Journal of Global Optimization (2006) 36: 89–114
DOI 10.1007/s10898-006-9001-8

A Complementarity-based Partitioning and Disjunctive Cut Algorithm for Mathematical Programming Problems with Equilibrium Constraints

JOAQUIM J. JÚDICE[1,*], HANIF D. SHERALI[2,‡], ISABEL M. RIBEIRO[3] and ANA M. FAUSTINO[3]
[1]*Departamento de Matemática da Universidade de Coimbra and Instituto de Telecomunicações, Coimbra, Portugal (e-mail: joaquim.judice@co.it.pt)*
[2]*Grado Department of Industrial & Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, United States (e-mail: hanifs@vt.edu)*
[3]*Secção de Matemática do Departamento de Engenharia Civil, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal (e-mail: iribeiro@fe.up.pt, afausti@fe.up.pt)*

(Received 14 June 2005; accepted in revised form 20 January 2006)

Abstract. In this paper a branch-and-bound algorithm is proposed for finding a global minimum to a Mathematical Programming Problem with Complementarity (or Equilibrium) Constraints (MPECs), which incorporates disjunctive cuts for computing lower bounds and employs a Complementarity Active-Set Algorithm for computing upper bounds. Computational results for solving MPECs associated with Bilivel Problems, NP-hard Linear Complementarity Problems, and Hinge Fitting Problems are presented to highlight the efficacy of the procedure in determining a global minimum for different classes of MPECs.

Key words: active-set algorithm, branch-and-bound method, complementarity, disjunctive cuts, global optimization

1. Introduction

In this paper we address the Mathematical Programming Problems with Complementarity (or Equilibrium) Constraints (MPECs) when all the constraints but the complementarity are linear and the objective function is convex on the convex set defined by the linear constraints. Specifically, we focus on the following problem

* Support for this author was provided by *Instituto de Telecomunicações* and by *FCT* under grant POCTI/35059/MAT/2000.
‡ Support for this author was provided by the *National Science Foundation* under grant DMI-0094462.

$$\textbf{MPEC:}\quad \begin{array}{ll} \text{Minimize} & f(y,z), \\ \text{subject to} & Ew = q + Mz + Ny, \\ & z \geqslant 0, \quad w \geqslant 0, \quad y \in K_y, \\ & z^T w = 0, \end{array} \tag{1}$$

where $q \in \mathbb{R}^l$, $z, w \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $E \in \mathbb{R}^{l\times n}$, $M \in \mathbb{R}^{l\times n}$, $N \in \mathbb{R}^{l\times m}$, $f : \mathbb{R}^{m\times n} \to \mathbb{R}$ is twice continuously differentiable on an open set that contains the feasible region associated with the linear constraints of the problem, and $K_y \subseteq \mathbb{R}^m$ is a convex polyhedron in y given by

$$K_y = \left\{ y \in \mathbb{R}^m : Ay = b,\ y \geqslant 0 \right\},$$

with $A \in \mathbb{R}^{p\times m}$ and $b \in \mathbb{R}^p$. Note that inequality constraints in the definition of K_y can always be reduced to equalities by introducing slack variables and accommodating them within (1). In many applications of MPEC the matrices E and M are square, with E the identity matrix and M a positive semi-definite (PSD) or positive definite (PD) matrix [8, 11, 13, 20]. Moreover, we assume that the function f is convex on the convex set defined by the linear constraints.

The MPEC (1) is an NP-hard optimization problem, since the determination of a feasible solution for MPEC consists of solving a Generalized Linear Complementarity Problem (GLCP), which is NP-hard [15]. Due to this, we expect some sort of enumeration to be required for finding a global minimum to such a problem. In the last few decades, several algorithms of this sort have been proposed for the MPEC. Among these, branch-and-bound algorithms [2, 9], a penalty technique [24] and a sequential complementarity method [12] are considered to be the most efficient procedures to perform this task. A number of local methods [6–8, 10, 14, 16, 22] have also been recently developed to find stationary points for the MPEC. Among these techniques, the complementarity active-set algorithm introduced in [14] is particularly recommended to process the MPEC (1) when E is the identity matrix and M is a PSD matrix or can be transformed into an MPEC having this property. The algorithm employs an active-set methodology, maintains complementarity during the entire procedure, and is shown to be quite efficient in practice for finding stationary points to the MPEC (1).

In this paper a new branch-and-bound algorithm for the MPEC (1) with a convex function f is introduced. The algorithm employs the complementarity active-set algorithm to compute upper-bounds and disjunctive cuts to find lower-bounds. An interesting feature of this algorithm that distinguishes it from other branch-and-bound methods is that branching is done at the stationary points that are achieved during the procedure. Due to the characteristics of the MPEC at hand, it is shown that the new branch-and-bound method can be implemented within an active-set code such as MINOS [17]. Computational experience with the algorithm is reported for solving MPECs associated with Bilevel Programs (BP) [12], Bilinear Programming formulations of LCPs [11, 13], and Hinge-Fitting Problems (HFP) [20]. The numerical results indicate that the branch-and-bound algorithm is particularly effective when the global minimum of the MPEC has a known optimal value. The use of disjunctive cuts can, in certain cases, help the search for a global minimum for more difficult cases of MPEC where the optimal function value is not known.

The remainder of this paper is organized as follows. In Section 2, a branch-and-bound algorithm (BBASET) for finding a global minimum for MPEC is discussed. The computation of lower bounds is discussed in Section 3, while Section 4 addresses the computation of upper bounds. Finally, a report of the computational experience and some conclusions are included in the last section.

2. Branch-and-Bound Algorithm

In this section, we describe a branch-and-bound algorithm for finding a global minimum to the MPEC (1). This algorithm partially explores a binary tree that is generated according to the dichotomy $z_i = 0$ or $w_i = 0$, $\forall i$, presented in any complementary solution. Thus, if k is a given node of the branch-and-bound enumeration tree at any stage of this process, then two children nodes of the form depicted in Figure 1 are generated.

Based on this partitioning process and by examining the chain from any node k to the root, we define the following sets:

$$\begin{aligned} J_k &= \{i : \; w_i \text{ fixed at zero for node } k\} \\ L_k &= \{i : \; z_i \text{ fixed at zero for node } k\}. \end{aligned} \quad (2)$$

The corresponding subproblem at node k is as follows

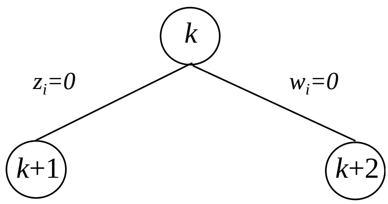

Figure 1. Binary tree.

$$\begin{array}{llll} \text{MPEC}_{JL}: & \text{Minimize} & f(z, y), & \\ & \text{subject to} & Ew = Mz + Ny + q, & \\ & & z \geqslant 0, \quad w \geqslant 0 & \\ & & y \in K_y = \{y \in \mathbb{R}^m: \ Ay = b, y \geqslant 0\}, & \\ & & z^T w = 0, & \\ & & w_i = 0, & i \in J, \\ & & z_i = 0 & i \in L, \end{array} \tag{3}$$

where $J = J_k$ and $L = L_k$ are the sets defined by (2).

Within the framework of a branch-and-bound algorithm, a lower bound LB(k) for the node k subproblem can be found by solving the convex relaxation (linear if f is a linear function), obtained upon omitting the complementarity constraint $z^T w = 0$ in the $\text{MPEC}_{J_k L_k}$. As we show later, disjunctive cuts [1, 23] can also be gainfully employed for this purpose.

The Complementarity Active-Set Algorithm (Caset) [14] can be used for finding upper bounds on the optimal value of MPEC. This algorithm uses the active-set strategy [18] along with a solution to the GLCP to reduce the objective value, always maintains complementarity, and finds a stationary point for MPEC. Indeed, if $(\bar{z}, \bar{y})$ is a stationary point for $\text{MPEC}_{J_k L_k}$ associated with node k, then $\text{UB} = f(\bar{z}, \bar{y})$ yields the desired upper bound.

