Orbitopal fixing for the full (sub)-orbitope and application to the Unit Commitment Problem

Pascale Bendotti, Pierre Fouilhoux, Cécile Rottner

Orbitopal fixing for the full orbitope and application to the Unit Commitment Problem

Pascale Bendotti*[1,2], Pierre Fouilhoux†[2], and Cécile Rottner‡[1,2]

[1]EDF R&D, 7 Boulevard Gaspard Monge, 91120 Palaiseau, France
[2] Sorbonne Université, Université Pierre et Marie Curie, LIP6, 4 Place Jussieu, 75005 Paris, France

October, 2017

Abstract

It is common knowledge that symmetries arising in integer programs could impair the solution process, in particular when symmetric solutions lead to an excessively large branch and bound (B&B) search tree. Techniques like isomorphic pruning [11], orbital branching [16] and orbitopal fixing [8] have been shown to be essential to solve very symmetric instances from the literature. This paper focuses on formulations involving a set of 2-index variables, also referred to as matrix, such that the corresponding symmetry group is the set of all column permutations. Such formulations arise for example from scheduling problems with a discrete time horizon. Orbitopal fixing as introduced in [8] is restricted to the special case of partitioning (resp. packing) formulations involving a solution matrix with exactly (resp. at most) one 1-entry in each row. It relies on the linear description of the partitioning (resp. packing) orbitope [9], i.e., the convex hull of binary matrices with lexicographically non-increasing columns and exactly (resp. at most) one 1-entry per row. The main result of this paper is to extend orbitopal fixing to the full orbitope, namely with no restriction on the number of ones in each row. We propose a linear time orbitopal fixing algorithm for the full orbitope, referred to as the static version, as it is defined for the natural lexicographical order. We also introduce a dynamic version of this algorithm where the lexicographical order follows the branching decisions occurring along the B&B process. Experimental results for the Unit Commitment Problem are presented. A comparison with state of the art techniques like modified orbital branching [14] is also considered to show the effectiveness of the proposed algorithms.

*pascale.bendotti@edf.fr
†pierre.fouilhoux@lip6.fr
‡cecile.rottner@edf.fr

1 State of the art

1.1 Definitions

Consider an ILP of the form :

$$(ILP) \quad \min \left\{ cx \mid x \in P \right\}$$

where A is an $m \times n$ matrix and $P = \left\{ x \in \{0,1\}^n \mid Ax \geq b \right\}$. Let Π^n be the set of all permutations of the indices $I^n = \{1, ..., n\}$.

The symmetry group $\mathcal{G}$ of (ILP) is the set of all permutations π mapping each feasible solution to a feasible solution of same value, *i.e.*:

$$\mathcal{G} = \left\{ \pi \in \Pi^n \mid \forall x \in P,\ cx = c\pi(x) \text{ and } \pi(x) \in P \right\}$$

For instance, consider the problem presented as Example 1:

Example 1.

$$\min \left\{ x_1 + x_2 + 2(x_3 + x_4 + x_5) \ \ s.\ t.\ \ 3(x_1 + x_2) + (x_3 + x_4) + 3x_5 = 4 \text{ and } x \in \{0,1\}^5 \right\} \quad \textit{(Ex1)}$$

The symmetry group $\mathcal{G}_1$ of this problem contains $\{id,\ \pi_{1,2},\ \pi_{3,4}\}$, where id is the identity permutation, $\pi_{i,j}$ is the transposition of variables i and j.

Subset $S \subset I^n$ and its characteristic vector will be used interchangeably in the following.

The *orbit* of S under $\mathcal{G}$ is defined as the set of all sets S' symmetric to S under $\mathcal{G}$:

$$orb(S, \mathcal{G}) = \{S' \subset I^n \mid S' = \pi(S),\ \pi \in \mathcal{G}\}$$

Referring to Example 1, let $S = [1,\ 1,\ 1,\ 0,\ 0]$. Then $orb(S, \mathcal{G}_1)$ contains $\{[1,\ 1,\ 1,\ 0,\ 0], [1,\ 1,\ 0,\ 1,\ 0]\}$, since $\pi_{3,4}(S) = [1,\ 1,\ 0,\ 1,\ 0]$ and $\pi_{1,2}(S) = S$.

Subset $R \subset I^n$ is said to be *lexicographically larger than* subset $T \subset I^n$ if there exists $j \in \{1, ..., n-1\}$ such that:

- $\forall i \leq j,\ R[i] = T[i]$
- $R[j+1] > T[j+1]$

We write $T \leq R$ if R is equal to T or if R is lexicographically larger than T. Note that R is lexicographically larger than or equal to T if the binary number encoded by R (with the most significant bit on the left) is larger than or equal to the binary number encoded by T:

$$\sum_{i=1}^{n} 2^{n-i} R[i] \ \geq \ \sum_{i=1}^{n} 2^{n-i} T[i].$$

Subset $R \in orb(S, \mathcal{G})$ is said to be a *representative* among $orb(S, \mathcal{G})$ if R is lexicographically maximum among the sets in the orbit of S under $\mathcal{G}$, *i.e.*, $R \geq g(S)$, $\forall g \in \mathcal{G}$. Note that, in this case, R is also a representative among its own orbit, since $orb(S, \mathcal{G}) = orb(R, \mathcal{G})$.

For instance, referring to Example 1, subset $S = [1,\ 1,\ 1,\ 0,\ 0]$ is lexicographically maximal among its orbit, thus S is a representative.

1.2 Branch & Bound

The Branch & Bound algorithm is to enumerate candidate solutions by means of a rooted tree, the root corresponding to the full solution set. The principle of the algorithm is to split recursively the search space in smaller spaces. The algorithm explores branches of this tree, each node representing a subset of the solution set. At each node, a lower and an upper bound on the solution value is computed, and if no better solution than the one found by the algorithm so far can be produced, the node is discarded.

When implementing a Branch & Bound algorithm, one has two strategies to elaborate:

- Exploration strategy, *i.e.* in which order the branching tree is explored,
- Branching strategy, *i.e.* which disjunction is branched on at each node.

Note that whichever branching strategy is chosen, at some point in the branching tree, there will be variables whose values are fixed as a result of the preceding branching decisions taken from the root to the current node. For a given node a of the enumeration tree, F_1^a (resp. F_0^a) is defined as the set of indices of variables fixed to 1 (resp. 0) at node a. F^a is the set of indices of free variables at node a.

For each solution S, all elements in $orb(S, G)$ are enumerated in the tree, whereas the optimal value obtained would be the same if only one representative of $orb(S, G)$ were enumerated.

Referring to Example 1, a Branch & Bound algorithm would produce solutions $x = [1, 1, 1, 0, 0]$ and $x' = [1, 1, 0, 1, 0]$. Both are elements of $orb(x, \mathcal{G}_1)$ and have value 4. Solution x' could have been obtained by applying permutation $\pi_{3,4}$ to the representative solution x.

The key idea is to prune non-representative solutions from the Branch & Bound tree. This can be done, for example, by combining pruning strategies to adequate exploration and branching strategies.

A pruning strategy is said to be *flexible* if it does not constrain the exploration and branching strategies to be used in the Branch & Bound tree. For example, a pruning strategy which can be applied only if a given branching rule is used is not flexible.

Note that, as the branching process fixes variables, the symmetry group $\mathcal{G}(a)$ of the subproblem associated to node a evolves and differs from the global symmetry group $\mathcal{G}$.

