclusion and Final Remarks

The *ESLP* algorithm uses M-networks and investigates the support of active set constraints sets associated to the KKT formulation of problem (2.1) and performs well in practice; it is a compromise that leads to a good evaluation of the optimal solution of a nonlinear BPP problem with linear constraints. In this paper, no theoretical results have been established concerning the termination of our algorithm; therefore the *ESLP* algorithm should now be

seen as a heuristic technique for processing a BLP. Further research needs to be done on the efficiency of using monotonicity constraints within networks which guarantee in theory a global optimum for for linear case, as well as a generalization to the nonlinear cases of the sequential linear programming algorithms developed in this paper. Note that quasi concave bilevel programs can be solved by using an enumeration sequential quadratic programming algorithm, based on the same framework developed in this paper. Another perspective of research may consist in solving a mathematical programming problem with equilibrium constraints by an enumeration sequential linear programming algorithm that exploits efficiently the disjunctive nature of constraints in this problem.

Acknowledgment

The author would like to thank Professor Gilles Savard[†]bec), H2C 3A7 Canada (Gilles.Savard @gerad.ca). and anonymous referees for their constructive review and useful comments or remarks that help him very much in revising and improving this paper.

References

[1] H. Abdou-Kandil and P. Bertrand. Government-private sector relations as a Stackelberg game: a degenerate case, *Journal of Economical Dynamics and Control* 11 (1987) 513–517.

[2] C. Audet, P. Hansen, B. Jaumard and G. Savard, Links between linear bilevel and mixed 0-1 programming problems, *Journal of Optimization Theory and Applications* 93 (1997) 273–300.

[3] C. Audet, G. Savard and W. Zghal, New branch-and-cut algorithm for bilevel linear programming, *Journal of Optimization Theory and Applications* 134 (2007) 353–370.

[4] C. Audet, P. Hansen, B. Jaumard and G. Savard, A symetrical linear maximin approach to disjoint bilinear programming, *Mathematical Programming* 85 (1999) 573–592.

[5] J.F. Bard, An algorithm for solving the general bilevel programming problem, *Mathematics of Operations Research* 8 (1983) 260–272.

[6] J.F. Bard, Optimality conditions for the bilevel programming problem, *Naval Research Logistics Quarterly* 31 (1984) 13–26.

[7] J.F. Bard and J.T. Moore, A branch and bound algorithm for the bilevel programming problem, *SIAM J. Sci. Stat. Comput.* 11 (1990) 281–292.

[8] O. Ben-Ayed and C. Blair, Computational difficulties of bilevel linear programming, *Operations Research* 38 (1990) 556–560.

[9] H.P. Benson, On the structure and properties of a linear multilevel programming problem, *Journal of Optimization Theory and Applications* 60 (1989) 353–373.

[10] W.F. Bialas and M.H. Karwan, Two-level linear programming, *Management Science* 30 (1984) 1004–1020.

[†]MAGI and GERAD, École Polytechnique de Montréal, C.P. 6079, succ. Centre-ville Montréal (Qu{e

[11] L. Brotcorne, M. Labbé, P. Marcotte and G. Savard, A bilevel model and solution algorithm for a freight tariff setting problem, *Transportation Science* 34 (2000) 289–302.

[12] H.I. Calvette and C. Galé, On the quasiconcave bilevel programming problem, *Journal of Optimization Theory and Applications* 98 (1998) 613–622.

[13] M. Campêlo, S. Dantas and S. Scheimberg, A note on a penalty function approach for solving bilevel linear programs, *Journal of Global Optimization* 16 (2000) 245–255.

[14] W. Candler and R. Norton, Multi-level programming, World Bank Development Research Center Discussion Paper, 20, Washington DC, January, 1977a.

[15] W. Candler and R. Norton, Multi-level programming and development policy, World Bank Development Research Center Discussion Paper, 258, Washington DC, May 1977b.

[16] P.A. Clark and A.W. Westerberg, A note on the optimality conditions for the bilevel programming problem, *Naval Research Logistics* 3 (1988) 414–418.

[17] L.E. Clarke, On Cayley's formula for counting trees, *Proc., Cambridge Phil. Soc.* 59 (1963) 509–517.

[18] B. Colson, P. Marcotte and G. Savard, Bilevel programming: a survey, *4OR* 3 (2005) 87–107.

[19] S. Dempe, A Necessary and sufficient optimality for bilevel programming problem, *Optimization* 2 (1992) 341–354.

[20] S. Dempe, Annoted bibliography on bilevel programming and mathematical programs with equilibrium constraints, *Optimization* 52 (2003) 333–359.

[21] J.B. Etoa Etoa, *Contribution à la résolution des programmes mathématiques à deux niveaux et des programmes mathématiques avec contraintes d'équilibre*, PhD Thesis, École Polytechnique de Montréal, December 2005.

[22] J.B. Etoa Etoa. *Optimisation hiérarchique : théorie, algorithmes et applications,* Publibook (Eds. Sciences Mathématiques) 14, rue des Volontaires, 75015 Paris, France, www.publibook.com. 2007.

[23] J.B. Etoa Etoa. Solving convex quadratic bilevel programming problems using an enumeration sequential quadratic programming algorithm. *Journal of Global Optimization*, DOI 10.1007/s10898-009-9482-3 (2009).