The Caset algorithm is also used in the branching process employed by the proposed algorithm for generating the tree. Let us assume that the nodes of the tree generated by the algorithm constitute the elements of the list $\mathcal{L}$, and let $|\mathcal{L}|$ be its cardinality. Suppose further that at node k, the Caset algorithm finds a stationary point $(\bar{z}, \bar{w}, \bar{y})$ for $\text{MPEC}_{J_k L_k}$. If λ_i^w and λ_i^z are the Lagrange multipliers associated with the respective constraints $w_i \geqslant 0$ and $z_i \geqslant 0$, that are obtained for this solution, then there are two possible cases:

1. If

$$\lambda_i^w \geqslant 0, \quad \forall i \notin J_k \quad \text{and} \quad \lambda_i^z \geqslant 0, \quad \forall i \notin L_k,$$

then $(\bar{z}, \bar{w}, \bar{y})$ is a global minimum of $\text{MPEC}_{J_k L_k}$.

2. If

$$\lambda_i^w < 0 \quad \text{for some } i \notin J_k, \quad \text{or} \quad \lambda_i^z < 0 \quad \text{for some } i \notin L_k$$

then let $\lambda_{i_1}^w < 0, \ldots, \lambda_{i_r}^w < 0$ and $\lambda_{j_1}^z < 0, \ldots, \lambda_{j_s}^z < 0$, with $r + s \geqslant 1$. In this case, based on these multipliers, multiple simultaneous partitions of the binary tree are generated. Figure 2 depicts the consequent partitioning of node k that is constructed for the case where $r = s = 2$.

In general this branching of the tree creates $(r + s + 1)$ new active nodes and $(\bar{z}, \bar{w}, \bar{y})$ is a global minimum for MPEC_{JL} associated with the

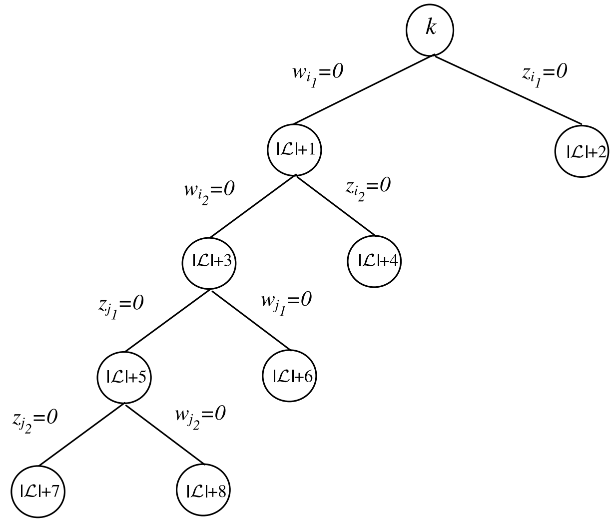

Figure 2. Branching scheme.

node $|\mathcal{L}|+2(r+s)-1$. Therefore this node can be fathomed and so, just $(r+s)$ active nodes need to be added to the list. It is also important to notice that the stationary point $(\bar{z}, \bar{w}, \bar{y})$ can be used as an initial solution for each MPEC_{JL} associated with these $(r+s)$ new nodes that have been introduced into the list $\mathcal{L}$. In this way, the real number $\text{Val}(t)=f(\bar{z}, \bar{y})$ can be associated with each one of these nodes t.

The steps of the branch-and-bound algorithm are formally stated below.

Branch-and-Bound Algorithm

Step 0. Let $\mathcal{L}=\{1\}$ be the initial list of active nodes, $\text{Ub}=\infty$ and $\text{Val}(1)=\infty$.

Step 1. *Optimality* – If $\mathcal{L}=\emptyset$, stop the algorithm. If $\text{Ub}=\infty$, then MPEC has no optimal solution (GLCP has no solution or MPEC is unbounded). Otherwise, the solution $(\bar{z}, \bar{w}, \bar{y})$ associated with $\text{Ub}<\infty$ is a global minimum of MPEC and the optimal value is Ub.

Step 2. *Node Selection* – If $\mathcal{L}\neq\emptyset$, choose the node $k\in\mathcal{L}$ having the least value of $\text{Val}(k)$, breaking ties by selecting the least-indexed (highest level) node. Let J_k and L_k be the sets of variables w_i and z_i fixed at zero and let $x^k=(z^k, w^k, y^k)$ be the initial solution associated with this node having the value $\text{Val}(k)$. Replace $\mathcal{L}\leftarrow\mathcal{L}\setminus\{k\}$.

Step 3. *Computing a Lower Bound* – Find a lower bound $\text{Lb}(k)$ at node k, using x^k as an initial solution. If $\text{Lb}(k)\geqslant\text{Ub}$ return to Step 1.

Step 4. *Computing an Upper Bound* – Let $\text{MPEC}_{J_k L_k}$ be defined by (3), with $J = J_k$ and $L = L_k$, and let x^k be the initial solution for this node k. Starting with x^k, find a feasible solution for $\text{MPEC}_{J_k L_k}$ (that is, a solution of the associated GLCP). If a solution does not exist, return to Step 1. Otherwise, find a stationary point $x^k = (z^k, w^k, y^k)$ of $\text{MPEC}_{J_k L_k}$ using the CASET algorithm and set

$$\text{UB} = \min\left\{\text{UB}, f(z^k, y^k)\right\}.$$

Update the incumbent solution (z^k, w^k, y^k) if a decrease occurs in the value of UB.

Step 5. Let λ_i^w, and λ_j^z be the Lagrange multipliers associated with the constraints $w_i \geqslant 0$ and $z_j \geqslant 0$, respectively, at the stationary point $x^k = (z^k, w^k, y^k)$ calculated in Step 4. If $\lambda_i^w \geqslant 0$ for all $i \notin J_k$ and $\lambda_j^z \geqslant 0$ for all $j \notin L_k$, return to Step 1. Otherwise, let $i = i_1, \ldots, i_r \notin J_k$ and $j = j_1, \ldots, j_s \notin L_k$ be the respective indices corresponding to $\lambda_i^w < 0$ and $\lambda_j^z < 0$, with $r + s \geqslant 1$. Add $(r + s)$ nodes to the list $\mathcal{L}$ according to the branching process previously described (see Figure 2). For each of these nodes, let $x^k = (z^k, w^k, y^k)$ be the initial solution and set $\text{VAL}(k) = f(z^k, y^k)$. Return to Step 2.

It follows from the description of this algorithm that at each node, a GLCP corresponding to the constraints of the MPEC_{JL} has to be solved. This topic is discussed in Section 4. Furthermore, the computation of the lower bound is based on the solution of the relaxed convex program, which is obtained from MPEC_{JL} by omitting the complementarity constraint $z^T w = 0$ and augmenting this problem with suitable disjunctive cuts, to be described in Section 3.

The effectiveness of the proposed branch-and-bound algorithm naturally depends on computing good quality lower and upper bounds. Some simple rules can be also incorporated in the algorithm to control the size of the list $\mathcal{L}$.

- *Order of variable indices in the branching process* (Step 5)
  In the implementation of the branching process of Figure 2, the pairs of complementary variables (w_i, z_i) are selected for branching in increasing order of the (negative) values of the Lagrange multipliers λ_i^w and λ_j^z.
- *Node selection* (Step 2)
  Select the node k having the minimum value of $\text{VAL}(k)$, where $\text{VAL}(k)$ is the objective function value at the initial solution associated with node k. Break tries by choosing the least-indexed node, i.e., the node at the highest level in the tree.

3. Computation of Lower Bounds

As stated in the previous section, the computation of the lower bound at each node is based on solving the relaxed convex program, RMPEC_{JL}, obtained from the corresponding MPEC_{JL} by omitting the complementarity constraint $z^T w = 0$. However, computational experience with MPECs [11, 12] has shown that solving this simple relaxed program of each node does not in general lead to good lower bounds. Therefore, additional disjunctive cuts (see [23], for example) may be useful for improving the derived lower bounds.