Indeed, suppose at a given node a of the enumeration tree relative to problem $(Ex1)$, variable x_1 is fixed to 1 and variable x_2 is fixed to 0. Then for any feasible solution x at node a, $\pi_{1,2}(x) = [0,\ 1,\ x_3,\ x_4,\ x_5]$ which is not a feasible solution of the subproblem associated to node a. Thus, although $\pi_{1,2}$ is in the global symmetry group $\mathcal{G}_1$ of the problem, it is no longer in the symmetry group $\mathcal{G}(a)$ associated to a.

However, as reported in [16], it may be computationally prohibitive to compute the symmetry group for every node of the enumeration tree, since all known algorithms have exponential running time. Thus, an alternative possibility is to consider a subgroup, called $\mathcal{G}^a$, of the global symmetry group, defined as follows:

$$\mathcal{G}^a = \{g \in \mathcal{G} \mid g(F_1^a) = F_1^a\}.$$

Example 2 proves that at node a, the subgroup $\mathcal{G}^a$ of the global symmetry group may be different from the symmetry group of the subproblem $\mathcal{G}(a)$.

Example 2. *Problem* $\min\{cx \mid Ax \leq b,\ x \in \{0,1\}^3\}$ *whose solution set is*

$$\{[0,\ 1,\ 1],\ [1,\ 0,\ 1],\ [1,\ 1,\ 0],\ [0,\ 1,\ 0]\}.$$

Then $\mathcal{G} = \{id,\ \pi_{1,3}\}$. *Consider a node* a *of the enumeration such that* $F_1^a = \{3\}$ *and* $F_0^a = \varnothing$. *Then* $\mathcal{G}^a = stab(F_1^a, \mathcal{G}) = id$. *However, the solution set of subproblem* a *is* $\{[0,\ 1,\ 1],\ [1,\ 0,\ 1]\}$, *with symmetry group* $\{id,\ \pi_{1,2}\}$.

This example shows how fixed variables in the Branch & Bound enumeration tree can introduce new symmetries in the solution set of the subproblem considered at each node.

1.3 Pruning by isomorphism in Branch & Bound

In [11], Margot considers a general framework, where the symmetry group $\mathcal{G}$ is not restricted.

A pruning strategy called *isomorphism pruning* (ISP) is defined, such that at any node a, if F_1^a is not a representative then node a is pruned.

The branching strategy called *minimum index branching* (MIB) is defined as branching on the minimum index free variable x_i at each node, with disjunction:

$$x_i = 0 \quad \vee \quad x_i = 1$$

Minimum index branching used alongside with isomorphism pruning can ensure that only representative solutions are explored in the tree. It can be shown that the optimal value remains the same. For any set $S \subset I^n$ representative under $\mathcal{G}$, subset $S' = S \backslash \{v\}$ with $v = \max\{w \in S\}$ is also a representative. Hence, at a given node a of the Branch & Bound enumeration tree, if F_1^a is not a representative, then any solution S such that $F_1^a \subset S$ is not a representative neither, provided that rule MIB was used in the enumeration tree.

Margot introduces another operation called *0-setting*, which sets to 0 free variables that would induce a non-lexicographically maximum solution.

0-setting consists in the two following operations:

(i) Let b be a node in the enumeration tree and let x_f be the branching variable at b. If a is the son of b where x_f is fixed to 0 then set to 0 all free variables in $orb(\{f\}, \mathcal{G}^a)$.

(ii) Let $f = \min\{r \in F^a\}$. If $F_1^a \cup \{f\}$ is not a representative then set to 0 all free variables in $orb(\{f\}, \mathcal{G}^a)$.

For example, given a variable x_f fixed to 0 by branching at a node a. Suppose some free variable $x_{f'} \in orb(\{f\}, \mathcal{G}^a)$ is fixed to 1 at a descendant node a'. Note that $f' > f$, provided that rule MIB is used. Then for $g \in \mathcal{G}^a$ such that $g(\{f'\}) = \{f\}$, we would have $g(F_1^{a'}) > F_1^{a'}$, thus $F_1^{a'}$ would not be a representative. Thus, variable $x_{f'}$ has value 0 in any representative solution S such that $F_1^a \subset S$.

Margot proves that the use of 0-setting alongside with minimum index branching and isomorphism pruning does not change the optimal value returned by the Branch & Bound.

Referring to Example 1, consider the enumeration tree of a Branch & Bound algorithm using MIB and ISP. Let node a be such that $F_0^a = \{1\}$ and $F_1^a = \{2\}$. Then F_1^a is not a representative, because $\pi_{1,2}(\{2\}) = \{1\}$ which is lexicographically larger. Thus node a is eliminated by isomorphism pruning.

Referring again to Example 1, consider the enumeration tree of a Branch & Bound algorithm using MIB, ISP and 0-setting. Let b be the node such that $F_0^b = \{1\}$ and $F_1^b = \varnothing$. Then $\mathcal{G}^b = \mathcal{G}$ and $orb(\{1\}, \mathcal{G}^b) = \{2\}$. By rule (i) of 0-setting, variable x_2 is set to 0 at node b. Therefore, if 0-setting is used, node a will not be explored by the enumeration tree. It indicates that the use of 0-setting enables early detection and pruning of non-representative solutions in the tree.

Isomorphism pruning must be used with minimum index branching in order to be valid. Consequently, isomorphism pruning is not flexible with respect to the definition of flexibility given in Section 1.2.

In practice, Margot represents symmetry group $\mathcal{G}$ using the *Schreier-Sims* representation [17]. A backtracking algorithm is proposed to compute $orb(\{f\}, \mathcal{G}^a)$.

In [12], a more flexible branching rule for isomorphism pruning is defined, alongside with more general 0- and 1-setting operations.

1.4 Orbital branching

In [16], the authors introduce a branching strategy called *orbital branching*, which is suitable to the general case where the symmetry group $\mathcal{G}$ is arbitrary.

The notion of orbit is extended to variables. We say that $\{x_{i_1}, x_{i_2}, ..., x_{i_k}\}$ is a variable orbit if

$$orb(\{i_1\}, \mathcal{G}) = \{\{i_1\}, \{i_2\}, ..., \{i_k\}\}$$

Let a be a node in the Branch & Bound tree. For a given variable orbit $O = \{x_{i_1}, x_{i_2}, ..., x_{i_k}\}$ of $\mathcal{G}^a$, orbital branching is to branch on the disjunction:

$$x_{i_1} = 1 \quad \vee \quad \sum_{\ell=1}^{k} x_{i_\ell} = 0 \tag{1}$$

By fixing either one or k variables, this disjunction often leads to an unbalanced branching tree. In order to create a more balanced tree, the authors in [14] consider an alternate branching strategy called *modified orbital branching* (MOB). For any $\alpha \in \mathbb{N}$, consider the disjunction:

$$\sum_{\ell=1}^{k} x_{i_\ell} \geq \alpha \quad \vee \quad \sum_{\ell=1}^{k} x_{i_\ell} \leq \alpha - 1$$

Since variables x_{i_ℓ}, $\ell \leq k$, belong to the same orbit, α variables can be arbitrarily chosen to take value 1 to enforce $\sum_{\ell=1}^{k} x_{i_\ell} \geq \alpha$. Similarly, $|O| - \alpha + 1$ variables can be arbitrarily chosen to take value 0 to enforce $\sum_{\ell=1}^{k} x_{i_\ell} \leq \alpha - 1$. Thus the disjunction can be strengthened to:

$$x_{i_\ell} = 1, \ \forall \ell \in \{1, ..., \alpha\} \quad \vee \quad x_{i_\ell} = 0, \ \forall \ell \in \{\alpha, ..., |O|\}$$

For example, consider a problem with orbit $O = \{x_1, x_2, x_3\}$. Suppose one uses MOB in the Branch & Bound tree and branches on orbit O at the root node. Then, for $\alpha = 2$, the left child is created by fixing $x_1 = x_2 = 1$ and the right child is created by fixing $x_2 = x_3 = 0$.

