Algorithms and applications for a class of bilevel MILPs

Pierre-Louis Poirion[a], Sonia Toubaline[b], Claudia D'Ambrosio[c], Leo Liberti[c]

[a] *Huawei Technologies Mathematical and Algorithmic Sciences Laboratory, Paris, France*
[b] *Université Paris Dauphine, PSL Research University, CNRS, LAMSADE, 75016 Paris, France*
[c] *LIX CNRS (UMR7161), École Polytechnique, 91128 Palaiseau, France*

Abstract

We study a class of bilevel mixed-integer linear programs with the following restrictions: all upper level variables x are binary, the lower level variables y occur in exactly one upper level constraint $\gamma x + \beta y \geq c$, and the lower level objective function is $\min_y \beta y$. We propose a new cut generation algorithm to solve this problem class, based on two simplifying assumptions. We then propose a row-and-column generation algorithm that works independently of the assumptions. We apply our methods to two problems: one is related to the optimal placement of measurement devices in an electrical network, and the other is the *minimum zero forcing set* problem, a variant of the dominating set problem. We exhibit computational results of both methods on the application-oriented instances as well as on randomly generated instances.

Keywords: Bilevel MILP, Power edge set, Zero forcing set.

1. Introduction

1.1. Bilevel programming

A Bilevel Mixed-Integer Linear Program (BMILP) is a generalization of a standard Mixed-Integer Linear Program (MILP), which models a hierarchical decision process. The general formulation of a BMILP is:

$$(*)\begin{cases} \min\limits_{x} & \alpha^1 x + \alpha^2 y & & \\ & Ax & \geq & b \\ & Gx + Hy & \geq & c \\ & x & \in & \mathcal{X} \\ & y & \in & \arg\min\limits_{y \in \Omega(x) \cap \mathcal{Y}} \beta(x)\, y, \qquad \text{(LL)} \end{cases}$$

where:

- $\alpha^1, \alpha^2 \in \mathbb{Q}^q$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^p$;
- $A \in \mathbb{Q}^{m \times n}$, $G \in \mathbb{Q}^{p \times n}$, $H \in \mathbb{Q}^{p \times q}$;

Email addresses: plpoirion@gmail.com (Pierre-Louis Poirion), sonia.toubaline@dauphine.fr (Sonia Toubaline), dambrosio@lix.polytechnique.fr (Claudia D'Ambrosio), liberti@lix.polytechnique.fr (Leo Liberti)

- $\beta(x) \in \mathbb{Q}^q$ is a linear function of x;
- $\Omega(x)$ is a (lower level) polyhedron for each x;
- $\mathcal{X} = \mathbb{N}^{n_1} \times \mathbb{R}_+^{n_2}$, $\mathcal{Y} = \mathbb{N}^{q_1} \times \mathbb{R}_+^{q_2}$ (with $n = n_1 + n_2$ and $q = q_1 + q_2$).

The decision variables are split into two classes: upper level variables x and lower level variables y. Similarly, the functions $(x, y) \mapsto \alpha^1 x + \alpha^2 y$ and $(x, y) \mapsto \beta(x)y$, are the upper and lower level objective functions.

The BMILP is a non-convex mixed-integer program, known to be **NP**-hard [8] (a thorough study of complexity of the bilevel knapsack problem can be found in [4]). Application-wise, bilevel and multilevel programming are extremely useful paradigms as they model dependencies within hierarchically organized entities (such as, e.g., industrial headquarters and its branches). The bilevel optimization literature often considers bilevel programming as essentially ill-defined, since there may be multiple (global) optima at the lower level, and each might influence the "behaviour" of the upper level as regards its own optimality. If one defines the upper level optima to be the best possible, given that all lower level optima are allowed to occur, one can eschew this ambiguity: this is also sometimes known as the "optimistic strategy" in the bilevel optimization community, since it is equivalent to the lower level "choosing" the optimum which "favors" the upper level.

To the best of our knowledge no existing approach is able to solve $(*)$ in full generality (again, the special case of the bilevel knapsack problem was addressed in [5]). Moreover, as the interest in this problem has grown over the past years, several methods have been developed to solve some sub-classes of $(*)$. One of the best studied cases is where the lower level variables are all continuous, i.e. $\mathcal{Y} = \mathbb{R}_+^q$ [6]. Under this condition, the lower level problem is convex and regular, and it can be replaced by its Karush-Kuhn-Tucker (KKT) conditions, yielding a single level reformulation of the problem. Several methods exist to solve this variant [1, 21].

Another well studied case is when $H = 0$ (i.e. the lower level variables do not appear in the upper level constraints), and $\beta(x) = \beta$ (i.e. the lower level objective function coefficients do not depend on the upper level variables), see [17, 23, 31]. In this case, at each iteration the decision on the upper level and the lower level variables is taken separately. Specifically, one can alternate between solving upper and lower level: solve the upper level first, obtain optimal values for x, fix them within the lower level polyhedron $\Omega(x)$ during the solution of the lower level, and repeat. This is not the only possible solution approach: in [7], for example, the authors generalize a MILP branch-and-cut algorithm to the bilevel setting by introducing integer no-good cuts in order to separate bilevel infeasible solutions from the convex hull of bilevel feasible solutions. Another classic separation approach is applied to a relaxation of the problem in [17]. In [23], a basic implicit enumeration scheme is developed in order to find good feasible solutions within relatively few iterations. In [31], a scheme based on a reformulation and decomposition strategy is presented. The decomposition algorithm is based on the row-and-column generation method. The number of iterations needed for the convergence of this algorithm is shown to be finite.

The case where both sets $\mathcal{X}$ and $\mathcal{Y}$ are binary and $\beta(x)$ is a constant is considered in [29]. The problem is reduced to a multilevel Linear Program (LP) using a penalty function method. Solving the multilevel LP does not appear to

offer a practically viable solution method. In [16, 24], the lower level polyhedron $\Omega(x)$ does not depend on x, and the problem is solved using row generation.

More recently, Fischetti et al. [10] proposed a branch-and-cut approach to solve a fairly large subclass of $(*)$, i.e. BMILPs with lower level objective function independent from the upper level constraints.

1.2. Applications

We present two important applications of the BMILP $(*)$: one from the energy industry (placement of electrical measuring devices in smart grids, also discussed in more depth in [28, 26]), and one in graph theory (determination of the rank of a graph using bilevel programming: this is an original contribution as far as we know).

1.2.1. Observability in a smart grid

An important problem arising in the design of smart grids is the placement of various devices in an electrical network. In particular, measuring devices called Phasor Measuring Units (PMU) are crucial in observing the state (voltage, current) of the grid. The network is modelled by a graph where edges correspond to transmission lines between (sub)stations, represented by nodes. The problem of placing the minimum number of PMUs on the edges of the graph, so as to be able to observe the state of the whole grid is called the Power Edge Set (PES) problem [28]. A graph is said to be fully observed if the voltage is known at each node, and the current known on each edge.

The PES can also be seen as the "edge version" of the Power Dominating Set (PDS) problem [13], which has been largely studied in literature. The PDS is shown to be **NP**-complete even for bipartite, chordal graphs [13] and planar bipartite graphs [3], and polynomial for trees and grids [9]. Different solution methods have been proposed to solve the PDS [19, 20].

The PES is also **NP**-complete [27]. Its bilevel formulation occurs from a fixed-point condition over the repeated application of two observability rules over the network [26]. It was the solution procedure proposed in [26] that gave us the idea for the rather general algorithms proposed in the present paper.

1.2.2. Zero forcing set

We also consider an application of our techniques to the Minimum Zero Forcing Set (MZFS) problem, introduced in [22], which consists in finding the minimum set of nodes in a graph that covers all the nodes given a specific propagation rule. Zero forcing sets are useful when computing the rank of a graph, i.e. the minimum rank over all its adjacency matrices. By formulating the MZFS as a BMILP, we will provide the first practically useful method for solving the MZFS problem.

Let $G = (V, E)$ be a graph where every vertex $v \in V$ has to be assigned an initial color, either in black or white. A *zero forcing set* (ZFS) of G is defined as follows [22]:

- *Color-change rule:* if $v \in V$ is a black vertex and all its neighbors (denoted by $N(v)$) are black besides one, say u, then change the color of u to black.
- Given a coloring of G, the *derived coloring* is the result of applying the color-change rule until no more changes are possible.

- A ZFS for G is a subset of black vertices Z such that, if all the remaining vertices $V \setminus Z$ are white, the derived coloring of G is all black.
- $Z(G)$ is the ZFS with minimum cardinality $|Z|$ over all ZFSs $Z \subseteq V$.

In [22], the authors prove that the size of a ZFS gives a bound on the minimum rank of the graph. We formulate the MZFS as a BMILP in Section 6. The ZFS can also represent the propagation of various quantities (including influence or information) in system networks [14].

1.3. Content of the paper

In this paper we propose what we believe to be the first practically viable method to solve the following BMILP variant: the upper level variables x are binary, G and H are one-row matrices, and $\beta(x) = \beta = H^\top$. The crucial ideas (in preliminary form) can be found in the 2015 technical report [25]. An important theoretical contribution is that we provide a solution method for the case where lower level variables appear in the upper level constraints.

We prove that, under certain assumptions, it is possible to reformulate the BMILP $(*)$ into a single-level Binary Linear Program (BLP). This BLP, however, cannot be solved directly since the explicit polyhedral description of its constraints is not known. We therefore propose a finitely terminating cut generation algorithm. When the aforementioned assumptions do not hold, we propose a row-and-column generation framework.