Let us consider the optimal solution $(\bar{z}, \bar{w}, \bar{y})$ of RMPEC_{JL} associated with a particular node of the tree. If this solution is complementary, then it provides a new upper bound $\text{UB} = f(\bar{z}, \bar{y})$ and the node can be fathomed. Otherwise, there exists at least one pair of positive complementary variables. To explain the disjunctive cut generation process for this solution $(\bar{z}, \bar{w}, \bar{y})$, consider, without loss of generality, the constraints of the relaxed program RMPEC_{JL} in the following form:

$$\begin{aligned}
&Ew = Mz + Ny + q,\\
&Ay = b,\\
&w \geqslant 0, \;\; z \geqslant 0, \; y \geqslant 0,\\
&w_i = 0, \quad i \in J,\\
&z_i = 0, \quad i \in L.
\end{aligned}$$

Then these constraints can be written in the form (where u_j can possibly be infinity)

$$\begin{aligned}
&Cx = g,\\
&0 \leqslant x_j \leqslant u_j, \quad \forall j
\end{aligned}$$

with

$$g = \begin{bmatrix} q \\ b \end{bmatrix}, \quad C = \begin{bmatrix} -M & E & -N \\ 0 & 0 & A \end{bmatrix}, \quad x = \begin{bmatrix} z \\ w \\ y \end{bmatrix}.$$

The case where the optimal solution of the relaxed program is basic is considered first, and the case in which the optimal solution does not verify this property is treated thereafter. Throughout, we assume that the final solution obtained by solving the convex relaxation is presented in a simplex tableau format with the partitioning of the variables into basic, nonbasic, and superbasic variables, as in [17].

3.1. DISJUNCTIVE CUTS FOR BASIC SOLUTIONS

Let us assume that $\bar{x}=(\bar{z},\bar{w},\bar{y})$ is a noncomplementary basic solution. Then there must exist a pair of positive complementary variables (z_k, w_k) that are both basic. Let r and s be the basic rows corresponding to these variables, and let (L, U, J) be the partition associated with this basic solution, where L and U are the indices of the nonbasic variables that are presently at their lower and upper bounds, respectively, and J is the index set of the basic variables. Then, we can write

$$\begin{aligned} w_k &= \bar{g}_s - \sum_{j\in L}\bar{c}_{sj}x_j - \sum_{j\in U}\bar{c}_{sj}x_j, \\ z_k &= \bar{g}_r - \sum_{j\in L}\bar{c}_{rj}x_j - \sum_{j\in U}\bar{c}_{rj}x_j, \end{aligned}$$

where $\bar{g}_i$ and $\bar{c}_{ij}$ are the elements of the simplex tableau corresponding to the basic solution. Imposing the disjunction that $w_k \leqslant 0$ or $z_k \leqslant 0$, we get

$$\begin{aligned} &\sum_{j\in L}\bar{c}_{sj}x_j + \sum_{j\in U}\bar{c}_{sj}x_j \geqslant \bar{g}_s \\ &\vee \\ &\sum_{j\in L}\bar{c}_{rj}x_j + \sum_{j\in U}\bar{c}_{rj}x_j \geqslant \bar{g}_r. \end{aligned} \tag{4}$$

Since $u_j < \infty$, $\forall j \in U$, then we can write (4)

$$\begin{aligned} &\sum_{j\in L}\bar{c}_{sj}x_j - \sum_{j\in U}\bar{c}_{sj}(u_j - x_j) \geqslant \tilde{g}_s \\ &\vee \\ &\sum_{j\in L}\bar{c}_{rj}x_j - \sum_{j\in U}\bar{c}_{rj}(u_j - x_j) \geqslant \tilde{g}_r, \end{aligned}$$

where

$$\tilde{g}_s = \bar{g}_s - \sum_{j\in U}\bar{c}_{sj}u_j > 0, \quad \text{and} \quad \tilde{g}_r = \bar{g}_r - \sum_{j\in U}\bar{c}_{rj}u_j > 0.$$

As $\tilde{g}_s$ and $\tilde{g}_r$, are the current positive values of w_k and z_k, respectively, then we can divide the two expressions by $\tilde{g}_s$ and $\tilde{g}_r$, respectively, to obtain

$$\begin{aligned} &\sum_{j\in L}\frac{\bar{c}_{sj}}{\tilde{g}_s}x_j - \sum_{j\in U}\frac{\bar{c}_{sj}}{\tilde{g}_s}(u_j - x_j) \geqslant 1 \\ &\vee \\ &\sum_{j\in L}\frac{\bar{c}_{rj}}{\tilde{g}_r}x_j - \sum_{j\in U}\frac{\bar{c}_{rj}}{\tilde{g}_r}(u_j - x_j) \geqslant 1. \end{aligned}$$

But

$$x_j \geqslant 0 \quad \text{and} \quad u_j - x_j \geqslant 0 \quad \text{for all } j \in L \cup U.$$

If we now choose

$$d_j = \begin{cases} \max\left\{\dfrac{\bar{c}_{sj}}{\tilde{g}_s}, \dfrac{\bar{c}_{rj}}{\tilde{g}_r}\right\}, & \forall j \in L, \\ \min\left\{\dfrac{\bar{c}_{sj}}{\tilde{g}_s}, \dfrac{\bar{c}_{rj}}{\tilde{g}_r}\right\}, & \forall j \in U, \end{cases}$$

then the following disjunctive cut is obtained:

$$\sum_{j \in L} d_j x_j + \sum_{j \in U} (-d_j)(u_j - x_j) \geqslant 1$$

that is

$$\sum_{j \in L \cup U} d_j x_j \geqslant \lambda,$$

where $\lambda = 1 + \sum_{j \in U} d_j u_j$.

3.2. DISJUNCTIVE CUTS FOR SOLUTIONS THAT ARE NOT BASIC

Analogous to the previous case, let us assume that there exists at least one pair of positive complementary variables. As the solution is not basic, one of the following three scenarios can occur:

- Both are basic variables.
- One is a basic variable and the other is a superbasic variable.
- Both are superbasic variables.

Let (z_k, w_k) be a pair of positive complementary variables. Let us first assume that these two variables are basic in the rows r and s of the associated optimal tableau. Thus, we have

$$w_k = \bar{g}_s - \sum_{j \in L \cup U} \bar{c}_{sj} x_j - \sum_{j \in S} \bar{c}_{sj} x_j \quad \text{and} \quad z_k = \bar{g}_r - \sum_{j \in L \cup U} \bar{c}_{rj} x_j - \sum_{j \in S} \bar{c}_{rj} x_j,$$

where S, L, and U are the index sets of the superbasic variables, and the nonbasic variables at their lower and upper bounds, respectively. The disjunction

$$w_k \leqslant 0 \qquad \vee \qquad z_k \leqslant 0$$

can thus be restated as

$$\sum_{j\in L\cup U}\bar{c}_{sj}x_j+\sum_{j\in S}\bar{c}_{sj}x_j\geqslant\bar{g}_r \qquad \vee \qquad \sum_{j\in L\cup U}\bar{c}_{rj}x_j+\sum_{j\in S}\bar{c}_{rj}x_j\geqslant\bar{g}_s.$$

If $\bar{x}_j$ for $j\in S$ are the values of the superbasic variables, then we can rewrite this disjunction as:

$$\sum_{j\in L}\bar{c}_{sj}x_j-\sum_{j\in U}\bar{c}_{sj}(u_j-x_j)+\sum_{j\in S}\bar{c}_{sj}(x_j-\bar{x}_j)\geqslant\tilde{g}_s \tag{5}$$

$$\vee$$

$$\sum_{j\in L}\bar{c}_{rj}x_j-\sum_{j\in U}\bar{c}_{rj}(u_j-x_j)+\sum_{j\in S}\bar{c}_{rj}(x_j-\bar{x}_j)\geqslant\tilde{g}_r, \tag{6}$$

where

$$\tilde{g}_s=\bar{g}_s-\sum_{j\in U}\bar{c}_{sj}u_j-\sum_{j\in S}\bar{c}_{sj}\bar{x}_j>0, \quad \text{and} \quad \tilde{g}_r=\bar{g}_r-\sum_{j\in U}\bar{c}_{rj}u_j-\sum_{j\in S}\bar{c}_{rj}\bar{x}_j>0.$$

We again note that $\tilde{g}_s$ and $\tilde{g}_r$, are the current positive values of w_k and z_k, respectively. If $S=\emptyset$, then the disjunctive cut can be generated as in the case for basic solutions. Otherwise, we shift indices from S to L or U, as described below. To do this, we replace each term associated with $j\in S$ in (5) and (6) by one of the following expressions:

$$\bar{c}_{ij}(x_j-\bar{x}_j)=\bar{c}_{ij}x_j-\bar{c}_{ij}\bar{x}_j, \tag{7}$$

$$\bar{c}_{ij}(x_j-\bar{x}_j)=-\bar{c}_{ij}(u_j-x_j)-\bar{c}_{ij}(\bar{x}_j-u_j) \tag{8}$$

with $i=s,r$, where (8) applies only if $u_j<\infty$. If we use the expression (7) we have,

$$\begin{aligned}&S\leftarrow S\setminus\{j\}, \qquad L\leftarrow L\cup\{j\},\\ &\tilde{g}_s\leftarrow g_s+\bar{c}_{sj}\bar{x}_j,\\ &\tilde{g}_r\leftarrow g_r+\bar{c}_{rj}\bar{x}_j.\end{aligned} \tag{9}$$