Even though orbital branching removes a significant proportion of symmetries, it is not guaranteed that only one representative of each orbit is explored.

In [15], the authors extend orbital branching so that the branching disjunction can be based on an arbitrary constraint.

1.5 Symmetry-breaking inequalities

Another way to break symmetries is to add symmetry breaking inequalities. A general description of symmetry breaking inequalities is given in [13], which is a generalization of the framework described in [4]. A closed set $F \subset \mathbb{R}$ is said to be a fundamental region for a symmetry group $\mathcal{G}$ if:

(i) $g(int(F)) \cap int(F) = \varnothing, \quad \forall g \in G,\ g \neq id$

(ii) $\cup_{g \in \mathcal{G}}\ g(F) = \mathbb{R}^n$

where $int(F)$ denotes the interior of F.

If F is a fundamental region for the symmetry group $\mathcal{G}$ of problem (ILP), then the following holds:

$$\min\Big\{cx \mid x \in P\Big\} \ = \ \min\Big\{cx \mid x \in P \cap F\Big\}$$

Indeed, for any optimal solution x^*, point (ii) guarantees that there exists $g \in \mathcal{G}$ such that $g^{-1}(x^*) \in F$. Point (i) guarantees that region F is not too large.

In [5], a linear description of a fundamental region is proposed:

$$F = \{x \in \mathbb{R}^n \mid (g(\overline{x}) - \overline{x}) \cdot x \leq 0, \forall g \in \mathcal{G}\} \tag{2}$$

where $\overline{x} \in \mathbb{R}$ is such that $g(\overline{x}) \neq \overline{x}$ for all $g \in \mathcal{G}$, $g \neq id$.

Thus, any inequality from linear description (2) can be added to (ILP). In [13], it is shown that these inequalities can be used even if the condition $g(\overline{x}) \neq \overline{x}$ for all $g \in \mathcal{G}$ does not hold.

Example 3. *Suppose $\mathcal{G}$ contains all permutations of I^n. In order to enforce a lexicographical ordering, one can use inequalities*

$$x_j \geq x_{j+1},\ \forall j \in \{1, ..., n-1\}$$

obtained, for each $j \in \{1, ..., n-1\}$, from description (2) by setting $\overline{x}_i = n - i$ for all $i \in I^n$ and $g = \pi_{j,j+1}$, the transposition of entries j and $j+1$.

1.6 Symmetry-breaking polytopes

For a symmetry group $\mathcal{G}$, the authors of [6] define the symmetry breaking polytope $P_S(\mathcal{G})$, called *symretope*, as the convex hull of the lexicographically maximal binary points w.r.t. $\mathcal{G}$:

$$P_S(\mathcal{G}) = conv\big\{x \in \{0,1\}^n \mid x \geq g(x),\ \forall g \in \mathcal{G}\big\}$$

The binary points in $P_S(\mathcal{G})$ are exactly those in fundamental region F with $\overline{x}_i = 2^{n-i}$.

As proved in [6], optimization over binary points in symretopes is NP-hard, thus a complete linear description is not available in general. It is still useful to have an IP formulation for binary points in those polytopes in order to handle the symmetries defined by $\mathcal{G}$. Inequalities defined in (2) provide such a formulation, but it has exponentially large coefficients and may not be computationally tractable. The authors of [6] consider *symresacks*, a special case of symretopes, where

the symmetry group $\mathcal{G}$ contains a unique non-trivial permutation. These polytopes can be seen as knapsack polytopes, where the knapsack constraint has exponential coefficients. This constraint can be replaced by an exponential number of minimal cover inequalities with $\{-1, 0, 1\}$ coefficients. It is proved in [6] that the separation problem of these minimal cover inequalities for the symresack can be solved in $O(n^2)$ time.

As an arbitrary symretope $P_S(\mathcal{G})$ can be written as the intersection of the symresacks $P_S(g)$ for each $g \in \mathcal{G}$, the authors derive IP formulations with small coefficients for symretopes, with separation in $O(|\mathcal{G}|n^2)$ time.

1.7 Variable fixing in symmetry-breaking polytopes

At a given node a of the Branch & Bound tree, some variables are set to 0, *i.e.* variables in F_0^a, and some to 1, *i.e.* variables in F_1^a. Based on these variables already fixed by previous branching decisions, some fixings of the remaining free variables can be performed. The idea of *variable fixing* is to restrict the solution space at each node a to be in a given symmetry-breaking polytope P. This is done by fixing to 0 (resp. 1) variables that would yield a solution outside P if fixed to 1 (resp. 0).

This method is introduced by Kaibel and Pfetsch [8].

Let C^d be the d-dimensional 0/1-cube. A face F of C^d is given by sets I_0, $I_1 \subset I^d$ as follows:

$$F = \{x \in C^d \mid x_i = 0 \; \forall i \in I_0 \text{ and } x_i = 1 \; \forall i \in I_1\}$$

For a polytope $P \subset C^d$ and a face F of C^d defined by (I_0, I_1), the smallest face of C^d that contains $P \cap F \cap \{0,1\}^d$ is denoted by $\text{Fix}_F(P)$, *i.e.* $\text{Fix}_F(P)$ is the intersection of all faces of C^d that contain $P \cap F \cap \{0,1\}^d$.

Referring to Example 2, consider $C^3 = \{x \in \mathbb{R}^3,\ 0 \leq x_i \leq 1 \text{ for all } i \in \{1, ..., 3\}\}$, and polytope $P_{ex} \subset C^3$:

$$P_{ex} = conv\{[0,\ 1,\ 1],\ [1,\ 0,\ 1],\ [1,\ 1,\ 0],\ [0,\ 1,\ 0]\}.$$

Let F be the face defined by $I_0 = \{2\}$ and $I_1 = \varnothing$. Namely, $F = \{x \in C^3 \mid x_2 = 0\}$. Then $P_{ex} \cap F \cap \{0,1\}^3 = [1,\ 0,\ 1]$ thus $\text{Fix}_F(P)$ is defined by $I_0^\star = \{2\}$ and $I_1^\star = \{1, 3\}$

Lemma 1. *([8]) If $Fix_F(P)$ is the non-empty face defined by $I_0^\star$ and $I_1^\star$, then:*
Index $i \in I_0^\star$ (resp. $I_1^\star$) iff all solutions of $P \cap F \cap \{0,1\}^d$ are such that $x_i = 0$ (resp. $x_i = 1$).

From an optimization perpective, P can be seen as the polytope defining the solution space $P \cap \{0,1\}^d$. Then, when optimizing over $P \cap \{0,1\}^d$, to each node a of the Branch & Bound tree corresponds a face $F(a)$ defined by F_0^a and F_1^a. The aim is thus to compute sets $I_0^\star$ and $I_1^\star$ defining $\text{Fix}_{F(a)}(P)$, at each node a. Then, if $\text{Fix}_F(P) = \varnothing$ then the node can be pruned. If $\text{Fix}_F(P) \neq \varnothing$, by Lemma 1, any free variable in $I_0^\star$ (resp. $I_1^\star$) can be set to 0 (resp. 1) (otherwise it would yield a solution outside $P \cap F(a) \cap \{0,1\}^d$).