The rest of the paper is organized as follows: in Section 2 we describe the single-level reformulation under two specific monotonicity assumptions. In Section 3 we propose a cut generation algorithm to solve the bilevel problem. In Section 4 we relax the assumptions, and derive monotonicity from a simple standard form of $(*)$. In Section 5, we apply our framework to the PES problem. In Section 6 we apply our framework to the MZFS problem.

2. Bilevel Binary Linear Program

The standard Bilevel Binary Linear Program (BBLP) formulation we are interested in is given by the following formulations:

$$(\dagger) \left\{ \begin{array}{ll} \min\limits_{x} & \alpha x \\ & Ax \geq b \\ & x \in \{0,1\}^n \\ & \beta y \geq c - Gx \\ & y \in \left\{ \begin{array}{ll} \arg\min\limits_{y} & \beta y \\ & y \in \Omega(x) \\ & y \in \mathcal{Y} \end{array} \right. \end{array} \right. \equiv \left\{ \begin{array}{ll} \min\limits_{x} & \alpha x \\ & Ax \geq b \\ & x \in \{0,1\}^n \\ & f(x) \geq c - \gamma x \\ & f(x) = \left\{ \begin{array}{ll} \min\limits_{y} & \beta y \\ & y \in \Omega(x) \\ & y \in \mathcal{Y} \end{array} \right. \end{array} \right.$$

which are equivalent since we stipulated that G is a one-row matrix. Moreover, $\alpha \in \mathbb{Q}^n$, $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $G \in \mathbb{Q}^{1 \times n}$, $c \in \mathbb{Q}$, $\beta \in \mathbb{Q}^q$, $\Omega(x)$ is a (lower level) polyhedron for each x, $\gamma = G^\top$, and $\mathcal{Y} = \mathbb{N}^{q_1} \times \mathbb{R}_+^{q_2}$. The x variables are the upper level variables and the y variables are the lower level variables.

Notice that here we consider that the optimization is done over the field $\mathbb{R}$ of real numbers, which does not include $+\infty$. Hence we always have $f(x) < +\infty$, which justifies the fact the two programs above are indeed equivalent. In particular, if $\Omega(x)$ is empty, y does not exist in the lower level of the LHS of (†), which makes (†) infeasible; likewise, $f(x)$ takes no value in the RHS of (†), which, again, makes (†) infeasible.

In order to formulate the BBLP as a single level program, we use the following well-known lemma.

Lemma 1 *For any $X \subseteq \{0,1\}^n$, $\mathsf{conv}(X) \cap \{0,1\}^n = X$.*

Proof. By definition we have $X \subset \{0,1\}^n$ and $X \subseteq \mathsf{conv}(X)$, hence X is a subset of their intersection. Conversely, let $x \in \mathsf{conv}(X)$ and $x \in \{0,1\}^n$; and suppose, to get a contradiction, that $x \notin X$. Then there must be $y \neq z \in X$ such that x is in the segment between y and z. However, because y, z are in a hypercube, their linking segment only contains points with at least one fractional component, which contradicts $x \in \{0,1\}^n$, as claimed. □

As a direct corollary, Lemma 1 has the more surprising formulation below.

Corollary 1 *Every subset of hypercube vertices has a linear description.*

In other words, no matter how many nonlinear terms the original description of a subset of hypercube vertices might involve, there is always a reformulation of that description that uses linear forms only.

Let $\mathcal{F} = \{x \in \{0,1\}^n \mid Ax \geq b \ \wedge f(x) \geq c - \gamma x\}$ be a linear description of the feasible set of the BBLP. By Lemma 1, the BBLP (†) can be rewritten as the following Binary Linear Program (BLP):

$$\min_x \{\alpha x \mid x \in \mathsf{conv}(\mathcal{F}) \cap \{0,1\}^n\}. \tag{1}$$

Notice that, in general, a feasible upper level variable x for the continuous relaxation of the BLP in Eq. (1) will not be feasible for the continuous relaxation (in the x variables) of (†). More precisely: in general the continuous relaxation of the BBLP (†) is not convex. However, the continuous relaxation of its linear reformulation (i.e. the BLP in Eq. (1)) is convex, since its description is linear. On the other hand, the polyhedral description of $\mathsf{conv}(\mathcal{F})$ is unknown, so we cannot solve the BLP in Eq. (1) using well-known MILP solution methods.

In order to find a polyhedral description of $\mathsf{conv}(\mathcal{F})$, we look for a polyhedron $\mathcal{P}$ such that $\mathcal{P} \cap \mathbb{Z}^n = \mathcal{F}$. In other words, a polyhedron $\mathcal{P}$ containing $\mathsf{conv}(\mathcal{F})$, but no other binary point than those in $\mathsf{conv}(\mathcal{F})$. The BBLP (†) can thus be formulated as:

$$\min_x \{\alpha x \mid x \in \mathcal{P} \cap \{0,1\}^n\}.$$

Next, we consider the *Restricted*-BBLP, i.e. the BBLP problem under the two additional assumptions below.

Hypothesis 1 *For all $x, x' \in \{0,1\}^n$ such that $Ax \geq b$ and $Ax' \geq b$, if $x \leq x'$ then the lower level polyhedron $\Omega(x)$ is such that $\Omega(x) \supseteq \Omega(x')$. In other words, f is a non-decreasing function over the set of x such that $Ax \geq b$.*

Hypothesis 2 *For all* $i \in \{1, \ldots, n\}$, $\gamma_i \geq 0$.

We remark that Hypotheses 1-2 were naturally motivated by our applications (see Sect. 5-6). We shall discuss our most general case in Sect. 4.

Given $S \subseteq \{0,1\}^n$, let $M(S)$ be the set of $\leq$-maximal points in S, i.e.

$$M(S) = \{s \in S \mid \forall t \in S \; (t \geq s \rightarrow t = s)\}.$$

For any $v \in \{0,1\}^n$, let $\zeta(v) = \{i \leq n \mid v_i = 0\}$ be the complement of the support of v, where we recall that the support of a vector is the set of indices of its non-zero elements. Let

$$\bar{\mathcal{F}} = \{x \in \{0,1\}^n \mid Ax \geq b \wedge f(x) < c - \gamma x\}$$

be the subset of $\{0,1\}^n$ which is feasible w.r.t. the "easy" constraints of the upper level, but infeasible w.r.t. the "hard" constraints (i.e. those involving the lower level). If $\bar{\mathcal{F}} = \varnothing$, the lower level problem can simply be removed. Therefore, we assume in the following that $\bar{\mathcal{F}} \neq \varnothing$. We prove in the next section that a polyhedral description of $\mathsf{conv}(\mathcal{F})$ is obtained by adding valid inequalities generated using points in the set $M(\bar{\mathcal{F}})$ of $\leq$-maximal upper level points infeasible w.r.t. the lower level.

Motivated by our applications (Sections 5 and 6), we assume, in the next sections, that the lower level polyhedron $\Omega(x)$ is never empty. We will however also remark how to deal with the case $\Omega(x) = \varnothing$.

3. A solution method for the *Restricted*-BBLP

3.1. Strengthening the relaxation of $\mathsf{conv}(\mathcal{F})$

In order to iteratively strengthen the relaxation $\mathcal{P}$ of the upper level feasible set $\mathcal{F}$, we introduce a class of valid inequalities for $\mathcal{F}$.

Proposition 1 *For all* $\bar{x} \in \bar{\mathcal{F}}$, $\sum_{i \in \zeta(\bar{x})} x_i \geq 1$ *is a valid inequality for* $\mathcal{F}$.

Proof. Assume there exists $x \in \mathcal{F}$ such that $\sum_{i \in \zeta(\bar{x})} x_i < 1$. Since $x \in \{0,1\}^n$ we have that $\sum_{i \in \zeta(\bar{x})} x_i = 0$, which implies that, for all $i \in \zeta(\bar{x})$, $x_i = 0$. Hence $x \leq \bar{x}$ and $f(x) \leq f(\bar{x}) < c - \gamma\bar{x}$ by Hypothesis 1. Therefore by Hypothesis 2, we conclude that $f(x) < c - \gamma x$, contradicting the assumption. □

Let

$$\mathcal{P} = \{x \in [0,1]^n \mid Ax \geq b \wedge \forall \bar{x} \in M(\bar{\mathcal{F}}) \sum_{i \in \zeta(\bar{x})} x_i \geq 1\}$$

and $\mathcal{P}_I = \mathcal{P} \cap \mathbb{Z}^n$. By Proposition 1, we have that $\mathcal{F} \subseteq \mathcal{P}_I$. In the following, we prove that $\mathcal{P}_I = \mathcal{F}$ and that for all $\bar{x} \in M(\bar{\mathcal{F}})$, the constraint $\sum_{i \in \zeta(\bar{x})} x_i \geq 1$ is a facet of $\mathcal{P}$.

Proposition 2 *The following statements hold: (i)* $\mathcal{P}_I = \mathcal{F}$*; (ii) for each* $\bar{x} \in M(\bar{\mathcal{F}})$, $\sum_{i \in \zeta(\bar{x})} x_i \geq 1$ *is a facet of* $\mathcal{P}$.