On the other hand, (8) leads to the following transformations (where $u_j<\infty$):

$$\begin{aligned}&S\leftarrow S\setminus\{j\}, \qquad U\leftarrow U\cup\{j\},\\ &\tilde{g}_s\leftarrow\tilde{g}_s+\bar{c}_{sj}(\bar{x}_j-u_j),\\ &\tilde{g}_r\leftarrow\tilde{g}_r+\bar{c}_{rj}(\bar{x}_j-u_j).\end{aligned} \tag{10}$$

As before for the generation of the disjunctive cuts, $\tilde{g}_s$ and $\tilde{g}_r$ must be both positive after these transformations. Hence, the options (9) and (10) must be chosen in such a way that this condition holds, if possible. Three cases are possible and are discussed below.

Case 1. $\bar{c}_{sj} \geqslant 0$ and $\bar{c}_{rj} \geqslant 0$.
We update S, L, $\tilde{g}_s$, and $\tilde{g}_r$ through (9).

Case 2. $\bar{c}_{sj} \leqslant 0$ and $\bar{c}_{rj} \leqslant 0$.
If $u_j < \infty$, update S, U, $\tilde{g}_s$, and $\tilde{g}_r$ according to (10). Otherwise ($u_j = \infty$), update S, L, $\tilde{g}_s$, and $\tilde{g}_r$ by using (9).

Case 3. $\bar{c}_{sj} > 0$ and $\bar{c}_{rj} < 0$ (or symmetrically, vice versa).
In this case, determine

$$\alpha = \min\left\{\bar{c}_{sj}(u_j - \bar{x}_j), \bar{c}_{rj}(l_j - \bar{x}_j)\right\}$$

and consider the following two possible cases:

1. If $\alpha = \bar{c}_{sj}(u_j - \bar{x}_j)$, update S, U, $\tilde{g}_s$, and $\tilde{g}_r$ according to (10);
2. If $\alpha = \bar{c}_{rj}(l_j - \bar{x}_j)$, update S, L, $\tilde{g}_s$, and $\tilde{g}_r$ by using (9).

It is important to point out that in cases 1 and 2 with finite bounds, the updated values of $\tilde{g}_s$ and $\tilde{g}_r$ are nondecreasing. However, if the other cases occur, we are no longer guaranteed that $\tilde{g}_s$ and $\tilde{g}_r$ remain positive. Thus, at the end of the process we examine the resulting disjunction in the form:

$$\begin{aligned}
&\sum_{j\in L} \bar{c}_{sj} x_j - \sum_{j\in U} \bar{c}_{sj}(u_j - x_j) \geqslant \tilde{g}_s \\
&\vee \\
&\sum_{j\in L} \bar{c}_{rj} x_j - \sum_{j\in U} \bar{c}_{rj}(u_j - x_j) \geqslant \tilde{g}_r.
\end{aligned}$$

If $\tilde{g}_s > 0$ and $\tilde{g}_r > 0$, then the disjunctive cut can be determined as in case (I). Otherwise, the disjunctive cut is not generated.

In the case where the solution is not basic and at least one of the complementary variables z_k and w_k of the pair that violates complementarity is superbasic, then a pivot operation exchanging this variable with a basic variable should be performed. If it is possible by pivoting to obtain a representation of the current solution in which both the complementary variables z_k and w_k are basic, then the foregoing process for the construction of the disjunctive cut can be applied. Otherwise, the disjunctive cut is not constructed for this pair.

3.3. EXAMPLE OF A DISJUNCTIVE CUT CONSTRUCTION

Consider an MPEC with the following set of constraints (GLCP):

$$\begin{aligned}
&2x_1 + 3x_2 + x_3 = 6, \\
&-x_1 + x_2 + x_4 = 1, \\
&x_i \geqslant 0, \quad i = 1, 2, 3, 4, \\
&x_3 x_4 = 0
\end{aligned}$$

and let $\bar{x} = (2, 0, 2, 3)$ be a noncomplementary solution that is not basic, where the basic variables are x_3 and x_4, the nonbasic variable x_2 is at its lower bound, and the variable x_1 is superbasic. Since there exists only one superbasic variable, we have:

$$\begin{cases} x_3 = 2 - 2(x_1 - 2) - 3x_2, \\ x_4 = 3 - (-1)(x_1 - 2) - x_2. \end{cases}$$

Then

$$\begin{cases} x_3 \leqslant 0 \;\Leftrightarrow\; 2(x_1 - 2) + 3x_2 \geqslant 2, \\ x_4 \leqslant 0 \;\Leftrightarrow\; (-1)(x_1 - 2) + x_2 \geqslant 3 \end{cases}$$

and we are in Case 3. As $u_1 = \infty$, we do:

$$S \leftarrow S \setminus \{1\}, \quad L \leftarrow L \cup \{1\}$$

and update $\tilde{g}_s$ and $\tilde{g}_r$ from (7). This yields

$$\tilde{g}_s = 2 + 2 \times 2 = 6,$$
$$\tilde{g}_r = 3 + (-1) \times 2 = 1.$$

Since $\tilde{g}_s > 0$, $\tilde{g}_r > 0$, then the disjunctive cut can be generated according to

$$\left.\begin{array}{l} 2x_1 + 3x_2 \geqslant 6 \\ \text{or} \\ -x_1 + x_2 \geqslant 1 \end{array}\right\rangle \Rightarrow \left.\begin{array}{l} \frac{1}{3}x_1 + \frac{1}{2}x_2 \geqslant 1 \\ \text{or} \\ -x_1 + x_2 \geqslant 1 \end{array}\right\rangle \Rightarrow \frac{1}{3}x_1 + x_2 \geqslant 1.$$

As depicted in Figure 3, this cut deletes the current solution and reduces the feasible region without removing any complementary solutions.

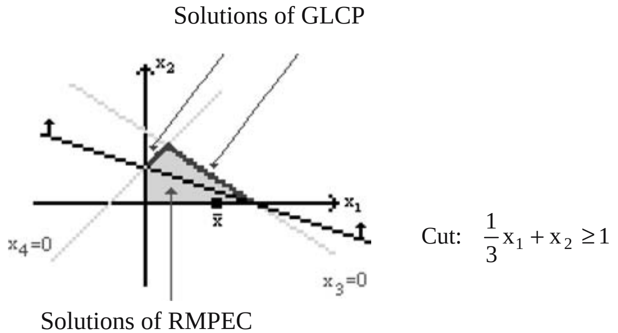

Figure 3. Example of a disjunctive cut application.

3.4. APPLICATION OF DISJUNCTIVE CUTS TO COMPUTE LOWER BOUNDS

The implementation of the disjunctive cuts is done in Step 3 of the branch-and-bound algorithm. In this step, at node t, say, a lower bound LB(t) for the MPEC is first calculated by solving the relaxed problem RMPEC$_{J_tL_t}$ associated with MPEC$_{J_tL_t}$ in which the complementary constraint is relaxed. If LB(t) $\geqslant$UB, then the algorithm returns to Step 1. If the optimal solution is complementary, then this solution provides a new upper bound UB, where UB$=$LB(t), and so the algorithm again returns to Step 1. Otherwise, the optimal solution of RMPEC$_{J_tL_t}$ is a noncomplementary solution. For each pair (z_k, w_k) of positive complementary variables, a corresponding disjunctive cut is generated, according to the previously discussed process, if possible. The relaxed program with the addition of these cuts is then resolved, a new lower bound LB(t) is computed, and the process is repeated. In practice, for each noncomplementary solution, a number of disjunctive cuts equal to the number of pairs of positive complementary variables is added. However, the generation of too many cuts should be avoided and imposing a limit on the number of cuts to be added is advisable.