In general, the problem of computing $\text{Fix}_F(P)$ is NP-hard. However, if one can optimize a linear function over $P \cap \{0,1\}^d$ in polynomial time, the fixing $(I_0^\star, I_1^\star)$ at (I_0, I_1) can be computed in polynomial time by solving $2(d - |I_0| - |I_1|)$ many linear optimization problems over $P \cap \{0,1\}^d$ [8].

If sets $I_0^\star$ and $I_1^\star$ relative to P cannot be computed efficiently, some relaxations of P can be considered. Instead of computing $\text{Fix}_F(P)$, one may only compute $\text{Fix}_F(P')$ where $P \subset P'$.

2 State of the art for matrix formulations with structured symmetry

In general, the symmetry group $\mathcal{G}$ acting on the variables is arbitrary. In this section, we consider a class of problems such that the symmetry group $\mathcal{G}$ has a particular structure. We suppose the variable set can be represented as a matrix $x = (x_{i,j})_{i \leq m, j \leq p}$. We suppose the symmetry group $\mathcal{G}$ is the set of all column permutations of the x matrix. This kind of symmetries arise naturally in many scheduling problems.

The MILP considered has the form:

$$\min \left\{ \sum_i \sum_j c_{i,j} x_{i,j} \mid \sum_i \sum_j a^k_{i,j} x_{i,j} \geq b^k \ \ \forall k \in \{1, ..., l\} \text{ and } x \in \mathcal{P}(m,p) \right\} \tag{3}$$

where $\mathcal{P}(m,p)$ is the set of the $m \times p$ binary matrices.

Note that in this case, the symmetry group $\mathcal{G}^a$ at a given node a of the Branch & Bound tree can be easily computed: permutations that act on columns j_1, ..., j_k are in $\mathcal{G}^a$ if and only if for each $i \in \{1, ..., m\}$, either variables x_{i,j_1},..., x_{i,j_k} are fixed to the same value or are all free.

For example, this type of symmetry can be found in graph coloring, where symmetry arises in particular from the permutation of colors. If entry $x_{i,j}$ of the solution matrix x corresponds to assigning color j to vertex i, then permuting colors corresponds to permuting columns of matrix x.

One common method used to break such kind of symmetry is to restrict the solution space to lexicographically non-increasing matrices. Hence, in this setting, a matrix $x \in \mathcal{P}(m,p)$ is said to be representative if its columns are lexicographically non-increasing.

2.1 Symmetry-breaking inequalities

For a problem of the form (3) whose symmetry group $\mathcal{G}$ is the set of all column permutations of the x matrix, Margot describes some symmetry-breaking inequalities in [13], obtained from linear description (2) for specific values of $\overline{x}$.

The first family of inequalities enforces a lexicographic order on the columns of x:

$$\sum_{i=1}^{m} 2^{m-i} x_{i,j} \geq \sum_{i=1}^{m} 2^{m-i} x_{i,j+1}, \ \ \forall j \in \{1, ..., p-1\}.$$

These inequalities state that the binary number encoded by the j^{th} column is larger than the binary number encoded by the $j + 1^{th}$ column.

However, they may get hard to handle when m is large. Thus, weaker inequalities can be used to break symmetries, such as:

$$\sum_{i=1}^{m} x_{i,j} \geq \sum_{i=1}^{m} x_{i,j+1}, \ \ \forall j \in \{1, ..., p-1\} \tag{4}$$

These inequalities do not enforce a lexicographical order on the columns, but restrict the search space to matrices such that the columns are ordered with respect to their total number of 1-entry. This ordering is weaker than lexicographic ordering since two different columns can have the same number of ones.

2.2 Orbital branching with branching rules

When the symmetry group $\mathcal{G}$ is the set of all column permutations of matrix x, Ostrowski *et al* [14] show the Branch & Bound search with modified orbital branching can be restricted to only representative solutions. The key idea is to enforce an additional branching rule restricting the variable orbits which can be branched on at each node. Namely, they define the minimum row-index (MI) branching rule which states that variable $x_{i,j}$ is eligible for branching if and only if for all rows $i' < i$, variables $x(i', j)$ have already been fixed. They prove that modified orbital branching alongside with MI branching rule is sufficient to ensure that only representative solutions are explored. As the MI rule may seem highly restrictive, they also propose some relaxations for which the same property holds, the most flexible branching rule being what is called *relaxed minimum-rank index* (RMRI).

Note that MOB used with MI can be seen as a particular case of Margot's isomorphism pruning used with MIB and 0-setting. Indeed, MOB and MI ensure that at each node a, F_1^a is a representative (otherwise it would lead to a non-representative solution). Furthermore, if a is the son of b where $x_{i,j}$ is fixed to 0, then all variables $x_{i,j'} \in orb(\{f\}, \mathcal{G}^a)$ with $j' > j$ are also fixed to 0 by disjunction (1). Finally, if f is a minimum row index free variable at node a, then by construction $F_1^a \cup \{f\}$ is a representative.

2.3 Orbitopes and orbitopal fixing

The convex hull of all $m \times p$ binary matrices with lexicographically non-increasing columns is called a *full orbitope* and is denoted by $\mathcal{P}_0(m, p)$. Special cases of full orbitopes are *packing* and *partitioning orbitopes*, which are restrictions to matrices with at most (resp. exactly) one 1-entry in each row.

The key idea is to restrict to $\mathcal{P}_0(m, p)$ the search space of MILP (3), whose symmetry group $\mathcal{G}$ is the set of all column permutations of the x matrix, in order to explore only lexicographically non-increasing solutions in the Branch & Bound tree.

If constraints $\sum_i \sum_j a_{i,j}^k x_{i,j} \geq b^k$, $\forall k \in \{1, ..., l\}$ include the *row-sum* inequalities $\sum_j x_{i,j} = 1$ (resp. $\sum_j x_{i,j} \leq 1$), $\forall i \in \{1, ...m\}$, then the search can be restricted to a partitioning (resp. packing) orbitope.

2.3.1 Full orbitopes

Full orbitopes can be seen as a special case of symretopes.

The authors of [6] specifically address this particular case, defining *orbisacks* as the convex hull of all $m \times 2$ binary matrices whose first column is lexicographically larger than or equal to the second column. It is shown that only $p - 1$ orbisacks need to be considered to obtain an IP-formulation with small coefficients for the full orbitope $\mathcal{P}_0(m, p)$. Furthermore, they prove these inequalities can be separated in $O(mp)$ time.

No complete description of the full orbitope $\mathcal{P}_0(m, p)$ is known, and computer experiments conducted in [7] indicate that its facet defining inequalities are extremely complicated. However, exploiting the general framework of polyhedral branching systems defined in [7], a compact extended formulation is constructed by combining extended formulations of simpler polyhedra.

In [10], a complete description of orbisacks is given.

2.3.2 Packing and partitioning orbitopes

In [9], shifted columns inequalities are introduced. The authors prove that these inequalities, together with non-negativity constraints and row-sum inequalities, completely describe both packing and partitioning orbitopes. A polynomial time separation algorithm for the exponentially large class of shifted columns inequalities is also given.

2.3.3 Variable fixing in partitioning orbitopes: orbitopal fixing

Orbitopal fixing is variable fixing with polytope P being an orbitope.

In [8], the authors take advantage of the shifted columns inequalities for partitioning and packing orbitopes in order to characterize the sets $I_0^\star$ and $I_1^\star$ defining $\text{Fix}_F(P)$ where P is the partitioning (or packing) orbitope and F is defined by (F_0^a, F_1^a), at a given node a of the Branch & Bound tree.