PROOF. (i) Let $x \in \mathcal{P}_I$ and suppose, to get a contradiction, that $x \notin \mathcal{F}$. By definition, $x \in \bar{\mathcal{F}}$. Pick a $\bar{x} \in M(\bar{\mathcal{F}})$ such that $\bar{x} \geq x$. Since $x \in \mathcal{P}_I$ and $\bar{x} \in M(\bar{\mathcal{F}})$, we have, by definition of $\mathcal{P}_I$, that $\sum_{i \in \zeta(\bar{x})} x_i \geq 1$ (¶). However, since $\bar{x}, x$ are binary vectors and $\bar{x} \geq x$, it follows that $x_i = 0$ for all $i \leq n$ such that $\bar{x}_i = 0$. This means that $\sum_{i \in \zeta(\bar{x})} x_i = 0$, against (¶). Therefore $x \in \mathcal{F}$ and $\mathcal{P} \cap \mathbb{Z}^n = \mathcal{F}$ as claimed.
(ii) Let $\bar{x} \in M(\bar{\mathcal{F}})$. Let us now prove that $\sum_{i \in \zeta(\bar{x})} x_i \geq 1$ (§) is a facet of $\mathcal{P}$. Suppose, to get a contradiction, that removing (§) from the definition of $\mathcal{P}$ yields $\mathcal{P}$ again. Since $\bar{x} \in M(\bar{\mathcal{F}})$, $\bar{x} \in \bar{\mathcal{F}}$. Note that for any $x' \in \bar{\mathcal{F}}$ we have $\sum_{i \in \zeta(x')} x'_i = 0$ by definition of ζ. Thus, by Proposition 1 and by definition of $\mathcal{P}$, $x' \notin \mathcal{P}$ and in particular $\bar{x} \notin \mathcal{P}$. Then there must be some inequality in the definition of $\mathcal{P}$ which cuts off $\bar{x}$, and none of the inequalities in $Ax \geq b$ qualify since, by definition of $\bar{\mathcal{F}}$, $A\bar{x} \geq b$. Hence it must be one of the other inequalities, say $\sum_{i \in \zeta(x')} x_i \geq 1$ for some $x' \in M(\bar{\mathcal{F}})$ with $x' \neq \bar{x}$. This means that $\sum_{i \in \zeta(x')} \bar{x}_i = 0$ which implies that $\bar{x}_i = 0$ for all $i \leq n$ such that $x'_i = 0$. Together with $x' \neq \bar{x}$, it follows that $\bar{x} < x'$, which contradicts the maximality of $\bar{x}$ in $\bar{\mathcal{F}}$. □

3.2. A cut generation algorithm

Now that we have a method for strengthening $\mathcal{P}$, we can build a cut generation algorithm to solve P under Hypotheses 1 and 2.

Algorithm 1 describes our cut generation algorithm for the *Restricted*-BBLP. Its main framework is the following: assume that, at step k, we have a set $\mathcal{P}^k$ such that $\mathsf{conv}(\mathcal{F}) \subseteq \mathcal{P}^k$ and let x^k be an optimal solution of the relaxed problem

$$Q^k \equiv \min_x \{\alpha x \mid x \in \mathcal{P}^k,\ x \in \{0,1\}^n\}.$$

Then x^k induces a lower bound for the *Restricted*-BBLP. Therefore, if $x^k \notin \mathcal{F}$, it follows that $x^k \in \bar{\mathcal{F}}$. Based on Hypothesis 1, we shall show in the next section how to formulate and solve an auxiliary MILP, called FM, to find an element $\bar{x}^k \in M(\bar{\mathcal{F}})$ (the set of $\leq$-maximal upper level points infeasible w.r.t. the lower level). Therefore, at each iteration of Algorithm 1, either we find an optimal solution, or we add a facet of $\mathcal{P}$. Because rational polyhedra possess a finite number of facets, Algorithm 1 will converge in a finite number of iterations.

3.3. Finding maximal lower level infeasible points

We explain how, for an infeasible solution $\bar{x} \in \bar{\mathcal{F}}$, we can compute a maximal infeasible solution $x' \geq \bar{x}$, i.e. $x' \in M(\bar{\mathcal{F}})$. Consider, for a small enough $\varepsilon > 0$, the following single-level MILP:

$$(\mathsf{FM}) \quad \begin{cases} \max\limits_{x,y} & \sum\limits_{i \in \zeta(\bar{x})} x_i & & \\ & \beta\, y + \gamma\, x & \leq & c - \varepsilon \\ & Ax & \geq & b \\ & x & \geq & \bar{x} \\ & x & \in & \{0,1\}^n \\ & y & \in & \Omega(x) \cap \mathcal{Y}. \end{cases}$$

Algorithm 1 Cut generation algorithm to solve the *Restricted*-BBLP

$\mathcal{P}^0 \leftarrow \{x \in [0,1]^n \mid Ax \geq b\}$
$k \leftarrow 0$
$x^* \leftarrow \infty$
Condition $\leftarrow$ False
while Condition = False **do**
  Solve the MILP Q^k and let x^k be an optimal solution
  $x^* \leftarrow x^k$
  if $f(x^*) \geq c - \gamma x^*$ **then**
    Condition $\leftarrow$ True
  else
    Find $x \geq x^*$ such that $x \in M(\bar{\mathcal{F}})$ (solve FM, see Section 3.3)
    $x^* \leftarrow x$
    $\mathcal{P}^{k+1} \leftarrow \mathcal{P}^k \cap \{x \mid \sum_{i \in \zeta(x^*)} x_i \geq 1\}$
    $k \leftarrow k + 1$
return x^*

Let $x \in \mathcal{F}$. Notice that, as $x \in \{0,1\}^n$ belongs to a finite set, there exists a small enough $\varepsilon > 0$ such that x satisfies $Ax \geq b$ but not $\beta\, y + \gamma\, x \leq c - \varepsilon$, for any y. Now, we claim that the optimal solution x^* of FM is maximally infeasible, i.e. it belongs to $M(\bar{\mathcal{F}})$. Indeed, since there exists $y \in \Omega(x^*)$ such that $\beta\, y + \gamma\, x^* < c$, we have that $f(x^*) < c - \gamma\, x^*$, i.e. $x^* \in \bar{\mathcal{F}}$. Furthermore, $x^* \geq \bar{x}$ hence x^* dominates $\bar{x}$. If $x^* \notin M(\bar{\mathcal{F}})$, there must exist $x' \in \bar{\mathcal{F}}$ such that $\sum_{i \in \zeta(\bar{x})} x'_i > \sum_{i \in \zeta(\bar{x})} x^*_i$ which implies, by the optimality of x^*, that $\beta\, y + \gamma\, x' > c - \varepsilon$ for all y. Therefore, by the definition of ε, we have $x' \in \mathcal{F}$ which contradicts the initial assumption.

Notice that if ε is not small enough, then the optimal solution x^* of FM will dominate $\bar{x}$ but may not be maximal.

We now explain how to extend our study to the case where the lower level polyhedron $\Omega(x)$ is empty. Let $\bar{x} \in \{0,1\}^n$ such that $\Omega(\bar{x}) = \varnothing$. Then, by Hypothesis 1, for all $x \geq \bar{x}$, $\Omega(x) = \varnothing$. Hence, similarly to Proposition 1, we can prove that the inequality $\sum_{i:\bar{x}=1} x_i \leq -1 + \sum_i \bar{x}_i$ is valid.

4. A solution method for the BBLP

In this section we explain how and to what extent it is possible to generalize the previous approach when Hypotheses 1 and 2 do not hold. Accordingly, we shall propose a row-and-column generation algorithm for the BBLP (†).

4.1. Generalized domination in $\bar{\mathcal{F}}$

Let $B \in \mathbb{Q}^{r \times q}$, $C \in \mathbb{Q}^{r \times n}$, and $d \in \mathbb{Q}^r$ such that for all $x \in \{0,1\}^n$ the lower level polyhedron is

$$\Omega(x) = \{y \in \mathbb{R}^q_+ \mid By \geq d + Cx\}. \tag{2}$$

Consider now $\bar{x} \in \bar{\mathcal{F}}$. In the following, we first prove that binary vectors dominated by $\bar{x}$ with respect to C and γ are infeasible. We also develop some

nonlinear inequalities to separate these infeasible points from the feasible region $\mathcal{F}$. We then present a linear reformulation of these inequalities.

Lemma 2 *Let $\bar{x} \in \bar{\mathcal{F}}$. For all $x \in \{0,1\}^n$ such that $Ax \geq b$, $Cx \leq C\bar{x}$ and $\gamma x \leq \gamma \bar{x}$, we have that $x \notin \mathcal{F}$.*

Proof. Let $\bar{y} \in \Omega(\bar{x})$ such that $f(\bar{x}) = \beta\bar{y} < c - \gamma\bar{x}$. Since $Cx \leq C\bar{x}$, we have that $B\bar{y} \geq d + Cx$. Hence $\bar{y} \in \Omega(x)$. Therefore $f(x) \leq f(\bar{x}) < c - \gamma\bar{x}$ ($\sharp$). However, since $\gamma x \leq \gamma\bar{x}$ and $\bar{x}$ is subtracted from c in the RHS of ($\sharp$), we deduce that $f(x) < c - \gamma x$ and therefore $x \notin \mathcal{F}$. □

Let $\mathcal{C}(\bar{x}) = \{x \in \mathbb{R}^n \mid \exists z \in \mathbb{R}^n \ (x = \bar{x} + z \wedge Cz \leq 0 \wedge \gamma z \leq 0)\}$. It is easy to see that $Cx \leq C\bar{x}$ and $\gamma x \leq \gamma\bar{x}$ if and only if $x \in \mathcal{C}(\bar{x})$. Let $\bar{x} \in \bar{\mathcal{F}}$ and $\Delta_{\bar{x}} : \{0,1\}^n \mapsto \mathbb{R}_+$ be the function defined by $\min_{x'}\{\|\bar{x} - x'\|_1 \mid x' \in \mathcal{C}(\bar{x})\}$.