4. Computing Upper Bounds

As stated in Section 2, computation of upper bounds is achieved using the Complementarity Active-Set Algorithm (CASET) [14] for finding a stationary point of the MPEC$_{JL}$ associated with each node, as given by (3), where $J \cap L = \emptyset$ and $J \cup L \subseteq \{1, \dots, n\}$. This algorithm needs a solution of the GLCP$_{JL}$, which corresponds to the set of constraints of MPEC$_{JL}$. In [14], it is shown that if $J = L = \emptyset$, E is the identity matrix and M is a PSD matrix, then a solution of the corresponding GLCP can be obtained from a stationary point of the function $z^T w$ subject to the set of the linear constraints of this GLCP. In this section we first investigate if this result remains valid or not for the GLCP$_{JL}$. As E is the identity matrix, we consider the following quadratic program:

$$
\begin{aligned}
QP_{GLCP_{JL}}: \text{Minimize} \quad & z^T(q + Mz + Ny), \\
\text{subject to } (Mz + Ny + q)_i &\geqslant 0, \quad i \notin J, \\
Ay - b &= 0, \\
(Mz + Ny + q)_i &= 0, \quad i \in J, \\
z_i &\geqslant 0, \quad i \notin L, \\
z_i &= 0, \quad i \in L, \\
y &\geqslant 0.
\end{aligned}
$$

A stationary point for this program must verify the following conditions, in addition to feasibility in QP$_{GLCP_{JL}}$.

$$q+(M+M^T)z+Ny=M^T\alpha+\beta, \tag{11}$$
$$N^Tz=N^T\alpha+A^T\mu+\gamma, \tag{12}$$

$$\begin{array}{ll}
\alpha_i\geqslant 0,\quad \alpha_i(Mz+Ny+q)_i=0, & \forall i\notin J,\\
\alpha_i \text{ unrestricted}, & \forall i\in J,\\
\beta_i\geqslant 0,\quad \beta_i z_i=0 & \forall i\notin L,\\
\beta_i \text{ unrestricted}, & \forall i\in L,\\
\mu \text{ unrestricted}, & \\
\gamma\geqslant 0,\quad \gamma^T y=0. &
\end{array}$$

Then, the following result holds.

THEOREM 1. Let $M\in PSD$.

If $(z,y,\alpha,\mu,\beta,\gamma)$ *is a stationary point for the quadratic program* $QP_{GLCP_{JL}}$ *and* $\alpha^T\beta\geqslant 0$, *then* (z,w,y) *is a solution of* $GLCP_{JL}$.

Proof. To show that (z,w,y) is a solution of the GLCP, we have to prove that

$$z^T(q+Mz+Ny)=0.$$

Multiplying the constraints (11) by $(\alpha-z)^T$, we get

$$(\alpha-z)^T(q+Mz+Ny)=(\alpha-z)^TM^T(\alpha-z)+(\alpha-z)^T\beta. \tag{13}$$

As $z^T\beta=0$, then

$$(\alpha-z)^T\beta=\alpha^T\beta.$$

Similarly, by primal feasibility and complementary slackness, we have

$$\alpha^T(q+Mz+Ny)=\sum_{i\in J}\alpha_i(q+Mz+Ny)_i+\sum_{i\notin J}\alpha_i(q+Mz+Ny)_i=0.$$

Hence, (13) becomes

$$-z^T(q+Mz+Ny)=(\alpha-z)^TM^T(\alpha-z)+\alpha^T\beta. \tag{14}$$

Since by assumption $\alpha^T\beta\geqslant 0$ and $M\in PSD$, then $(\alpha-z)^TM^T(\alpha-z)\geqslant 0$, and so by primal feasibility, this implies that

$$z^T(q+Mz+Ny)=0.$$

Thus, (z,y) is a solution of $GLCP_{JL}$. □

As a consequence of this theorem the following result can be established.

THEOREM 2. *Let* $(z, y, \alpha, \mu, \beta, \gamma)$ *be a stationary point for* $QP_{GLCP_{JL}}$.

1. *If* $M \in PD$, *then*

$$\alpha^T \beta \geqslant 0 \Rightarrow z^T w = 0, \quad \alpha = z, \quad \beta = w \quad \textit{and} \quad \alpha^T \beta = 0.$$

2. *If* M *is skew symmetric* $(M = -M^T)$, *then*

$$\alpha^T \beta \geqslant 0 \Leftrightarrow z^T w = 0.$$

Proof.

1. If $\alpha^T \beta \geqslant 0$, then $z^T w = 0$, by the proof of Theorem 1. By (14) this implies that $\alpha^T \beta = 0$ and $(\alpha - z)^T M^T (\alpha - z) = 0$. Hence $\alpha = z$, since $M \in PD$. Therefore by (11), we also have $\beta = q + Mz + Ny = w$.
2. If M is skew symmetric, then $(\alpha - z)^T M^T (\alpha - z) = 0$. Therefore $z^T w = -\alpha^T \beta$. □

We now provide some sufficient conditions for $\alpha^t \beta \geqslant 0$ to hold. These conditions guarantee that a stationary point of $\text{QP}_{GLCP_{JL}}$ is a solution of GLCP_{JL}.

THEOREM 3. *Let* $M \in PSD$.

If $(z, y, \alpha, \mu, \beta, \gamma)$ *is a stationary point for the quadratic program* $QP_{GLCP_{JL}}$ *and if one of the following conditions holds true:*

1. $J \cup L = \emptyset$
2. $z_i + w_i > 0, \quad \forall i \in J \cup L$
3. $\sum_{i \in J \cup L} \alpha_i \beta_i \geqslant 0$,

then $\alpha^T \beta \geqslant 0$ *and* (z, w, y) *is solution of* $GLCP_{JL}$.

Proof. Because $\alpha_i \geqslant 0$ and $\beta_i \geqslant 0$, $\forall i \notin J \cup L$, we have,

$$\alpha^T \beta = \sum_{i \in J \cup L} \alpha_i \beta_i + \sum_{i \notin J \cup L} \alpha_i \beta_i \geqslant \sum_{i \in J \cup L} \alpha_i \beta_i. \tag{15}$$

Hence, the following hold true:

1. If $J \cup L = \emptyset$, then (15) directly yields $\alpha^T \beta \geqslant 0$.
2. If $z_i + w_i > 0, \quad \forall i \in J \cup L$, then

$$\begin{cases} i \in J \Rightarrow z_i > 0 \Rightarrow \beta_i = 0 \\ i \in L \Rightarrow w_i > 0 \Rightarrow \alpha_i = 0, \end{cases}$$

and so, $\sum_{i \in J \cup L} \alpha_i \beta_i = 0$. Hence $\alpha^T \beta \geqslant 0$, by (15).

3. More directly, if $\sum_{i\in J\cup L} \alpha_i\beta_i \geqslant 0$, then (15) yields $\alpha^T\beta \geqslant 0$.

The result now follows, noting Theorem 1. □

It follows from this theorem that a stationary point of the quadratic program $QP_{GLCP_{JL}}$ satisfying the strictly complementarity property ($z_i + w_i > 0$ for all i) provides a solution of $GLCP_{JL}$.