[24] F. Facchinei, A. Fisher and C. Kanzow. On the accurate identification of active constraints, *SIAM Journal on Optimization* 9 (1998) 14–32.

[25] J.E. Falk and J. Liu, On bilevel programming, part I: General nonlinear case, *Mathematical Programming* 70 (1995) 47–72.

[26] Z.H. Gümüs and C.H. Floudas, Global optimization of nonlinear bilevel programming problems, *Journal of Global Optimization* 20 (2001) 1–31.

[27] P. Hansen, B. Jaumard and G. Savard, New branch-and-bound rules for linear bilevel programming, *SIAM Journal on Scientific and Statistical Computing* 13 (1992) 1194–1217.

[28] A. Haurie, G. Savard and D. White, A note on an efficient point algorithm for a linear two-stage optimization problem, *Operations Research* 38 (1990) 553–555.

[29] R.G. Jeroslow, The Polynomial hierarchy and simple model for competitive analysis, *Mathematical Programming* 32 (1985) 146–164.

[30] J. Jùdice and A.M. Faustino, A sequential stockticker LCP method for bilevel linear programming, *Annals of Operations Research* 34 (1992) 86–106.

[31] C.D. Kolstas and L. Lasdon, Derivative evaluation and computational experience with large bilevel mathematical programs, *Journal of Optimization Theory and Applications* 65 (1990) 6485–499.

[32] J. Kornaj and T. Liptak, Two-level planning, *Econometrica* 33 (1965) 141–169.

[33] J. Lu, C. Shi and G. Zhang, An extended kth best approach for linear bilevel programming, *Applied Mathematics and Computation* 164 (2005) 843–855.

[34] J. Lu, C. Shi and G. Zhang, An extended Kuhn-Tucker approach for linear bilevel programming, *Applied Mathematics and Computation* 162 (2005) 51–63.

[35] J. Lu, C. Shi and G. Zhang, On the definition of linear bilevel programming, *Applied Mathematics and Computation* 160 (2005) 169–176.

[36] Z.Q. Luo, J.S. Pang and D. Ralph, *Mathematical Programs with Nonlinear Complementary Constraints*, University Press, Cambridge, UK, 1996.

[37] P. Marcotte, G. Savard and D. L. Zhu, A trust region algorithm for nonlinear bilevel programming, *Operations Research Letters* 29 (2001) 171–179.

[38] P. Marcotte, and G. Savard, *Bilevel Programming Applications*, Encyclopedia of Optimization: Kluwer Academic Publishers, Dordrecht, 2001.

[39] P. Marcotte, and G. Savard, Bilevel Programming in *A Combinatorial Perspective, Graph Theory and Combinatorial Optimization*, D. Avis, A. Hertz, O. Marcotte (eds), Springer, Boston, 2005.

[40] A.G. Mersha and S. Dempe, Linear bilevel programming with upper level constraints depending on the lower level solution, *Applied Mathematics and Computation* 180 (2006) 247–254.

[41] M. Minoux, *Programmation Mathématique: Théorie et Algorithmes Tome* 1: Paris, Dunod Bordas et C.N.E.T, E.N.S.T., 1983.

[42] L.D. Muu and N.V. Quy, A global optimization method for solving convex quadratic bilevel programming problems, *Journal of Global Optimization* 26 (2003) 199-219.

[43] O. Pironneau and E. Polak, Rate of convergence of a class of methods of feasible directions, *SIAM Journal on Numerical Analysis*, 10 (1973) 161–174.

[44] G. Savard, *Contribution à la programmation mathématique à deux niveaux*, PhD thesis, École polytechnique de Montréal, 1989.

[45] G. Savard and J. Gauvin, The steepest descent direction for the nonlinear bilevel programming problem, *Operations Research Letters* 1 (1994) 265–272.

[46] M. Simaan and J.B. Cruz, On the Stackelberg strategy in nonzero-sum games, *Journal of Optimization Theory and Applications* 11 (1973) 533–555.

[47] G. Still, Linear bilevel problems: genericity results and efficient method for Computing local minima, *Mathematical Methods of Operations Research* 5 (2002) 383–400.

[48] D.M. Topkis and A.F. Veinott, On the convergence of some feasible directions algorithms for nonlinear programming, *SIAM Journal on Control* (1967) 268–279.

[49] L. N. Vicente and P. H. Calamai, Bilevel and multilevel programming: a bibliography review, *Journal of Global Optimization* 5 (1994) 291–306.

[50] L.N. Vicente and P.H. Calamai, Generating quadratic bilevel programming test problems, *ACM Transactions on Mathematical Software* 20 (1994) 103–119.

[51] D. Wilde, Monotonicity and dominance in optimal hydraulic cylinder design, *Journal of Engineering for Industry* 97 (1975) 13–26.

Manuscript received 13 February 2009
revised 13 July 2009, 24 November 2009
accepted for publication 24 November 2009

Jean Bosco Etoa Etoa
High Commission for the Republic of Cameroon
170 Clemow Avenue Ottawa (ON) K1S 2B4 Canada
E-mail address: jbetoa_etoa@hotmail.com