Lemma 3 *Let $\bar{x} \in \bar{\mathcal{F}}$. $\Delta_{\bar{x}}$ defines a semimetric to the set $\mathcal{C}(\bar{x})$, i.e. for all $x \in \{0,1\}^n$, $\Delta_{\bar{x}}(x) \geq 0$ and $\Delta_{\bar{x}}(x) = 0$ if and only if $x \in \mathcal{C}(\bar{x})$.*

Proof. The function $\Delta_{\bar{x}}$ is positive by definition. Assume that there exists $x \in \{0,1\}^n$ such that $\Delta_{\bar{x}}(x) = 0$ and let x^* be the optimal solution of the optimization problem associated to $\Delta_{\bar{x}}(x)$. By definition, $x = x^*$, and then $x \in \mathcal{C}(\bar{x})$. Conversely, if $x \in \mathcal{C}(\bar{x})$ then we can set $x' = x$ which concludes the proof. □

The optimization problem associated to $\Delta_{\bar{x}}(x)$ can be rewritten as follows:

$$\Delta_{\bar{x}}(x) = \left\{ \begin{array}{lrcl} \min\limits_{z,e,f} & \sum\limits_{i=1}^{n} e_i + \sum\limits_{i=1}^{n} f_i & & \\ & \bar{x} + z + e - f & = & x \\ & Cz & \leq & 0 \\ & \gamma z & \leq & 0 \\ & e, f \in \mathbb{R}^n_+, \ z & \in & \mathbb{R}^n. \end{array} \right.$$

4.2. Valid nonlinear cuts

Proposition 3 *If $\bar{x} \in \bar{\mathcal{F}}$ then there exists $\delta_{\bar{x}} > 0$ such that inequality $\Delta_{\bar{x}}(x) \geq \delta_{\bar{x}}$ is valid for $\mathcal{F}$.*

Proof. Let $\delta_{\bar{x}} = \min_{x \in \mathcal{F}} \Delta_{\bar{x}}(x)$. Since $\mathcal{F}$ is a finite set, $\delta_{\bar{x}}$ exists. Suppose, to get to a contradiction, that $\delta_{\bar{x}} = 0$. Then there must exist $x \in \mathcal{F}$ such that $\Delta_{\bar{x}}(x) = 0$, which implies that $x \in \mathcal{C}(\bar{x})$ by Lemma 3. Also, since $x \in \mathcal{F}$, we have in particular that $Ax \geq b$. However, Lemma 2 states that for each $\bar{x} \in \bar{\mathcal{F}}$ and $x \in \{0,1\}$ with $Ax \geq b$, if $x \in \mathcal{C}(\bar{x})$ then $x \notin \mathcal{F}$, which contradicts the assumption $x \in \mathcal{F}$; hence $\delta_{\bar{x}} > 0$. Furthermore, by definition of $\delta_{\bar{x}}$, we have that the inequality $\Delta_{\bar{x}}(x) \geq \delta_{\bar{x}}$ is valid for $\mathcal{F}$. □

We now show how to compute $\delta_{\bar{x}}$. Let $\bar{x} \in \bar{\mathcal{F}}$, $\varepsilon > 0$, and consider the following equivalent programs:

$$D_{\bar{x}}(\varepsilon) = \left\{ \begin{array}{ll} \min\limits_{x} & \Delta_{\bar{x}}(x) \\ & Ax \geq b \\ & \Delta_{\bar{x}}(x) \geq \varepsilon \\ & x \in \{0,1\}^n \end{array} \right. \equiv \left\{ \begin{array}{ll} \min\limits_{\substack{x\in\{0,1\}^n \\ e,f\in\mathbb{R}^n_+}} & \sum\limits_{i=1}^n e_i + \sum\limits_{i=1}^n f_i \\ & Ax \geq b \\ & \varepsilon \leq \min\limits_{e,f,z\in\mathbb{R}^n} \quad \sum\limits_{i=1}^n e_i + \sum\limits_{i=1}^n f_i \\ & \qquad \bar{x} + z + e - f = x \\ & \qquad\qquad Cz \leq 0 \\ & \qquad\qquad \gamma z \leq 0. \end{array} \right\}$$

It is easy to see that taking $\varepsilon \leq \delta_{\bar{x}}$ will give an optimal solution (x^*, z^*, e^*, f^*) of $D_{\bar{x}}(\varepsilon)$ such that $\sum\limits_{i=1}^n e_i^* + \sum\limits_{i=1}^n f_i^* = \delta_{\bar{x}}$. Therefore, if for some ε we have $D_{\bar{x}}(\varepsilon) \leq \varepsilon$, then we can decrease the value of ε and repeat the process until $D_{\bar{x}}(\varepsilon) > \varepsilon$, in which case $D_{\bar{x}}(\varepsilon) = \delta_{\bar{x}}$ holds.

Notice that the mathematical program above is written as a bilevel program, however, since all the lower level variables, z, e and f are continuous, it can be rewritten as a MILP (see Problem (3) in Section 4.3 below).

4.3. Linearizing the cuts

Our inequalities are nonlinear since they are expressed as minimization problems. However, since these problems are LPs, we can replace them by their duals.

Proposition 4 *Let $\bar{x} \in \bar{\mathcal{F}}$. The constraint $\Delta_{\bar{x}}(x) \geq \delta_{\bar{x}}$ is equivalent to the system of inequalities:*

$$\left\{ \begin{array}{rcl} (x - \bar{x})(C^\top \sigma_{\bar{x}} + \gamma \vartheta_{\bar{x}}) & \geq & \delta_{\bar{x}} \\ -1 \leq C^\top \sigma_{\bar{x}} + \gamma \vartheta_{\bar{x}} & \leq & 1 \\ \sigma_{\bar{x}} \in \mathbb{R}^r_+,\ \vartheta_{\bar{x}} \in \mathbb{R}_+. & & \end{array} \right.$$

Proof. Let us recall the LP formulation of $\Delta_{\bar{x}}(x)$:

$$\Delta_{\bar{x}}(x) = \left\{ \begin{array}{ll} \min\limits_{z,e,f} & \sum\limits_{i=1}^n e_i + \sum\limits_{i=1}^n f_i \\ & x = \bar{x} + z + e - f \\ & Cz \leq 0 \\ & \gamma z \leq 0 \\ & e, f \in \mathbb{R}^n_+,\ z \in \mathbb{R}^n. \end{array} \right.$$

For all $x \in \{0,1\}^n$, we introduce its dual, $\Pi_{\bar{x}}(x)$:

$$\Pi_{\bar{x}}(x) = \left\{ \begin{array}{ll} \max\limits_{\sigma_{\bar{x}}, \vartheta_{\bar{x}}} & (x - \bar{x})(C\sigma_{\bar{x}} + \gamma\vartheta_{\bar{x}}) \\ & -1 \leq C\sigma_{\bar{x}} + \gamma\vartheta_{\bar{x}} \leq 1 \\ & \sigma_{\bar{x}} \in \mathbb{R}^r_+,\ \vartheta_{\bar{x}} \in \mathbb{R}_+. \end{array} \right.$$

Since $\Delta_{\bar{x}}(x)$ has always a finite optimal solution, by the strong duality theorem of LP, we have that the optimal values of the primal and of the dual are equal.

Therefore the constraint $\Delta_{\bar{x}}(x) \geq \delta_{\bar{x}}$ is equivalent to $\Pi_{\bar{x}}(x) \geq \delta_{\bar{x}}$. However, given the expression of the dual, we conclude that $\Pi_{\bar{x}}(x) \geq \delta_{\bar{x}}$ if and only if there exist $\sigma_{\bar{x}} \in \mathbb{R}^r_+,\ \vartheta_{\bar{x}} \in \mathbb{R}_+$ such that:

$$\mathcal{V}(\bar{x}) = \left\{ \begin{array}{l} (x-\bar{x})(C^\top \sigma_{\bar{x}} + \gamma \vartheta_{\bar{x}}) \geq \delta_{\bar{x}} \\ -1 \leq C^\top \sigma_{\bar{x}} + \gamma \vartheta_{\bar{x}} \leq 1 \end{array} \right.$$

is verified, as claimed. □

Notice that, if x is not fixed, the constraint $(x-\bar{x})(C^\top \sigma_{\bar{x}} + \gamma \vartheta_{\bar{x}}) \geq \delta_{\bar{x}}$ is not linear. Nevertheless, x is a binary variable, and we can reasonably assume that $\sigma_{\bar{x}}$ and $\vartheta_{\bar{x}}$ are bounded, so we can reformulate the products $x_i \sigma_{\bar{x}j},\ x_i \vartheta_{\bar{x}}$ exactly by means of the Fortet reformulation [11].