3 Orbitopal fixing for the full orbitope

Orbitopal fixing introduced in [8] is particularly interesting to break symmetries as it restricts the search in the Branch & Bound tree to only representative solutions while remaining flexible. In particular, it does not constrain the set of variables nor the disjunction that can be branched on. Note also that no additional inequalities need to be appended to the model, thus it does not increase the size of the LP solved at each node. The authors of [8] have proved that for any face F of C^d, the sets $I_0^\star$ and $I_1^\star$ defining $\text{Fix}_F(P)$ can be characterized when P is a partitioning or a packing orbitope.

There are many problems whose symmetry group $\mathcal{G}$ is the set of all column permutations among given subsets of columns of the x matrix, but whose search space cannot be restricted to a partitioning or a packing orbitope. For example, the UCP with identical units is such that the plans of the units, *i.e.* the columns, can be permuted in any solution, but there is no general restriction on the number of ones on each row t of matrix X^h, corresponding to the number of type h units up at time t.

As detailed in Sections 1.4 and 2.2, the authors in [14] propose to break this kind of "all-column permutations" symmetries using MOB, *i.e.* by branching on a disjunction that fixes a larger number of variables than the classical disjunction $x_{i,j} = 0 \ \vee \ x_{i,j} = 1$. If the use of MOB removes a large number of symmetries, it provides no guarantee that only representative solutions are explored in the Branch & Bound tree. The only way to ensure that MOB will remove all non-representative solutions is to use it alongside with a branching rule that restricts the choice of the variables to be branched on.

We explore a different approach, where, at each node, orbitopal fixing for the full orbitope is used to fix some of the remaining variables left free by branching. At a given node a, once some variables have been fixed by branching, we restrict the solution at node a to be in the full orbitope by setting to 0 (resp. to 1) variables that would yield a non-lexicographically ordered solution if set to 1 (resp. to 0). This approach preserves flexibility as the choice of the branching disjunctions and variables remains totally free.

In this section, we will extend the notion of orbitopal fixing to the full orbitope, by characterizing sets $I_0^\star$ and $I_1^\star$ corresponding to the fixing of the full orbitope at (I_0, I_1).

3.1 Intersection with the full orbitope

Let P be the full orbitope $\mathcal{P}_0(m,n)$, and let F^a be a face of $\{0,1\}^{(m,n)}$ defined by sets $\mathcal{I}=(I_0,I_1)$. Recall that the smallest face of $[0,1]^{(m,n)}$ that contains $P\cap F^a\cap\{0,1\}^{(m,n)}$ is denoted by $\mathrm{Fix}_{F^a}(P)$. If $\mathrm{Fix}_{F^a}(P)\neq\varnothing$, there exist sets $I_0^\star$ and $I_1^\star$ defining $\mathrm{Fix}_{F^a}(P)$.

For any matrix $X\in P$, we denote by $X(j)$ the j^{th} column of X and by $X(i,j)$ the entry at row i, column j.

Definition 1. *For a given face F of $\{0,1\}^{(m,n)}$, a matrix X is said to be $F(P)$-minimal (resp. $F(P)$-maximal) if $X\in P\cap F\cap\{0,1\}^{(m,n)}$ and for any matrix $Y\in P\cap F\cap\{0,1\}^{(m,n)}$, $X(j)\leq Y(j)$ (resp. $X(j)\geq Y(j)$) $\forall j\in\{1,...,n\}$, i.e. $X(j)$ is lexicographically less (resp. greater) than or equal to $Y(j)$, $\forall j\in\{1,...,n\}$.*

In the following, we construct a $F(P)$-minimal matrix and a $F(P)$-maximal matrix in the case $\mathrm{Fix}_{F^a}(P)\neq\varnothing$.

First some definitions are introduced. For any couple of index sets $\mathcal{S}=(S_0,S_1)$, consider matrix $X_\mathcal{S}$ whose entries with index in S_0 are set to 0, entries with index in S_1 are set to 1 and remaining entries in $F_\mathcal{S}=\{1,...,m\}\times\{1,....,n\}\backslash(S_0\cup S_1)$ contain $\times$, representing free variables.

Definition 2. *Two columns j and j' of a matrix X are said to be fixedly different on a given row i if $X(i,j)\neq\times$ and $X(i,j')\neq\times$ and $X(i,j)\neq X(i,j')$.*

Definition 3. *Consider two columns j and j', of a matrix $X_\mathcal{S}$. Let index $i_f^\mathcal{S}(j,j')$ be the index of the first row of $X_\mathcal{S}$ where columns j and j' are fixedly different. If there is no such row, then $i_f^\mathcal{S}(j,j')$ is arbitrarily set to $m+1$.*

In other words,

- for each row $i<i_f^\mathcal{S}(j,j')$, either $X_\mathcal{S}(i,j)$ is free or $X_\mathcal{S}(i,j')$ is free or $X_\mathcal{S}(i,j)=X_\mathcal{S}(i,j')$.
- at row $i_f^\mathcal{S}(j,j')$, entries $[X_\mathcal{S}(i,j),X_\mathcal{S}(i,j')]$ have one of the following forms: $[1,\ 0]$ or $[0,\ 1]$.

Definition 4. *A row i of matrix $X_{M'}$ is said to be (j,j')-discriminating if it has one of the following forms on columns j and j': $[\times,\ \times]$ or $[\times,\ 0]$ or $[1,\ \times]$, where $\times$ stands for a free variable.*

In other words, row i is discriminating if it contains at least one free variable, and values 0 or 1 could be assigned to this or these variables in such a way that $X_\mathcal{S}(i,j)>X_\mathcal{S}(i,j')$.

In the following, two matrix sequences $(\overline{\mathcal{M}}^j)_{j\in\{1,...,n\}}$ and $(\underline{\mathcal{M}}^j)_{j\in\{1,...,n\}}$ are defined. Matrix $\overline{\mathcal{M}}^n$ is constructed to be $F^a(P)$-maximal and matrix $\underline{\mathcal{M}}^1$ is constructed to be $F^a(P)$-minimal, if $\mathrm{Fix}_{F^a}(P)\neq\varnothing$.

The idea of the construction is the following. For $j=1$, matrix $\overline{\mathcal{M}}^1$ corresponds to matrix $X_\mathcal{I}$ where values 1 are assigned to all free variables in the first column of $X_\mathcal{I}$. Obviously, for any $X\in P\cap F^a\cap\{0,1\}^{(m,n)}$, $\overline{\mathcal{M}}^1(1)\geq X(1)$. For each k, for any $X\in P\cap F^a\cap\{0,1\}^{(m,n)}$, $\overline{\mathcal{M}}^k(k)\geq X(k)$ holds. Thus, free variables in column $k+1$ of matrix $\underline{\mathcal{M}}^k$ are set to 0 or 1 in matrix $\underline{\mathcal{M}}^{k+1}$ by propagating values from column k, so that column $k+1$ is maximum among all columns less than or equal to column k. The other columns remain unchanged from $\underline{\mathcal{M}}^k$ to $\underline{\mathcal{M}}^{k+1}$.

Similarly, for $j=n$, matrix $\underline{\mathcal{M}}^n$ corresponds to matrix $X_\mathcal{I}$ where values 0 are assigned to all free variables in the last column of $X_\mathcal{I}$. Then for each $k<n$, the free variables in column k of

matrix $\underline{\mathcal{M}}^{k+1}$ are set to 0 or 1 in matrix $\underline{\mathcal{M}}^{k}$ by propagating values from column $k+1$, so that column k is minimum among all columns larger than or equal to column $k+1$. The other columns remain unchanged from $\underline{\mathcal{M}}^{k+1}$ to $\underline{\mathcal{M}}^{k}$.