The next example shows that it is possible to find a point stationary for $QP_{GLCP_{JL}}$ that is not a solution of $GLCP_{JL}$. Let us consider the $GLCP_{JL}$

$$\begin{aligned}
&\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} -2 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 3 \end{bmatrix} y,\\
&z_1 = 0,\\
&w_1 \geqslant 0,\ w_2 \geqslant 0,\ z_2 \geqslant 0,\ y \geqslant 0,\\
&w_2 z_2 = 0.
\end{aligned}$$

Note that $M \equiv \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \in PD$, $J = \emptyset$, and $L = \{1\}$. Moreover, by also setting $z_2 = 0$, we see that this GLCP has infinite solutions of the following form:

$$(w_1, w_2, z_1, z_2, y) = (-2 + y, 2 + 3y, 0, 0, y), \quad \text{with } y \geqslant 2.$$

A stationary point $(w_1, w_2, z_1, z_2, y, \alpha_1, \alpha_2, \mu, \beta_1, \beta_2)$ for the quadratic program associated with this GLCP must verify the following conditions

$$\begin{aligned}
&\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} -2 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 3 \end{bmatrix} y\\
&z_1 = 0\\
&\begin{bmatrix} -2 \\ 2 \end{bmatrix} + \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 3 \end{bmatrix} y = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} + \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix}\\
&[1\ 3]\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = [1\ 3]\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} + \gamma\\
&\alpha_1 \geqslant 0,\ \alpha_1 w_1 = 0,\\
&\alpha_2 \geqslant 0,\ \alpha_2 w_2 = 0,\\
&\beta_2 \geqslant 0,\ \beta_2 z_2 = 0,\\
&\gamma \geqslant 0,\ \gamma y = 0,\\
&w_1 \geqslant 0,\ w_2 \geq 0,\ z_2 \geqslant 0,\ y \geqslant 0,\\
&\beta_1 \text{ unrestricted.}
\end{aligned}$$

To show that there exists a stationary point for $QP_{GLCP_{JL}}$ that is not a solution to $GLCP_{JL}$, we must exhibit a solution to the above system for which $z_2 > 0$ and $w_2 > 0$. It is readily verified that the solution

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 4 \end{bmatrix}, \quad \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \quad y = 0, \quad \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \end{bmatrix}, \text{ and } \gamma = 0$$

provides such a stationary point. Note that $\alpha^T \beta = -48 \not\geq 0$ and $z_1 + w_1 = 0$. Since a solution exists to $GLCP_{JL}$ and the constraints of $QP_{GLCP_{JL}}$ are linear, there exists a stationary point of $QP_{GLCP_{JL}}$ that provides a solution to the underlying GLCP. Indeed, such a solution is given by $\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 8 \end{bmatrix}$, $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $y = 2$, $\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 8 \end{bmatrix}$, and $\gamma = 0$. Note that $\alpha^T \beta = 0$ in this case.

Computational experience with MPECs to be reported in next section has shown that, in general, a solution to $GLCP_{JL}$ is obtained from a stationary point of the associated $QP_{GLCP_{JL}}$. In case this does not happen, then $GLCP_{JL}$ has to be solved by the enumerative method described in [13]. It should be noted that the latter algorithm starts by finding a stationary point of $GLCP_{JL}$ at the initial node of the binary tree that is generated through this process.

In order to avoid the use of this latter algorithm in this case, it is more advisable to keep open the corresponding node until the end of search and to expect this node to be fathomed in a later stage of the branch-and-bound algorithm by showing that the lower-bound at this node is greater than or equal to the current upper-bound. Since we use least lower bound node selection rule, this would mean breaking ties, if any, in favor of nodes other than the one pertaining to such a case.

It is also important to note that some structured MPECs the $GLCP_{JL}$ can be solved by exploiting the special structure of the problems. This is the case of the MPECs associated with the so-called linear and quadratic HFP [20], to be discussed later in this paper.

5. Computational Experience

In this section we report some computational experience with the new branch-and-bound algorithm for solving BP, HFP, and NP-hard linear complementarity problems (LCP) associated with knapsack problems. All runs have been performed on a Pentium IV 2.4 GHz machine having 256 MB of RAM.

5.1. TEST PROBLEMS

The generator for linear and quadratic bilevel problems (QBP) used in our computations has been developed by Calamai and Vicente [4, 5], and allows the construction of problems having a known optimal solution. The BP generated by this technique has the following form:

$$\begin{array}{lll} \text{Minimize}_{x,y} & Q(x,y) & \\ \text{subject to} & x \geqslant 0 \quad \text{and} & y \text{ is an optimal solution of:} \\ & \text{Minimize}_y & q(x,y), \\ & \text{subject to} & A_1 y + A_2 x \leqslant b, \end{array}$$

where Q and q are twice continuously differentiable functions in an open set that contains the feasible region of BP, $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A_1 \in \mathbb{R}^{r \times m}$, $A_2 \in \mathbb{R}^{r \times n}$, and $b \in \mathbb{R}^r$. Table 1 presents the characteristics of the linear bilevel problems (LBP) obtained by this generator [14], and Table 2 displays the specifications for five QBP that have been generated [5]. In these problems we additionally assume that the feasible region is restricted to the nonnegative orthant, that is, $x \geqslant 0$ and $y \geqslant 0$. This modification does not alter the set of local and global minima for these test problems. In Tables 1 and 2, as well as in the remaining tables of this section, OP.OBJ. is the optimal value of the objective function, while NG and ML are the total number of global and local minima of the problem, respectively.

Using the generator in [4], it is also possible to modify the problems presented in Table 1 to obtain new LBPs that have the same characteristics

Table 1. Characteristics of LBP problems

PROB	n	m	OP.OBJ.	NG	NL
LBP1	4	2	6.000	1	3
LBP2	10	5	16.000	2	14
LBP3	15	10	30.200	1	511
LBP4	20	7	6.000	1	3
LBP5	20	7	14.000	4	12
LBP6	30	20	7.500	1	7
LBP7	30	20	23.500	8	56
LBP8	30	50	50.500	4	1020

Table 2. Characteristics of QBP problems

PROB	n	m	OP.OBJ.	NG	NL
QBP1	4	2	0.313	1	3
QBP2	6	4	0.593	1	15
QBP3	10	5	0.790	2	14
QBP4	25	15	1.040	4	28
QBP5	30	20	2.436	32	8160

Table 3. Characteristics of modified LBP problems

PROB	vx	vy	D
LBP1M	(−0.7,0.7,0.1,−0.1)	(0.8,−0.6)	(20,10,20,10,10,20)
LBP2M	(−0.8,0,0,0.5,0, −0.3,0,0,−0.1,0.1)	(0.6,0,0.7, 0.4,−0.3)	(50,30,80,60,24,64,78,82, 38,60,90,28,85,70,38)
LBP3M	(−0.8,0,0,−0.5,0,0.3, 0,0,0.3,0.1,0,0,0,0)	(−0.6,0.7,0, 0,−0.4,0,0, 0,0,−0.3)	(50,30,80,60,24,64,78,82, 38,60,90,28,85,70,38,25, 45,36,87,37,75,86,65,24,57)

and solutions as the initial LBPs, but are less sparse. As an illustration, we performed modifications for the three first problems listed in Table 1 to generate three new LBPs, which are denoted with the same name as the initial problem but with the addition of the letter 'M' (LBP1M, LBP2M, and LBP3M, respectively). These modifications were conducted using the parameters described in [4] as displayed in Table 3.

As a second class of test problems, we considered the NP-hard LCP associated with knapsack problems [13]. As discussed in [13], the knapsack problem can be formulated as an LCP in four different manners. We denote by N_1, N_2, I or P the four types of LCPs that result from these formulations. Furthermore we use the letter A, B, C to denote knapsack problems whose right-hand side are generated so that 75%, 50%, or 25% of the variables are set to one in a feasible solution [13]. Finally the number $t = 22, 52, 102$, and 152 is used to denote the dimension of the LCP_{ABt}.

The third class of test problems is related to the so-called HFP [3, 19]. As discussed in [20], this problem can be reformulated as a linear or quadratic MPEC, depending on whether the chosen norm is l_1 or l_2 respectively. The hinge fitting test problems used in our computational experience are the same as those described in [20].

5.2. DETERMINATION OF A GLOBAL MINIMUM FOR MPEC

In this subsection we describe three sets of experiments. In the first experiment we did not use any disjunctive cuts, and we assumed that the optimal value of MPEC is known a priori. Note that, in general, a solution to the LCP requires the computation of a global minimum to a quadratic program whose objective function is the function $z^T w$. The optimal value in this case is known, as the LCP has a solution if and only if there exists a global minimum for this quadratic program with a value equal to zero.

In the second computational experiment, the performance of the branch-and-bound algorithm without the incorporation of disjunctive cuts and without knowledge of the optimal value was studied, while in the third computational experiment the incorporation of disjunctive cuts for finding better lower bounds was investigated, again assuming an unknown optimal value.

These three experiments are discussed in turn below.

5.2.1. *No Disjunctive Cuts and Assuming a Known Optimal Value*

As stated in [11, 12], and confirmed by our computational experience, the simple solution of the relaxed convex program at each node does not generally lead to good lower bounds. In order to assess the best case scenario, in this first experiment, we used the known optimal objective function value of the MPEC as the lower bound LB(k) for each node k. Hence, the overall algorithm termines as soon as the procedure detects an upper-bound with this optimal value. The results of this experience illustrate the efficiency of the branching and upper bounding techniques and are reported in Tables 4 and 5. In these tables, as well as in the sequel, N represents the dimension of the MPEC, NC, ND and NI are, respectively, the total number of pairs of complementary variables, the number of searched nodes and the number of iterations (pivot steps) performed by the branch-and-bound algorithm, T is the total CPU time in seconds for solving the MPEC, and OP.OBJ. is the global minimum of the MPEC.