Notice that the above linearization of $\Delta_{\bar{x}}(x) \geq \delta_{\bar{x}}$ can be applied to linearize the constraint $\Delta_{\bar{x}}(x) \geq \varepsilon$ in $D_{\bar{x}}(\varepsilon)$ (for the computation of $\delta_{\bar{x}}$). Hence the bilevel formulation of $D_{\bar{x}}(\varepsilon)$ can be reformulated into the following program:

$$\left. \begin{array}{rcl} \min\limits_{x,z,e,f} \quad \sum\limits_{i=1}^n e_i + \sum\limits_{i=1}^n f_i & & \\ Ax & \geq & b \\ \sum\limits_{i=1}^n e_i + \sum\limits_{i=1}^n f_i & = & (x-\bar{x})(C^\top \sigma_{\bar{x}} + \gamma \vartheta_{\bar{x}}) \\ (x-\bar{x})(C^\top \sigma_{\bar{x}} + \gamma \vartheta_{\bar{x}}) & \geq & \varepsilon \\ -1 \leq C^\top \sigma_{\bar{x}} + \gamma \vartheta_{\bar{x}} & \leq & 1 \\ \bar{x} + z + e - f & = & x \\ Cz & \leq & 0 \\ \gamma z & \leq & 0 \\ x \in \{0,1\}^n,\ z \in \mathbb{R}^n,\ e,f & \in & \mathbb{R}^n_+, \\ \sigma_{\bar{x}} \in \mathbb{R}^r_+,\ \vartheta_{\bar{x}} & \in & \mathbb{R}_+. \end{array} \right\} \quad (3)$$

Again, by integrality of x and boundedness of σ, ϑ, we can linearize Problem (3) to a MILP.

It is not difficult to generalize the cut to include the case where the lower level polyhedron $\Omega(x)$ is empty. Indeed we notice that for all x such that $Cx \geq C\bar{x}$, $\Omega(x) = \varnothing$. Hence we could apply the same framework as above to derive valid inequalities for $\bar{x}$.

4.4. Row-and-column generation algorithm for BBLP

We can now solve the BBLP in the general case using the same iterative framework as in Section 3. Assume that, at step k, we have a relaxation $\mathcal{P}^k$ of $\mathsf{conv}(\mathcal{F})$ obtained by adding new variables σ, ϑ and constraints $\mathcal{V}(\bar{x})$ to $\mathcal{F}$. Let x^k be an optimal solution of the relaxed problem

$$Q^k = \min_{x,\sigma,\vartheta} \{\alpha x \mid (x,\sigma,\vartheta) \in \mathcal{P}^k, x \in \{0,1\}^n\},$$

which generalizes the corresponding relaxed problem introduced in Section 3.2. We remark that Q^k is a relaxation of Problem (1).

The above discussion proves the following result.

Theorem 1 *For each $k \in \mathbb{N}$, the optimal solution x^k of Q^k induces a lower bound for the original bilevel problem P independently of Hypotheses 1 and 2.*

Therefore, if $x^k \in \mathcal{F}$, i.e. $f(x^k) \geq c - \gamma x^k$, we know that x^k is an optimal solution of P. Otherwise, by Proposition 4, we add a set of new variables σ, ϑ and a set of new valid inequalities $\mathcal{V}(\bar{x}^k)$ to $\mathcal{P}^k$ cutting all the binary points in $\mathcal{C}(\bar{x}^k)$, where $\bar{x}^k$ is chosen from x^k in a similar way as in Section 3.

4.5. Finding $\leq_{\mathcal{C}}$-maximal lower level infeasible points

Let $\leq_{\mathcal{C}}$ be the partial preorder relation on $\{0,1\}^n$ such that $x^1 \geq_{\mathcal{C}} x^2$ if and only if $Cx^1 \geq Cx^2$ and $\gamma x^1 \geq \gamma x^2$. In order to determine $\leq_{\mathcal{C}}$-maximal elements of $\bar{\mathcal{F}}$, we proceed as in Section 3. From an infeasible element $\bar{x} \in \bar{\mathcal{F}}$ we find another element $x' \in \bar{\mathcal{F}}$ for which $\mathcal{C}(x')$ is largest (so as to cut the largest number of infeasible points); or, in other words, a $\mathcal{C}$-maximal element $x' \in \bar{\mathcal{F}}$ which dominates $\bar{x}$. Analogously to $M(S)$, for any set $S \subset \{0,1\}^n$, we define $M_{\mathcal{C}}(S)$ as the set of maximal elements in S under $\leq_{\mathcal{C}}$, in view of considering $\leq_{\mathcal{C}}$-maximal upper level points infeasible w.r.t. the lower level.

Let us consider the following optimization problem :

$$\mathsf{FM}_C = \left\{ \begin{array}{rrcl} \max\limits_{x,y,e,f} & \sum\limits_{i=1}^{m} e_i + f & & \\ & \beta\, y + \gamma\, x & \leq & c - \varepsilon \\ & Cx - e & = & C\bar{x} \\ & \gamma\, x - f & = & \gamma\, \bar{x} \\ & Ax & \geq & b \\ & e, f & \geq & 0 \\ & x & \in & \{0,1\}^n \\ & y & \in & \Omega(x) \cap \mathcal{Y} \end{array} \right.$$

for $\varepsilon > 0$ small enough.

As in the previous section there exists $\varepsilon > 0$ small enough such that if x satisfies $Ax \geq b$, but is infeasible in FM, then $x \in \mathcal{F}$. We claim that an optimal solution x^* of FM_C belongs to $M_C(\bar{\mathcal{F}})$. Notice first that by the second and third constraints of FM_C, we have that $x^* \geq_C \bar{x}$. Furthermore, since there exists $y \in \Omega(x^*)$ such that $\beta\, y + \gamma\, x^* < c$, we have that $f(x^*) < c - \gamma\, x^*$, i.e. $x^* \in \bar{\mathcal{F}}$. If $x^* \notin M_C(\bar{\mathcal{F}})$, there must exist $x' \in \bar{\mathcal{F}}$ and (e', f') such that

$$\sum_{i=1}^{m} e'_i + f' > \sum_{i=1}^{m} e^*_i + f^*,$$

which implies by the optimality of x^* that $\beta\, y + \gamma\, x' > c - \varepsilon$ for all y. Hence by definition of ε, we have that $x' \in \mathcal{F}$, contradiction.

5. The power edge set problem

Let $G = (V, E)$ be a graph modelling an electrical network where $V = \{1, \ldots, n\}$ is the set of nodes representing the (sub)stations and E the set of edges corresponding to transmission lines. For $i \in V$, $N(i) = \{j \mid \{i, j\} \in E\}$ is the set of neighbours (adjacent nodes) of i. For graph-theoretical notions, see [12, 30]. PMUs are physically placed on edges close to one of the adjacent nodes. From a modelling point of view, the fact that PMUs are placed close to a node is irrelevant. We shall therefore simply assume that placements occur on edges. A graph is said to be observable if all node voltages and current edges are

known either measured by a PMU or estimated using electrical laws. We denote by Ω the set of observed nodes: the ambiguity with the lower level polyhedron $\Omega(x)$ of $(*)$ is only apparent, since it will turn out that the lower level problem will yield the nodes which can be observed from a given set of placed PMUs.

The problem is formally defined as follows.

> Power Edge Set problem. Given a graph $G = (V, E)$, find a subset $\Pi \subseteq E$ of minimum cardinality such that, if PMUs are placed on edges in E, the graph G is fully observable.

It is important to remark that observability is defined using the two rules below, which encode Ohm's and Kirchoff's laws.

$R1$: If a PMU is placed on an edge $\{i, j\}$, then nodes i and j are observed, i.e. $\{i, j\} \in \Pi$ implies $i, j \in \Omega$.

$R2$: If all the neighbors of an observed node i are observed but one, then this node is also observed, i.e. $i \in \Omega \wedge |N(i) \setminus \Omega| \leq 1$ implies $N(i) \subseteq \Omega$.

By rule $R1$ and Ohm's law, the PMU placed at $\{i, j\}$ measures the voltage at i and j and the current on $\{i, j\}$ (so i and j are observed). By rule $R2$, if a node i and all its neighbors $k \in N(i)$ are observed, except a single node j, then using Ohm's law we can determine the current on $\{i, k\}$ for $k \in N(i) \setminus \{j\}$; knowing the currents on all $\{i, k\}$ (for $k \neq j$) we can deduce the current on $\{i, j\}$ using Kirchoff's law. Then, knowing the voltage at i and the current on $\{i, j\}$, we determine the voltage on j using Ohm's law. Hence, j is observed. More details can be found in [28].

We introduce two sets of binary decision variables: x defining Π, and y defining Ω, as follows.

- For each $i \in V$ and $j \in N(i)$, let $x_{ij} = 1$ iff a PMU is placed on edge $\{i, j\}$, i.e. if $\{i, j\} \in \Pi$.
- For each node $i \in V$ let $y_i = 1$ iff i is observed, i.e. $i \in \Omega$.

We remark that, to ease notation, there is some redundancy in the x variables: more precisely, we define both x_{ij} and x_{ji} for the same (undirected) edge $\{i, j\} \in E$. Since we want to minimize the number of installed PMUs, we will let the minimization direction assign the value 1 to at most one variable in x_{ij}, x_{ji} at the optimum.