Note that in matrix $\underline{\mathcal{M}}^{k}$ (resp. $\overline{\mathcal{M}}^{k}$), there is no remaining free variables in columns $\{k, ..., n\}$ (resp. $\{1, ..., k\}$).

In the following, we define matrix sequence $(\underline{\mathcal{M}}^{j})_{j\in\{1,...,n\}}$ using a sequence $\underline{\mathcal{S}}^{j} = (\underline{S}_0^j, \underline{S}_1^j)$, where $\underline{\mathcal{S}}^{j}$ is a couple of two disjoint index subsets. Matrix $\underline{\mathcal{M}}^{j}$ is defined as matrix $X_{\underline{\mathcal{S}}^{j}}$. Sequence $(\overline{\mathcal{M}}^{j})_{j\in\{1,...,n\}}$ can be defined similarly using a sequence $\overline{\mathcal{S}}^{j}$, where $\overline{\mathcal{S}}^{j}$ is a couple of two disjoint index subsets. The construction is not detailed here.

Sequence $(\underline{\mathcal{S}}^{j})$ is defined as follows.

- For $j = n$, $\underline{S}_1^n = I_1$ and $\underline{S}_0^n = \{(i, n) \in F_{\mathcal{I}}\} \cup I_0$.
- For $j < n$, define $i_f = i_f^{\underline{\mathcal{S}}^{j+1}}(j, j+1)$ for convenience. Consider row i_f in matrix $\underline{\mathcal{M}}^{j+1}$.
  - If $i_f = m+1$ or if row i_f has the form $[1,\ 0]$ on columns j and $j+1$, then,

$$\underline{S}_1^j = \underline{S}_1^{j+1} \cup \left\{(i,j) \in F_{\mathcal{I}} \mid (i, j+1) \in \underline{S}_1^{j+1} \text{ and } i < i_f\right\}$$

$$\underline{S}_0^j = \underline{S}_0^{j+1} \cup \left\{(i,j) \in F_{\mathcal{I}} \mid (i,j) \notin \underline{S}_1^j\right\}$$

    i.e. free variables in column j of $\underline{\mathcal{M}}^{j}$ are set such that columns j and $j+1$ are equal from row 1 to row $i_f - 1$. Every other free variables in column j are set to 0.
  - Otherwise, row i_f has the form $[0,\ 1]$. In this case:
    - If, in matrix $\underline{\mathcal{M}}^{j+1}$, there is no row $i \in \{1, ..., i_f - 1\}$ which is $(j, j+1)$-discriminating, then, for all $j' \leq j$, $\underline{\mathcal{S}}^{j'}$ is set to $\mathcal{S}^{\varnothing}$, which is arbitrarily defined as:

$$\mathcal{S}_0^{\varnothing} = \{(1,1)\}$$

$$\mathcal{S}_1^{\varnothing} = \{1, ..., m\} \times \{1, ..., n\} \backslash \mathcal{S}_0^{\varnothing}$$

    - Otherwise consider row i_{ld}, the last $(j, j+1)$-discriminating row before row i_f in matrix $\underline{\mathcal{M}}^{j+1}$, *i.e.*

$$i_{ld} = \max_{i\in\{1,...,i_f-1\}} \left\{i \text{ is } (j,j+1) - \text{discriminating in } \underline{\mathcal{M}}^{j+1}\right\}.$$

      Then,

$$\underline{S}_1^j = \underline{S}_1^{j+1} \cup \{(i_{ld}, j)\} \cup \left\{(i,j) \in F_{\mathcal{I}} \mid (i, j+1) \in \underline{S}_1^{j+1} \text{ and } i < i_{ld}\right\}$$

$$\underline{S}_0^j = \underline{S}_0^{j+1} \cup \left\{(i,j) \in F_{\mathcal{I}} \mid (i,j) \notin \underline{S}_1^j\right\}.$$

      i.e. free variables in column j of $\underline{\mathcal{M}}^{j}$ are set such that columns j and $j+1$ are equal from row 1 to row $i_{ld} - 1$, and such that row i_{ld} has the form $[1,\ 0]$ on columns j and $j+1$. Every other free variables in column j are set to 0.

The following result shows that either $\mathrm{Fix}_{F^a}(P)$ is empty, and in this case $\underline{\mathcal{S}}^1 = \overline{\mathcal{S}}^n = \mathcal{S}^{\varnothing}$, or matrices $\underline{\mathcal{M}}^1$ and $\overline{\mathcal{M}}^n$ are respectively $F^a(P)$-minimal and $F^a(P)$-maximal.

Theorem 1.

- *If* $\underline{\mathcal{S}}^1 = \mathcal{S}^{\varnothing}$ *or* $\overline{\mathcal{S}}^n = \mathcal{S}^{\varnothing}$ *then* $Fix_{F^a}(P) = \varnothing$.
- *Otherwise matrix* $\underline{\mathcal{M}}^1$ *is* $F^a(P)$*-minimal and matrix* $\overline{\mathcal{M}}^n$ *is* $F^a(P)$*-maximal.*

Proof. The following result is proved by induction: for all $j \in \{1, ..., n\}$,

- if $\underline{\mathcal{S}}^j \neq \mathcal{S}^{\varnothing}$, then, $\forall X \in \mathrm{Fix}_{F^a}(P)$, $\underline{\mathcal{M}}^j(j) \leq X(j)$,
- if $\underline{\mathcal{S}}^j = \mathcal{S}^{\varnothing}$ then $\mathrm{Fix}_{F^a}(P) = \varnothing$.

A similar proof can be done to obtain the corresponding result for $\overline{\mathcal{S}}^j$ and $\overline{\mathcal{M}}^j$. The details are not provided here.

If $j = n$, then by construction, $\underline{\mathcal{S}}^n \neq \mathcal{S}^{\varnothing}$. Since all $(i, n) \in F_{\mathcal{I}}$ are set to 0 in matrix $\underline{\mathcal{M}}^n$, necessarily $\forall X \in P \cap F^a \cap \{0,1\}^{(m,n)}$, $\underline{\mathcal{M}}^n(n) \leq X(n)$.

Otherwise, consider $j < n$ and suppose the result holds for $j + 1$.

If $\underline{\mathcal{S}}^j \neq \mathcal{S}^{\varnothing}$, suppose there exists $X \in P \cap F^a \cap \{0,1\}^{(m,n)}$ such that $\underline{\mathcal{M}}^j(j) > X(j)$. Consider the first row i such that columns $X(j)$ and $\underline{\mathcal{M}}^j(j)$ are different. As $\underline{\mathcal{M}}^j(j) > X(j)$, we have $X(i, j) = 0$ and $\underline{\mathcal{M}}^j(i, j) = 1$. By construction, since $(i, j) \in F_{\mathcal{I}}$ and $\underline{\mathcal{M}}^j(i, j) = 1$, for all $i' < i$, $\underline{\mathcal{M}}^j(i', j) = \underline{\mathcal{M}}^j(i', j + 1)$.