The results displayed in Tables 4 and 5 show that the process is quite efficient for finding a global minimum of the MPEC. It should be noted that in all these test problems, the GLCP$_{JL}$ at each node was always solved via a stationary point of QP$_{GLCP_{JL}}$ as discussed in Section 4. This means that there was no need to use the enumerative method [13] for solving these GLCPs in order to compute a stationary point of MPEC$_{JL}$ at each node.

5.2.2. *No Disjunctive Cuts and Unknown Optimal Value*

In this second experiment, the MPECs associated with HFP and BP were solved without previous knowledge of their optimal values and without the incorporation of disjunctive cuts in the branch-and-bound algorithm. In all these test problems the value zero was taken as a trivial lower bound for the objective function value, since this function is either the square of a given expression, or is the sum of nonnegative variables.

Table 6 presents the results obtained in the solution of the euclidean HFP. In this table, as well as in the sequel, the additional parameters NIS, TS, and NDS denote, respectively, the number of iterations (pivotal operations), the time in CPU seconds, and the number of searched nodes until detecting the best incumbent solution.

The results presented in Table 6 shows that when the optimal objective function value is equal to zero, the algorithm easily obtains the global minimum for the problem, revealing little sensitivity to the dimension of the problem. In the remaining cases, the difficulty of finding an optimal solution increases with the dimension of the problem. In Problem 4, the solution was interrupted because the maximum imposed limit of 10^7 iterations (pivot steps) was exceeded.

Table 4. Branch-and-bound algorithm for MPECs with known optimal value

Prob	Nc	N	Ni	T	Nd	Op. Obj.
LBP						
Lbp1	12	12×28	14	0.02	2	6.0000
Lbp1m	12	12×28	10	0.01	1	6.0000
Lbp2	30	30×70	47	0.03	15	16.0000
Lbp2m	30	30×70	21	0.02	1	16.0000
Lbp3	55	55×125	182	0.05	114	30.2000
Lbp3m	55	55×125	87	0.06	5	30.2000
Lbp4	48	48×116	35	0.02	2	6.0000
Lbp5	48	48×116	37	0.02	2	14.0000
Lbp6	110	110×250	65	0.06	3	7.5000
Lbp7	110	110×250	81	0.03	3	23.5000
Lbp8	190	190×410	283	0.02	125	50.5000
QBP						
Qbp1	8	8×20	12	0.02	2	0.3125
Qbp2	16	16×38	25	0.02	4	0.5925
Qbp3	20	20×50	28	0.00	2	0.7900
Qbp4	60	60×145	65	0.02	2	1.0400
Qbp5	80	80×190	135	0.08	16	2.4356
KNAPSACK						
N1a1	44	44×110	66	0.02	2	0.0000
N1a2	104	104×260	11894	2.71	280	0.0000
N1a3	204	204×510	206	0.06	2	0.0000
N1a4	304	304×760	528	0.17	2	0.0000
N1b1	44	44×110	304	0.19	18	0.0000
N1b2	104	104×260	72	0.05	2	0.0000
N1b3	204	204×510	415	0.17	2	0.0000
N1b4	304	304×760	205	0.09	2	0.0000
N1c1	44	44×110	163	0.09	6	0.0000
N1c2	104	104×260	217	0.06	2	0.0000
N1c3	204	204×510	208	0.11	2	0.0000
N1c4	304	304×760	162	0.11	2	0.0000
Ia1	44	44×110	66	0.05	2	0.0000
Ia2	104	104×260	11886	2.85	280	0.0000
Ia3	204	204×510	206	0.06	2	0.0000
Ia4	304	304×760	528	0.27	2	0.0000

Comparing the computational effort until first detecting an optimal solution with the effort of the entire process, we conclude that the difference is not very significant in most of the test examples, except for the most challenging cases such as problems 2–4.

It is also worth mentioning that the objective function value found for Problem 4 by the branch-and-bound method is about 83% better than the one obtained in [20], and that obtained for Problem 3 is slightly better than the value 0.1256 reported in [20].

Table 7 displays the computational results obtained for the linear HFP. The conclusions observed for the quadratic problems generally hold for this linear case as well. Again, the results presented in Table 7 shows that when

Table 5. Branch-and-bound algorithm for MPECs with known optimal value (cont.)

PROB	NC	N	NI	T	ND	OP. OBJ.
KNAPSACK						
IB1	44	44×110	304	0.06	18	0.0000
IB2	104	104×260	72	0.02	2	0.0000
IB3	204	204×510	415	0.19	2	0.0000
IB4	304	304×760	205	0.09	2	0.0000
IC1	44	44×110	163	0.05	6	0.0000
IC2	104	104×260	217	0.06	2	0.0000
IC3	204	204×510	208	0.08	2	0.0000
IC4	304	304×760	162	0.09	2	0.0000
N2A1	44	44×110	191	0.06	2	0.0000
N2A2	104	104×260	443	0.14	2	0.0000
N2A3	204	204×510	974	0.41	2	0.0000
N2A4	304	304×760	593	0.27	2	0.0000
N2B1	44	44×110	1357	0.25	46	0.0000
N2B2	104	104×260	638	0.14	2	0.0000
N2B3	204	204×510	256	0.09	2	0.0000
N2B4	304	304×760	682	0.36	2	0.0000
N2C1	44	44×110	29	0.08	2	0.0000
N2C2	104	104×260	1126	0.33	22	0.0000
N2C3	204	204×510	1269	0.33	2	0.0000
N2C4	304	304×760	733	0.36	2	0.0000
PA1	44	44×110	328	0.11	27	0.0000
PA2	104	104×260	3457	0.56	113	0.0000
PA3	204	204×510	1516	0.36	3	0.0000
PA4	304	304×760	1102	0.36	2	0.0000
PB1	44	44×110	34	0.03	2	0.0000
PB2	104	104×260	154	0.05	2	0.0000
PB3	204	204×510	726	0.17	2	0.0000
PB4	304	304×760	3864	1.23	8	0.0000
PC1	44	44×110	31	0.06	1	0.0000
PC2	104	104×260	72	0.02	1	0.0000
PC3	204	204×510	209	0.08	2	0.0000
PC4	304	304×760	343	0.11	2	0.0000

Table 6. Branch-and-bound algorithm for the Euclidean HFP

PROB	NC	N	NI	T	ND	NIS	TS	NDS	OP.OBJ.
1	30	60×100	14597	1.77	227	11972	1.47	174	1.2024
2	35	70×125	271215	42.08	1855	200661	31.93	1347	0.5307
3	40	80×150	2910363	596.30	15974	1529475	309.62	8252	0.1226
4	45	90×175	$> 10^7$	2968.49	44255	3019314	807.54	12660	0.0029
5	50	100×200	54826	19.0	44	54826	19.0	44	0.0000
6	55	110×225	632	0.25	1	632	0.25	1	0.0000
7	60	120×250	2268	1.03	2	2268	1.03	2	0.0000
8	65	130×275	698	0.38	1	698	0.38	1	0.0000
9	70	140×300	589	0.41	1	589	0.41	1	0.0000
10	75	150×325	823	0.52	1	823	0.52	1	0.0000

Table 7. Branch-and-bound algorithm for the Linear HFP

PROB	N_C	N	N_I	T	N_D	N_{IS}	T_S	N_{DS}	OP.OBJ.
1	30	60×130	119039	10.47	5879	1973	0.19	22	4.3627
2	35	70×160	438178	28.84	11673	1878	0.17	25	2.6287
3	40	80×190	2264946	189.04	54638	117415	10.01	2728	1.1231
4	45	90×220	$>10^7$	1242.56	242161	1267456	140.76	29178	0.1457
5	50	100×250	432	0.09	2	432	0.09	2	0.0000
6	55	110×280	193	0.06	1	193	0.06	1	0.0000
7	60	120×310	221	0.09	1	221	0.09	1	0.0000
8	65	130×340	233	0.13	1	233	0.13	1	0.0000
9	70	140×370	222	0.16	1	222	0.16	1	0.0000
10	75	150×400	255	0.19	1	255	0.19	1	0.0000

the optimal objective value is equal to zero, the process quickly finds an optimal solution, while in the first four test problems, the solution difficulty increases in proportion to the dimension of the problem. Problem 4 was prematurely interrupted because the algorithm exceeded the maximum limit of 10^7 iterations (pivot steps). Also, for the first four problems, the algorithm is able to find an optimal solution, but requires significantly more effort in verifying its optimality. As stated before, for these linear and quadratic MPECs associated with HFP, the GLCP$_{JL}$ instances examined by the branch-and-bound algorithm are easily solved by a special purpose algorithm [21].