Note that while $R1$ gives a direct dependency between x and y, which can be written as

$$\forall i \in V, j \in N(i) \quad y_i \geq x_{ij} + x_{ji} \tag{4}$$

$R2$ defines a "dynamic" relationship between x and y variables: we can only know if $j \in V$ is observed if we previously knew that some adjacent node i and all of its neighbours but j were observed. Moreover, once $i \in \Omega$, i "stays" in Ω, which can be interpreted as saying that, if $R2$ is used iteratively to construct Ω, the set Ω can only increase its cardinality. This monotonicity makes it possible to define an "observability propagation" function of which Ω represents the fixed point, which turns out to have the following properties [28]:

1. it is a minimum fixed point, namely $|\Omega| = \sum_{i \in V} y_i$ is minimum;

2. for each $i \in V, j \in N(i)$, the following fixed point invariant holds:

$$y_i - y_j \geq \sum_{\substack{k \in N(j) \\ k \neq i}} y_k + 1 - |N(j)|. \tag{5}$$

This allows us to define Ω for a given Π by means of a BLP which minimizes $|\Omega|$ subject to Eq. (4)-(5).

The PES can therefore be formulated as a BBLP, as follows:

$$\left.\begin{array}{l} \min_x \sum_{i \in V} \sum_{j \in N(i)} x_{ij} \\ \forall i \in V, j \in N(i) \quad x_{ij} \in \{0,1\} \\ |V| \leq |\Omega(x)| = \left\{ \begin{array}{llll} \min\limits_{y \in \{0,1\}^{|V|}} & \sum\limits_{i \in V} y_i & & \\ \forall i \in V, j \in N(i) & y_i & \geq & x_{ij} + x_{ji} \\ \forall i \in V, j \in N(i) & y_i - y_j & \geq & \sum\limits_{\substack{k \in N(j) \\ k \neq i}} y_k + 1 - |N(j)|. \end{array} \right. \end{array}\right\} \tag{6}$$

Proposition 5 *The BBLP* (6) *satisfies Hypotheses 1 and 2.*

Proof. Hypothesis 1 requires that, given feasible upper level vectors x, x', if $x \leq x'$ then the feasible region of the lower level problem in function of x contains the corresponding set in function of x'. This is readily seen using Eq. (4): for any $\{i, j\} \in E$ such that $x_{ij} + x_{ji} = 0$, the corresponding y variable on the LHS can be set at either 0 or 1. On the other hand, if $x'_{ij} + x'_{ji} = 1$, then the LHS can only take value 1, which proves the claim. Hypothesis 2 is trivially verified since $\gamma = 0$ in Eq. (6). □

By Prop. 5, Eq. (6) is a *Restricted*-BBLP, so we can use the cut generation algorithm as detailed in Section 3.

6. The minimum zero forcing set problem

In the MZFS problem, the color change rule is applied until a stable configuration is reached (see Section 1.2.2). We exploit the monotonicity of the color change rule dynamics to derive a fixed-point condition, which we describe by means of a BLP. We can then formulate the MZFS problem by means of a BBLP.

Let $Z \subseteq V$ be a set of vertices of G colored in black and let us define $S^0 = Z$. For each $t = 1, \ldots, |V|$, we define the set $S^t \subseteq V$ as the set of vertices $u \in V$ that are either in S^{t-1} or such that their color can be changed from white to black by an application of the color-change rule to a vertex $v \in S^{t-1}$. Notice that the derived coloring can be deduced from $S^{|V|}$.

Let $x \in \{0,1\}^{|V|}$ be the binary variable corresponding to the characteristic vector of Z (i.e. $x_v = 1 \Leftrightarrow v \in Z$), and let $f(x)$ denote the size of the black vertices in G in the derived coloring. The MZFS problem can be modeled as the following program:

$$\left.\begin{array}{llll} \min\limits_{x \in \{0,1\}^{|V|}} & \sum\limits_{i \in V} x_i & & \\ & f(x) & \geq & |V|. \end{array}\right\}$$

We now derive a fixed point condition similarly to Section 5 above. Given a characteristic vector $x \in \{0,1\}^{|V|}$, we introduce the function

$$\theta^x : \{0,1\}^{|V|} \mapsto \{0,1\}^{|V|}$$

given by

$$\forall v \in V, \theta^x_v(y) = \max\left(x_v, y_v, \max_{u \in N(v)}\left(1 - |N(u)| + y_u + \sum_{\substack{v' \in N(u) \\ v' \neq v}} y_{v'}\right)\right).$$

We aim to show that the stable configuration reached by a repeated application of the color change rule to the configuration x is the minimum fixed point of θ^x.

For all $t = 1, \ldots, |V|$, let $y^t \in \{0,1\}^{|V|}$ denotes the characteristic vector of the set S^t. We have the following lemma.

Lemma 4 *For all $t = 1, \ldots, |V|$ we have $y^t = \theta^x(y^{t-1})$ where $y^0 = x$.*

Proof. Let $v \in V$. By definition, $\theta^x_v(y^{t-1}) = \max(x_v, y^{t-1}_v, Q)$ where $Q = \max_{u \in N(v)}(1 - |N(u)| + y^{t-1}_u + \sum_{\substack{v' \in N(u) \\ v' \neq v}} y^{t-1}_{v'})$. We have $Q \leq 1$, hence $\theta^x_v(y^{t-1}) \in \{0,1\}$.

- If $x_v = 1$ or $y^{t-1}_v = 1$, then $v \in S^{t-1}$ and hence $v \in S^t$. Therefore $y^t_v = \theta^x_v(y^{t-1}) = 1$.
- Assume now that $x_v = y^{t-1}_v = 0$. We prove that $Q = 1$ if and only if the color of v can be changed from white to black using a vertex $u \in N(v)$. Indeed, $Q = 1$ if and only if there exists a vertex $u \in N(v)$ such that $y^{t-1}_u = 1$ and $y^{t-1}_{v'} = 1$ for all $v' \in N(u)$ with $v' \neq v$. Such a case happens when v has a neighbor u colored in black with all its other neighbors $v' \in N(u) \setminus \{v\}$ colored in black. Then $v \in S^t$. Therefore the function θ^x models the *color-change rule* of G when the initial vertices colored in black are modeled by x.

Therefore, $y^t = \theta^x(y^{t-1})$. □

Let $y = y^{|V|}$, i.e. y models the final colors of the vertices in the derived coloring, i.e. 1 if black, 0 otherwise. From Lemma 4, we deduce that $y = \theta^x(y)$. Hence y is a fixed point of θ^x. Furthermore y is a smallest fixed point of θ^x since if $\exists\, y' < y$ such that $y' = \theta^x(y')$ then the color-change rule cannot be applied further from the set S' characterized by y'. Hence $|S'| < |S|$ which is a contradiction.

From these observations, it follows that the value of $f(x)$ can be obtained by solving the following BLP:

$$f(x) = \left\{ \begin{array}{lrcl} \min\limits_{y \in \{0,1\}^{|V|}} & \sum\limits_{i \in V} y_i & & \\ & y & \geq & x \\ \forall v \in V, u \in N(v) & y_v - y_u & \geq & \sum\limits_{\substack{v' \in N(u) \\ v' \neq v}} y_{v'} + 1 - |N(u)|. \end{array} \right.$$

Notice furthermore that Hypotheses 1 and 2 are satisfied, similarly to Prop. 5.

7. Computational results

The purpose of this section is to test two variants of a cutting plane algorithm for solving instances of Problem (†), namely BBLPs having binary variables only in the upper level subproblem, a row $\beta\, y + \gamma\, x \geq c$ in the upper level subproblem, and a lower level subproblem having $\beta\, y$ as the objective function to be minimized w.r.t the lower level variables y. We do present comparative results with another bilevel MILP solver (see Section 7.5 below).

The first variant, presented in Section 3, addresses *Restricted*-BBLP instances (i.e. instances satisfying Hypotheses 1 and 2). The second variant, presented in Section 4, is independent of these hypotheses.

Both variants are iterative in nature (as all cutting plane algorithms are). The upper level subproblem is used as a "master problem", producing solutions which may violate the lower level subproblem constraint $\beta\, y \geq c - \gamma\, x$. At each iteration the lower level subproblem solution is exploited to derive a cut, which is then added to the master problem. Termination occurs where the master problem produces a solution which satisfies the lower level subproblem constraint.

The difference between the two variants is in how they derive the cut at each iteration. The *Restricted*-BBLP variant can exploit more structure, and constructs a cut at the expense of solving an auxiliary cut generation MILP. The second variant needs to find an appropriate threshold prior to solving the cut generation MILP, and valid thresholds can be found by solving a further MILP. From the **NP**-hardness point of view, BBLP is as hard as MILP, so these solution strategies appear to make little sense. From a practical point of view, as we shall see, our algorithms are able to solve bilevel MILP instances of considerable size.

7.1. Test set

Our test set consists of PES instances (Section 5) and MZFS instances (Section 6) over standard IEEE networks, as well as instances generated randomly in the following way:

- generate appropriately sized matrices A, B, C with uniformly chosen integers in $\{-100, \ldots, 100\}$
- generate a random binary vector x^0 and let $b = A\, x^0$ (so the upper level constraints $Ax \geq b$ are feasible)
- generate appropriately sized vectors d, α, γ, again with random integers in $\{-100, \ldots, 100\}$
- generate β with random integers in $\{0, \ldots, 100\}$
- generate a random scalar c in $\{50, \ldots, 150\}$.

The decision variable types are generated in the most general possible way for our code to work, i.e. the upper level variables are binary, and the lower level can be of any type (binary, general integer, continuous). Specifically, lower level variables may not have explicit lower and upper bound constraints. Note that the lower level problem can obviously be infeasible, as no effort is made towards feasibility. The upper level might also be infeasible because of the constraint

$\gamma x + \beta y \geq c$. We constructed two sets of these random instances: with $C, \gamma \geq 0$ to ensure that Hypothesis 1 is satisfied (prefixed by `hyp` in Table 1), and without this further condition (prefixed by `rnd` in Table 1).