We now consider two cases:

- If $\underline{\mathcal{M}}^j(i, j + 1) = 1$, then since $\underline{\mathcal{M}}^j(j + 1) = \underline{\mathcal{M}}^{j+1}(j + 1), \underline{\mathcal{M}}^{j+1}(j + 1) > X(j)$.
- If $\underline{\mathcal{M}}^j(i, j + 1) = 0$, then, from the construction of $\underline{\mathcal{M}}^j$, row $i_f = i_f^{M^{j+1}}(j, j + 1)$ in matrix $\underline{\mathcal{M}}^{j+1}$ has the form $[0,\ 1]$ on columns j and $j+1$ (otherwise $\underline{\mathcal{M}}^j(i, j)$ would have been set to 0). In this case, row i corresponds to the last $(j, j+1)$-discriminating row of matrix $\underline{\mathcal{M}}^{j+1}$ before row i_f. Thus, for each $i' \in \{i + 1, i_f - 1\}$ such that $(i', j) \in F_{\mathcal{I}}$, we have $\underline{\mathcal{M}}^j(i', j + 1) = 1$. If for such an i', $X(i', j) = 0$ then since $\underline{\mathcal{M}}^j(j + 1) = \underline{\mathcal{M}}^{j+1}(j + 1)$, $\underline{\mathcal{M}}^{j+1}(j + 1) > X(j)$. Otherwise, as row i_f in matrix $\underline{\mathcal{M}}^{j+1}$ has the form $[0,\ 1]$ on columns j and $j + 1$, it follows $(i_f, j) \in F_0^a$, thus $X(i_f, j) = 0$. Consequently $\underline{\mathcal{M}}^{j+1}(j + 1) > X(j)$ holds too.

By the induction hypothesis, $X(j + 1) \geq \underline{\mathcal{M}}^{j+1}(j + 1)$ thus $X(j + 1) > X(j)$, which contradicts $X \in P$.

If $\underline{\mathcal{S}}^j = \mathcal{S}^{\varnothing}$, consider the following two cases:

- If $\underline{\mathcal{S}}^{j+1} = \mathcal{S}^{\varnothing}$ then by the induction hypothesis, $\mathrm{Fix}_{F^a}(P) = \varnothing$.
- Otherwise, $\underline{\mathcal{S}}^{j+1} \neq \mathcal{S}^{\varnothing}$. Recall $i_f = i_f^{\mathcal{S}^{j+1}}(j, j + 1)$. Then, by construction of matrix $\underline{\mathcal{M}}^j$, row i_f of matrix $\underline{\mathcal{M}}^{j+1}$ has the form $[0,\ 1]$ on columns j and $j + 1$ and there is no row $i \in \{1, ..., i_f - 1\}$ in matrix $\underline{\mathcal{M}}^{j+1}$ which is $(j, j + 1)$-discriminating. As column $j + 1$ is completely fixed in matrix $\underline{\mathcal{M}}^{j+1}$, each row $i \in \{1, ..., i_f - 1\}$ of matrix $\underline{\mathcal{M}}^{j+1}$ has one the following forms on columns j and $j + 1$: $[1,\ 1]$ or $[0,\ 0]$ or $[\times,\ 1]$. Therefore, if $\mathrm{Fix}_{F^a}(P)$ were not empty, then $P \cap F^a \cap \{0,1\}^{(m,n)} \neq \varnothing$ and for any $X \in P \cap F^a \cap \{0,1\}^{(m,n)}$, even if $X(i, j) = 1$ for each $(i, j) \in F_{\mathcal{I}}$, $\underline{\mathcal{M}}^{j+1}(j+1) > X(j)$ would hold. By the induction hypothesis, $X(j + 1) \geq \underline{\mathcal{M}}^{j+1}(j + 1)$ thus $X(j + 1) > X(j)$, which contradicts $X \in P$.

□

Now, in case $\text{Fix}_{F^a}(P) \neq \varnothing$, sets $I_0^\star$ and $I_1^\star$ can be characterized, using sets $\underline{\mathcal{S}}^1 = (\underline{S}_0^1, \underline{S}_1^1)$ and $\overline{\mathcal{S}}^n = (\overline{S}_0^n, \overline{S}_1^n)$ defining $F^a(P)$-minimal and $F^a(P)$-maximal matrices $\underline{\mathcal{M}}^1$ and $\overline{\mathcal{M}}^n$.

First, for each $j \in \{1, ..., m\}$, consider row i_j, the first row at which columns $\underline{\mathcal{M}}^1(j)$ and $\overline{\mathcal{M}}^n(j)$ differs, defined as:

$$i_j = \min \left\{ i \in \{1, ..., m\} \mid \underline{\mathcal{M}}^1(i,j) \neq \overline{\mathcal{M}}^n(i,j) \right\}$$

If columns $\underline{\mathcal{M}}^1(j)$ and $\overline{\mathcal{M}}^n(j)$ are equal, then i_j is arbitrarily set to $m+1$. By definition of $F^a(P)$-minimal and $F^a(P)$-maximal matrices, $\underline{\mathcal{M}}^1(i_j, j) < \overline{\mathcal{M}}^n(i_j, j)$. Note that since for all $(i,j) \in I_0$ (resp. I_1), $\underline{\mathcal{M}}^1(i,j) = 0$ (resp. 1) and $\overline{\mathcal{M}}^n(i,j) = 0$ (resp. 1), (i_j, j) is a free variable *i.e.* $(i_j, j) \in F_{\mathcal{I}}$.

Now define sets I_0^+ and I_1^+:

$$I_0^+ = \left\{ (i,j) \in F_I \mid i < i_j \text{ and } \overline{\mathcal{M}}^n(i,j) = 0 \right\}$$

$$I_1^+ = \left\{ (i,j) \in F_I \mid i < i_j \text{ and } \underline{\mathcal{M}}^1(i,j) = 1 \right\}$$

The sets $(I_0^\star, I_1^\star)$ defining $\text{Fix}_{F^a}(P)$ can be characterized using I_0^+ and I_1^+, as shown in the following.

Theorem 2.

$$I_0^\star = I_0 \cup I_0^+$$
$$I_1^\star = I_1 \cup I_1^+$$

Proof. First, we prove that $I_0^+ \subset I_0^\star$ and $I_1^+ \subset I_1^\star$. Suppose $I_0^+ \not\subset I_0^\star$ or $I_1^+ \not\subset I_1^\star$. Let $(i,j) \in (I_0^+ \backslash I_0^\star) \cup (I_1^+ \backslash I_1^\star)$. Consider $i_0 = \min\{i' \mid (i',j) \in (I_0^+ \backslash I_0^\star) \cup (I_1^+ \backslash I_1^\star)\}$. Suppose $(i_0, j) \in I_0^+ \backslash I_0^\star$. As $(i_0, j) \notin I_0^\star$, there exists $X \in P \cap F^a \cap \{0,1\}^{(m,n)}$ such that $X(i_0, j) = 1$. As $(i_0, j) \in I_0^+$, $\overline{\mathcal{M}}^n(i_0, j) = 0$. If for all $i' < i_0$, $X(i', j) \geq \overline{\mathcal{M}}^n(i', j)$ then the following would hold: $X(j) > \overline{\mathcal{M}}^n(j)$, contradicting the fact that $\overline{\mathcal{M}}^n$ is $F^a(P)$-maximal. Thus, there exists a row $i_1 < i_0$ such that $\overline{\mathcal{M}}^n(i_1, j) = 1$ and $X(i_1, j) = 0$. As $(i_0, j) \in I_0^+$, $i_1 < i_0 < i_j$, so $\underline{\mathcal{M}}^1(i_1, j) = 1$ too. Thus $(i_1, j) \in I_1^+$. However, $(i_1, j) \notin I_1^\star$ because $X \in P \cap F^a \cap \{0,1\}^{(m,n)}$ and $X(i_1, j) = 0$. The contradiction comes from the fact that $i_1 < i_0$ and $i_1 \in \{i' \mid (i',j) \in (I_0^+ \backslash I_0^\star) \cup (I_1^+ \backslash I_1^\star)\}$.