Table 8 displays the results obtained for the solution of the BP with the branch-and-bound algorithm. As for the HFP, the results show that the algorithm efficiently finds an optimal solution, but needs greater effort in

Table 8. Branch-and-bound algorithm for BP

PROB	N_I	T	N_D	N_{IS}	T_S	N_{DS}	OP.OBJ.
LBP							
LBP1	17	0.02	8	14	0.02	2	6.0000
LBP1M	28	0.05	4	10	0.05	1	6.0000
LBP2	113	0.02	112	47	0.00	15	16.0000
LBP2M	114	0.03	75	21	0.00	1	16.0000
LBP3	3104	1.26	6144	182	0.05	114	30.2000
LBP3M	3105	0.81	2305	87	0.03	5	30.2000
LBP4	38	0.00	8	35	0.00	2	6.0000
LBP5	76	0.00	48	46	0.00	3	14.0000
LBP6	74	0.03	20	65	0.02	3	7.5000
LBP7	438	0.23	576	81	0.03	3	23.5000
LBP8	49243	47.77	131072	316	0.12	127	50.5000
QBP							
QBP1	15	0.00	5	12	0.00	2	0.3125
QBP2	55	0.02	16	25	0.00	4	0.5925
QBP3	61	0.02	16	28	0.00	2	0.7900
QBP4	143	0.02	32	74	0.02	6	1.0400
QBP5	24630	7.08	8192	1581	0.47	496	2.4356

verifying its optimality. So the incorporation of disjunctive cuts to boost the lower bound computation in the algorithm can potentially improve its performance. This is the subject of the next experiment.

5.2.3. *Disjunctive Cuts and Unknown Optimal Value*

In this third experiment the disjunctive cuts were implemented in the branch-and-bound algorithm according to the procedure discussed in Section 3. Table 9 displays the results obtained in the solution of the LBP and QBP test problems, where C_T denotes the maximum number of cuts generated at each node.

Comparing Tables 8 and 9, it is evident that the use of the disjunctive cuts in the branch-and-bound algorithm is a beneficial strategy, at least for this class of test problems. Indeed, the algorithm finds an optimal solution in many cases without having to enumerate beyond the initial node. But such an effectiveness of the disjunctive cuts does not hold in general. For example, when the disjunctive cuts were used in the solution of the HFP, they did not produce much improvement in the performance of the process. Therefore, the computational experience with this version of the method in the solution of the HFP is not presented in this section.

Since disjunctive cuts are relatively easy to generate and can potentially reduce the search process by tightening the lower bounding relaxations, we

Table 9. Branch-and-bound algorithm with disjunctive cuts

Prob	N_I	T	N_D	C_T	Obj
LBP					
Lbp1	22	0.00	1	11	6.000
Lbp1m	197	0.02	5	12	6.000
Lbp2	53	0.01	1	24	16.000
Lbp2m	62	0.01	1	17	16.000
Lbp3	102	0.03	1	51	30.200
Lbp3m	116	0.02	1	35	30.200
Lbp4	43	0.02	1	11	6.000
Lbp5	59	0.00	1	23	14.000
Lbp6	74	0.01	1	16	7.500
Lbp7	105	0.03	1	38	23.500
Lbp8	170	0.14	1	74	50.500
QBP					
Qbp1	22	0.00	3	2	0.313
Qbp2	45	0.00	5	4	0.593
Qbp3	78	0.02	9	4	0.790
Qbp4	164	0.05	16	5	1.040
Qbp5	4807	1.16	515	13	2.436

recommend their incorporation within the implementation of the proposed branch-and-bound algorithm.

References

1. Balas, E. (1979), Disjunctive programming, *Annals of Discrete Mathematics* 5, 3–51.
2. Bard, J. and Moore, J. (1990), A branch-and-bound algorithm for the bilevel linear program, *SIAM Journal on Scientific and Statistical Computing* 11, 281–292.
3. Breiman, L. (1993), Hinging hyperplanes for regression, classification and function approximation, *IEEE Transactions on Information Theory* 39, 999–1013.
4. Calamai, P. and Vicente, L. (1993), Generating linear and linear-quadratic bilevel programming problems, *SIAM Journal on Scientific Computing* 14, 770–782.
5. Calamai, P. and Vicente, L. (1994), Generating quadratic bilevel programming test problems, *ACM Transactions on Mathematical Software* 20, 103–119.
6. Facchinei, F., Jiang, H. and Qi, L. (1999), A smoothing method for mathematical programs with equilibrium constraints, *Mathematical Programming* 85, 107–134.
7. Fletcher, R., Leyffer, S. and Toint, P. (2002), On the global convergence of a filter-sqp algorithm, *SIAM Journal on Optimization* 13(1), 44–59.
8. Fukushima, M., Luo, Z. and Pang J. (1998), A globally convergent sequential quadratic programming algorithm for mathematical programs with linear complementarity constraints, *Computational Optimization and Applications* 10, 5–34.
9. Hansen, P., Jaumard, B. and Savard, G. (1992), New branch-and-bound rules for linear bilevel programming, *SIAM Journal on Scientific Computing* 13(5), 1194–1217.
10. Jiang, H. and Ralph, D. (2000), Smooth sqp methods for mathematical programs with nonlinear complementarity constraints, *SIAM Journal on Optimization* 10(3), 779–808.
11. Júdice, J. and Faustino, A. (1991), A computational analysis of lcp methods for bilinear and concave quadratic programming, *Computers and Operations Research* 18, 645–654.
12. Júdice, J. and Faustino, A. (1992), A sequential lcp algorithm for bilevel linear programming, *Annals of Operations Research* 34, 89–106.
13. Júdice, J., Faustino, A. and Ribeiro, I. (2002), On the solution of np-hard linear complementarity problems, TOP-*Sociedad de Estatística e Investigacion Operativa* 10(1), 125–145.
14. Júdice, J., Sherali, H., Ribeiro, I. and Faustino, A. (2005), A complementarity active-set algorithm for mathematical programming problems with equilibrium constraints, Working Paper.
15. Júdice, J. and Vicente, L. (1994), On the solution and complexity of a generalized linear complementarity problem, *Journal of Global Optimization* 4, 415–424.
16. Luo, Z., Pang, J. and Ralph, D. (1997), *Mathematical Programs with Equilibrium Constraints*, Cambridge University Press, New York.
17. Murtagh, B. and Saunders, A. (1983), MINOS 5.0 User's Guide, Technical Report SOL 83-20, Department of Operations Research, Stanford University.
18. Nocedal, J. and Wright, S. (1999), *Numerical Optimization*, Springer-Verlag, New York, N.Y.
19. Pucar, P. and Sjoberg, J. (1998), On the hinge finding algorithm for hinging hyperplanes, *IEEE Transactions on Information Theory* 44, 1310–1318.
20. Queiroz, M., Humes Jr, C. and Júdice, J. (2004), On finding global optima for the hinge fitting problem, *Computers and Operations Research* 31, 101–122.
21. Ribeiro, I. (2005), Global Optimization and Applications to Structural Engineering (in Portuguese), PhD thesis, University of Porto, Porto.

22. Scholtes, S. (1999), Active set methods for inverse linear complementarity problems, Judge Institute of Management Research Paper Series No. 28/1999.
23. Sherali, H. and Shetty, C. (1980), Optimization with disjunctive constraints, In: Beckman, M. and Künzi, H. (eds.), *Lecture Notes in Economics and Mathematical Systems*, Vol. 181, Springer-Verlag, New York, N.Y.
24. White, D. and Annandalingam, G. (1993), A penalty function approach for solving bilevel linear programs, *Journal of Global Optimization* 3, 397–419.