Our instance collection is described in Table 1, organized in six columns giving a description summary: name number of upper level, lower level and total binary variables followed by upper level and lower level constraints, and a flag indicating whether the instance is a *Restricted*-BBLP ("Y") or a general one ("N"). The cardinality symbols n, q, m, p (number of upper/lower level variables, number of upper level constraints) are mentioned in Problem $(*)$, with the provision that $p = 1$ as per Problem (†); the symbol r (number of lower level constraints) is mentioned in Eq. (2). PES instance names are prefixed by `pmu`, MZFS by `0fc`, random *Restricted*-BBLP instances by `hyp` and general random BBLP instances by `rnd`. We remark that the numbers appearing in the names of the randomly generated instances can be ignored, as they are concerned with the input of our instance generator.

7.2. Test platform

The tests have been run on a mid-2015 MacBook Pro with 16GB RAM and a 3.1GHz i7 dual core CPU, virtually configured as a quad-core CPU, with 16GB RAM. The MILP subsolver called on the auxiliary subproblem is CPLEX 12.6.2 [15]. The coding language is Julia [2], using the JuMP [18] module with the CPLEX API.

7.3. Measuring CPU time

When solving large instances, most of the CPU time is spent solving the MILP subproblems using the CPLEX solver, which is configured to use as many threads as the number of virtual cores (i.e. four). Our Julia code makes no explicit use of parallel computation. Moreover, all Julia code is Just-in-Time (JiT) compiled, which means that the first iteration of each run takes considerably more time to complete since it needs to compile the "macro" parts of the code (which vary with the instance).

For these reasons, every attempt to a meaningful "user CPU time" evaluation is prone to criticisms: do we count JiT compilation time or not? How would one define the user CPU time for parallel codes, since interprocess communication, considered system (rather than user) time by the OS, is a necessary part of the code run? These are all interesting questions that attract a lot of attention in the language design community. In these tests we chose to measure "wall clock time", which is the actual number of seconds elapsed to complete the run. We were careful to avoid running other processes at the same time on the test machine, so as to not invalidate our time measures, but of course some of the service/daemon software might have also used a hopefully irrelevant fraction of the clock cycles.

Auxiliarily, the use of the Julia language penalizes our CPU measures in two ways. On the one hand, the JiT compilation time appears to take the vast majority of the total time for small instances. On the other, the JuMP/CPLEX API appears to be slower than having the CPLEX command line tool. We believe these are acceptable trade-offs for the flexibility of the Julia language, but it must be kept in mind that our results could be vastly improved for small instances, and marginally improved for large instances.

Name	n	q	$\#bin$	$m+p$	r	*Restricted?*
pmu_IEEE_5	12	5	17	14	24	Y
pmu_IEEE_7	16	7	23	18	32	Y
pmu_IEEE_14	40	14	54	42	80	Y
pmu_IEEE_24	68	24	92	70	136	Y
pmu_IEEE_30	82	30	112	84	164	Y
pmu_IEEE_39	92	39	131	94	184	Y
pmu_IEEE_57	162	57	219	164	324	Y
pmu_IEEE_118	358	118	476	360	716	Y
0fc_IEEE_5	5	5	10	1	17	Y
0fc_IEEE_7	7	7	14	1	23	Y
0fc_IEEE_14	14	14	28	1	54	Y
0fc_IEEE_24	24	24	48	1	92	Y
0fc_IEEE_30	30	30	60	1	112	Y
0fc_IEEE_39	39	39	78	1	131	Y
0fc_IEEE_57	57	57	114	1	219	Y
0fc_IEEE_118	118	118	236	1	476	Y
hyp_5_20_15_30_30	20	60	50	6	15	Y
hyp_10_30_2_20_10	30	30	50	11	2	Y
hyp_5_30_10_20_20	30	40	50	6	10	Y
hyp_10_35_15_10_10	35	20	45	11	15	Y
hyp_5_35_2_0_30	35	30	35	6	2	Y
hyp_10_35_15_25_10	35	35	60	11	15	Y
hyp_5_40_2_10_10	40	20	50	6	2	Y
hyp_10_40_5_15_15	40	30	55	11	5	Y
hyp_20_40_10_30_10	40	40	70	21	10	Y
hyp_10_50_10_50_0	50	50	100	11	10	Y
rnd_2_10_2_10_10	10	20	20	3	2	N
rnd_5_10_5_10_10	10	20	20	6	5	N
rnd_1_10_6_20_10	10	30	30	2	6	N
rnd_5_10_4_20_10	10	30	30	6	4	N
rnd_2_20_8_20_10	20	30	40	3	8	N
rnd_0_20_10_20_20	20	40	40	1	10	N
rnd_4_20_8_40_0	20	40	60	5	8	N
rnd_15_25_2_30_5	25	35	55	16	2	N
rnd_15_25_10_10_20	25	30	35	16	10	N
rnd_10_30_2_0_30	30	30	30	11	2	N

Table 1: Instance statistics.

7.4. Results

We have two sets of results. In Table 2, we showcase the performance of the two code variants (*Restricted*-BBLP and general BBLP) on instances which are actually *Restricted*-BBLPs. This gives us an indication about the "cost of generality" in terms of CPU time. In Table 3, we only report results from solving general BBLP instances.

In Table 2, each row of which corresponds to the performances relative to a named instance. The first three columns refer to the *Restricted*-BBLP code

variant, and the remaining three to the general variant. For each instance and variant, we report the optimal (upper level) objective function value, the number of cutting plane algorithm iterations, and the wall clock CPU time. As

	Restricted			General		
Name	*obj*	*#itn*	*CPU*	*obj*	*#itn*	*CPU*
pmu_IEEE_5	1.0	0	5.11	1.0	0	4.70
pmu_IEEE_7	2.0	4	6.10	2.0	4	12.40
pmu_IEEE_14	2.0	6	7.77	2.0	3	13.26
pmu_IEEE_24	3.0	7	6.49	3.0	4	17.51
pmu_IEEE_30	5.0	12	6.86	5.0	13	820.96
pmu_IEEE_39	6.0	12	6.43	6.0	9	193.86
pmu_IEEE_57	6.0	24	8.74	–	*19*	*14832*
pmu_IEEE_118	18.0	216	50.33	–	–	–
0fc_IEEE_5	2.0	4	6.31	2.0	5	13.09
0fc_IEEE_7	2.0	5	5.96	2.0	6	12.43
0fc_IEEE_14	4.0	18	6.45	4.0	17	20.68
0fc_IEEE_24	6.0	15	6.25	6.0	16	38.21
0fc_IEEE_30	7.0	43	7.41	7.0	44	3117.47
0fc_IEEE_39	7.0	28	7.24	7.0	22	289.16
0fc_IEEE_57	9.0	248	47.28	–	*248*	*20721*
0fc_IEEE_118	–	*1251*	*83120*	–	–	–
hyp_20_40_10_30_10	-267.0	3	7.59	-267.0	2	15.27
hyp_10_35_15_25_10	-354.0	14	6.93	-354.0	6	30.97
hyp_10_30_2_20_10	-452.0	31	8.52	-452.0	5	23.41
hyp_5_20_15_30_30	-214.0	33	7.29	*	*	*
hyp_5_40_2_10_10	-531.0	68	9.48	-531.0	3	30.10
hyp_5_35_2_0_30	-96.0	79	14.28	-96.0	2	34.19
hyp_10_35_15_10_10	-707.0	79	14.77	*	*	*
hyp_5_30_10_20_20	-376.0	114	11.86	*	*	*
hyp_10_50_10_50_0	-797.0	193	68.94	*	*	*
hyp_10_40_5_15_15	–	*1067*	*7482*	*	*	*

Table 2: Computational results comparing *Restricted* and general BBLP code variants. Some of the largest instances (those with statistics typeset in italics) remain unsolved at termination, enforced by the operating system for resource exhaustion. In the instances marked with *, CPLEX yielded errors when solving the auxiliary subproblem $D_{\bar{x}}(\varepsilon)$ (see p. 10), causing a default value $\varepsilon = 0.001$ to be used, and hence very shallow cuts, yielding in turn a very slow convergence.

is apparent from Table 2, bypassing Hypotheses 1 and 2 adversely impacts the capacity of the code to scale.

Table 3 is not comparative: it simply validates the claim that our general method is able to find solutions for some problems not satisfying Hypotheses 1 and 2.

7.5. Comparative results

To the best of our knowledge, only one software package for solving bilevel MILPs is actually available: MibS [7]. Very recently, in a recently published

Name	*obj*	*#itn*	*CPU*
rnd_10_30_2_0_30	-251.0	3	26.86
rnd_1_10_6_20_10	3.0	7	15.58
rnd_15_25_2_30_5	-325.0	8	15.39
rnd_2_10_2_10_10	52.0	9	12.25
rnd_0_20_10_20_20	-475.0	9	86.74
rnd_5_10_5_10_10	73.0	12	14.76
rnd_2_20_8_20_10	-546.0	15	36.69
rnd_5_10_4_20_10	-151.0	16	15.27
rnd_4_20_8_40_0	-599.0	38	181.05
rnd_15_25_10_10_20	∞	51	2737.39

Table 3: Computational results on the general BBLP code variant on non-restricted random instances; ∞ marks an infeasible instance.

paper [10], another technique is described for solving bilevel MILPs, with corresponding computational results.

MibS is limited to integer variables only (in both levels) and integer input data, whereas our code is constrained to take binary variables in the upper level, but whatever type of variable in the lower level, and whatever type of rational data. We used MibS on our test-set, and managed to obtain solution statistics for the application-related instances (PES and zero forcing). MibS, however, failed on all our randomly generated instances — either because the instance was not successfully read, or because it was wrongly deemed infeasible. See Table 4 for more details. It is evident from Table 4 that our method outperforms MibS on our test set, other than on very small instances (on which our language choice, Julia, penalizes us because of the JiT compilation). Some interaction with one of the authors of MibS led us to believe that a new release, which might change this situation, is imminent.

The method described in [10] is more general than ours in a few respects: it accepts any number of upper level constraints involving variables from both levels (instead of just one); the lower level objective function is independent from the upper level constraints (ours is not); the upper level objective function can involve lower level variables (ours does not); and the upper level variables need not be binary (ours do). The code relative to [10], however, was not publically available at the time of writing this paper. The results obtained in [10] relate to a different hardware and software platform than ours. In short, even though we exchanged a few instances with the authors of [10], no meaningful comparison has been possible.

8. Conclusions

We proposed a new practically useable cutting plane algorithm for solving a large class of bilevel MILPs. With respect to the most general class of bilevel MILPs, we impose the following restrictions: the upper level objective function only involves upper level variables x; there is exactly one constraint $\gamma x + \beta y \geq c$ in the upper level involving variables x, y from both levels; and the objective function of the lower levels is precisely $\min_y \beta y$. We modelled two applications,

	CutGen		MibS	
Name	*obj*	*CPU*	*obj*	*CPU*
pmu_IEEE_5	1.0	5.11	1.0	0.00
pmu_IEEE_7	2.0	6.10	2.0	0.01
pmu_IEEE_14	2.0	7.77	2.0	0.26
pmu_IEEE_24	3.0	6.49	3.0	1.87
pmu_IEEE_30	5.0	6.86	5.0	1315.79
pmu_IEEE_39	6.0	6.43	–	*3625.08*
pmu_IEEE_57	6.0	8.74	–	*3822.31*
pmu_IEEE_118	18.0	50.33	–	*7204.92*
0fc_IEEE_5	2.0	6.31	2.0	0.00
0fc_IEEE_7	2.0	5.96	2.0	0.01
0fc_IEEE_14	4.0	6.45	4.0	0.50
0fc_IEEE_24	6.0	6.25	6.0	125.26
0fc_IEEE_30	7.0	7.41	7.0	3632.64
0fc_IEEE_39	7.0	7.24	–	*3632.64*
0fc_IEEE_57	9.0	47.28	–	*3630.64*
0fc_IEEE_118	–	*83120*	–	*3148.00*

Table 4: Comparative results between our cut generation algorithm (version for *Restricted*-BBLP) and the MibS solver. Statistics in italics indicate termination for resource exhaustion or time limit.

one about measuring the state of smart grids, and the other about computing the rank of a graph, as bilevel problems of the given form, and tested two variants of a cut generation algorithms. We then tested these two algorithms on a benchmark including instances from both applications as well as randomly generated and compared our approach against MibS, an open-source solver for bilevel programming. Our cutting plane methods appear to be more efficient on medium to large instances.

Acknowledgements

Financial support from the French ADEME agency under the SO-grid project is gratefully acknowledged. The third and fourth authors would also like to acknowledge: (i) financial support from the Fondation Mathématique Jacques Hadamard (FMJH) under the Gaspard Monge Program (PGMO) in optimization and operations research; (ii) EDF for supporting the PGMO program; (iii) financial support from the European Union Grant FP7-PEOPLE-2012-ITN No. 316647 "Mixed-Integer Nonlinear Optimization".

References

[1] C. Audet, G. Savard, and W. Zghal. New Branch-and-Cut Algorithm for Bilevel Linear Programming. *Journal of Optimization Theory and Applications*, 134(2):353–370, 2007.

[2] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: A Fresh Approach to Numerical Computing. Technical Report 1411.1607, arXiv, 2014.

[3] D. J. Brueni and L. S. Heath. The PMU placement problem. *SIAM Journal on Discrete Mathematics*, 19(3):744–761, 2005.

[4] A. Caprara, M. Carvalho, A. Lodi, and G. Woeginger. A study on the computational complexity of the bilevel knapsack problem. *SIAM Journal on Optimization*, 24(2):823–838, 2014.

[5] A. Caprara, M. Carvalho, A. Lodi, and G. Woeginger. Bilevel knapsack with interdiction constraints. *INFORMS Journal on Computing*, 28(2):319–333, 2016.

[6] B. Colson, P. Marcotte, and G. Savard. Bilevel programming: A survey. *4OR*, 3(2):87–107, 2005.

[7] S.T. DeNegre and T.K. Ralphs. A Branch-and-cut Algorithm for Integer Bilevel Linear Programs. In J. W. Chinneck et al., editor, *Operations Research and Cyber-Infrastructure*, pages 65–78, Boston, MA, 2009. Springer US.

[8] X. Deng. Complexity issues in bilevel linear programming. *Nonconvex Optimization and Its Application*, 20:149–164, 1998.

[9] M. Dorfling and M. A. Henning. A note on power domination in grid graphs. *Discrete Applied Mathematics*, 154(6):1023–1027, 2006.

[10] M. Fischetti, I. Ljubič, M. Monaci, and M. Sinnl. Intersection cuts for bilevel optimization. In Q. Louveaux and M. Skutella, editors, *Integer Programming and Combinatorial Optimization: 18th International Conference, IPCO*, volume 9682 of *Lecture Notes in Computer Science*, pages 77–88, Cham, 2016. Springer.

[11] R. Fortet. Applications de l'algèbre de Boole en recherche opérationelle. *Revue Française de Recherche Opérationelle*, 4, 1960.

[12] J. L. Gross and J. Yellen. *Handbook of graph theory*. Taylor and Francis, Abingdon, 2003.

[13] T. W. Haynes, S. M. Hedetniemi, S. T. Hedetniemi, and M. A. Henning. Domination in graphs applied to electric power networks. *SIAM Journal on Discrete Mathematics*, 15(4):519–529, 2002.

[14] L. Hogben, M. Huynh, N. Kingsley, S. Meyer, S. Walker, and M. Young. Propagation time for zero forcing on a graph. *Discrete Applied Mathematics*, 160(13):1994 – 2005, 2012.

[15] IBM. *ILOG CPLEX 12.6 User's Manual*. IBM, 2014.

[16] E. Israeli and R.K. Wood. Shortest-path network interdiction. *Networks*, 40(2):97–111, 2002.

[17] A. Lodi, T. K. Ralphs, and G. J. Woeginger. Bilevel Programming and the Separation Problem. *Mathematical Programming*, 148:437–458, 2014.

[18] M. Lubin and I. Dunning. Computing in operations research using Julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.

[19] N. M. Manousakis, G. N. Korre, and P. S. Georgilakis. Optimal placement of phasor measurement units: A literature review. In *16th International Conference on Intelligent System Application to Power Systems (ISAP)*, pages 416–421, Piscataway, 2011. IEEE.

[20] N. M. Manousakis, G. N. Korre, and P. S. Georgilakis. Taxonomy of PMU placement methodologies. *IEEE Transactions on Power Systems*, 27(2):1070–1077, 2012.

[21] A.G. Mersha and S. Dempe. Linear bilevel programming with upper level constraints depending on the lower level solution. *Applied Mathematics and Computation*, 180(1):247 – 254, 2006.

[22] AIM Minimum Rank-Special Graphs Work Group. Zero forcing sets and the minimum rank of graphs. *Linear Algebra and its Applications*, 428(7):1628–1648, 2008.

[23] J.T. Moore and J.F. Bard. The mixed integer linear bilevel programming problem. *Operations Research*, 38(5):911 – 921, 1990.

[24] F.M. Pajouh and V. Boginski. Minimum vertex blocker clique problem. *Networks*, 64(1):48 – 64, 2014.

[25] P.-L. Poirion, S. Toubaline, C. D'Ambrosio, and L. Liberti. Bilevel mixed-integer linear programs and the zero forcing set. Technical Report 5210, Optimization Online, 2015.

[26] P.-L. Poirion, S. Toubaline, C. D'Ambrosio, and L. Liberti. The power edge set problem. *Networks*, 68(2):104–120, 2016.

[27] S. Toubaline, C. D'Ambrosio, L. Liberti, P. L. Poirion, B. Schieber, and H. Schachnai. Complexité du Power Edge Set problem. In *Proceedings of the Annual Meeting*, Paris, 2016. ROADEF.

[28] S. Toubaline, P. L. Poirion, C. D'Ambrosio, and L. Liberti. Observing the state of a smart grid using bilevel programming. In Z. Lu, D. Kim, W. Wu, W. Li, and D.-Z. Du, editors, *Combinatorial Optimization and Applications*, volume 9486 of *Lecture Notes in Computer Science*, pages 364–376, New York, 2015. Springer.

[29] L. Vicente, G. Savard, and J. Judice. Discrete linear bilevel programming problem. *Journal of optimization theory and applications*, 89(3):597–614, 1996.

[30] West, D.B. *Introduction to Graph Theory*. Prentice Hall, Upper Saddle River, 2000.

[31] B. Zeng and Y. An. Solving Bilevel Mixed Integer Program by Reformulations and Decomposition. Technical Report 4455, Optimization Online, 2014.