The proof is the same if we suppose $(i_0, j) \in I_1^+ \backslash I_1^\star$. This proves $I_0^+ \subset I_0^\star$ and $I_1^+ \subset I_1^\star$, thus $I_0 \cup I_0^+ \subset I_0^\star$ and $I_1 \cup I_1^+ \subset I_1^\star$.

Now we prove $I_0^\star \subset I_0 \cup I_0^+$ and $I_1^\star \subset I_1 \cup I_1^+$. For this, it suffices to show that for each $(i,j) \notin I_0^\star \cup I_1^\star$, there exists a solution $X_0 \in P \cap F^a \cap \{0,1\}^{(m,n)}$ such that $X_0(i,j) = 0$ and a solution $X_1 \in P \cap F^a \cap \{0,1\}^{(m,n)}$ such that $X_1(i,j) = 1$.

Let $(i,j) \notin I_0^\star \cup I_1^\star$. Consider index $(i_j, j) \in F_{\mathcal{I}}$.

Solution $\underline{\mathcal{M}}^1$ is such that $\underline{\mathcal{M}}^1(i_j, j) = 0$ and solution $\overline{\mathcal{M}}^n$ is such that $\overline{\mathcal{M}}^n(i_j, j) = 1$. So if $i = i_j$, the result is proved.

Now suppose $i \neq i_j$. Note that for all $i' < i_j$, $\underline{\mathcal{M}}^1(i', j) = \overline{\mathcal{M}}^n(i', j)$, therefore $(i', j) \in I_0^\star \cup I_1^\star$. Thus $i > i_j$.

Consider solutions X_1 and X_0 defined as follows.

$$\forall j' \in \{1, ..., j-1\},\ \forall i' \in \{1, ..., m\},\ X_1(i', j') = X_0(i', j') = \overline{\mathcal{M}}^n(i', j')$$

$$\forall j' \in \{j+1, ..., n\},\ \forall i' \in \{1, ..., m\},\ X_1(i', j') = X_0(i', j') = \underline{\mathcal{M}}^1(i', j')$$
$$\forall i' < i_j,\ X_1(i', j) = X_0(i', j) = \underline{\mathcal{M}}^1(i', j) = \overline{\mathcal{M}}^n(i', j)$$
$$X_1(i_j, j) = 0 \text{ and } \forall i' \in F_{\mathcal{I}} \backslash \{i\},\ i' > i_j,\ X_1(i, j) = 1$$
$$X_0(i_j, j) = 1 \text{ and } \forall i' \in F_{\mathcal{I}} \backslash \{i\},\ i' > i_j,\ X_0(i, j) = 0$$

Recall that $\overline{\mathcal{M}}^n(i_j, j) = 1$ and $\underline{\mathcal{M}}^1(i_j, j) = 0$, therefore $\overline{\mathcal{M}}^n(j) \geq X_0(j) > X_1(j) > \underline{\mathcal{M}}^1(j)$.

As $\overline{\mathcal{M}}^n$ and $\underline{\mathcal{M}}^1 \in P$, $\overline{\mathcal{M}}^n(j-1) \geq \overline{\mathcal{M}}^n(j)$ and $\underline{\mathcal{M}}^1(j) \geq \underline{\mathcal{M}}^1(j+1)$. Thus X_1 and X_0 are also in $P \cap F^a \cap \{0,1\}^{(m,n)}$ and are such that $X_1(i,j) = 1$ and $X_0(i,j) = 0$. This concludes the proof. □

The fixing $\text{Fix}_{F^a}(\mathcal{P}_0(m,p))$ being now characterized, in the next two sections we will define two fixing algorithms for the full orbitope: *static orbitopal fixing* and *dynamic orbitopal fixing*.

3.2 Static orbitopal fixing

When solving MILP (3) with Branch & Bound, static orbitopal fixing can be performed at each node of the branching tree in order to ensure that any solution x enumerated is such that $x \in \mathcal{P}_0(m,p)$.

The static orbitopal fixing algorithm at node a is the following:

- Set $I_0 = F_0^a$ and $I_1 = F_1^a$
- Compute matrices $\underline{\mathcal{M}}^1$ and $\overline{\mathcal{M}}^n$
- For each $(i,j) \in I_0^+$, fix variable $x_{i,j}$ to 0
- For each $(i,j) \in I_1^+$, fix variable $x_{i,j}$ to 1

Let τ be a branch & bound tree of MILP (3), in which static orbitopal fixing is used. No assumptions are made on the branching strategy used.

Theorem 3. *For each feasible solution X to MILP (3), there is exactly one solution $X' \in orb(X)$ enumerated in tree τ.*

Proof. Since the sets $(I_0 \cup I_0^+,\ I_1 \cup I_1^+)$ define $\text{Fix}_{F^a}(\mathcal{P}_0(m,p))$, the result follows directly. □

3.3 Dynamic orbitopal fixing

In the previous sections, we have considered that a column c of a binary matrix is lexicographically larger than or equal to a column c' if:

$$\sum_{i=1}^{n} 2^{n-i} c[i] \ \geq \ \sum_{i=1}^{n} 2^{n-i} c'[i].$$

Note that the order on the n rows is arbitrary. Considering a bijection $\phi : \{1, ..., n\} \to \{1, ..., n\}$, one could define the lexicographical order as follows:

$$\sum_{i=1}^{n} 2^{n-i} c[\phi(i)] \ \geq \ \sum_{i=1}^{n} 2^{n-i} c'[\phi(i)].$$

The idea of dynamic fixing is to use the branching decisions to define reorderings ϕ_a of the row indices, depending on the branch a of the branch & bound tree.

First suppose that all branching disjunctions have the form:

$$x_{i,t} = 0 \quad \vee \quad x_{i,t} = 1,$$

For each node a, let

$$x_{i_a,t_a} = 0 \quad \vee \quad x_{i_a,t_a} = 1,$$

be the branching disjunction at node a. We define subset $T_a \subset \mathcal{T}$, integer $e_a \in \mathbb{N}$ and bijection $\phi_a : \{1, ..., e_a\} \to T_a$ as follows. If a is the root node, then

$$\begin{cases} T_a = \{t_a\} \\ e_a = 1 \\ \phi_a(1) = t_a \end{cases}$$

Otherwise, let b be the father node of a and:

$$\begin{cases} T_a = T_b \cup \{t_a\} & \\ e_a = |T_a| & \\ \forall i \in \{1, ..., e_b\} & \phi_a(i) = \phi_b(i) \\ \text{If } t_a \notin T_b, & \phi_a(t_a) = e_a \end{cases}$$

Thus, *dynamic orbitopal fixing* is such that at each node a, orbitopal fixing is performed on rows $T_a = \{ \phi_a(1), \phi_a(2), ..., \phi_a(e_a) \}$, with respect to lexicographical ordering ϕ_a.

Let τ be a branch & bound tree of MILP (3), in which dynamic orbitopal fixing is used and branching disjunctions have the form:

$$x_{i,t} = 0 \quad \vee \quad x_{i,t} = 1,$$

for given i, t.

Theorem 4. *For each feasible solution X to MILP (3), there is exactly one solution $X' \in orb(X)$ enumerated in tree τ.*

Proof. The idea of the proof is to compute the representative $X' \in orb(X)$ enumerated by τ by following the branching decisions taken in τ, and seeing there is a unique X' possible.

First consider the branching disjunction at the root node a:

$$x_{i_0,t_0} = 0 \quad \vee \quad x_{i_0,t_0} = 1,$$

Then $\phi_a(1) = t_0$.

Let n_{t_0} be the number of "1" entries on row t_0 of matrix X. Since row t_0 is the first row with respect to the lexicographical order ϕ_a, any $X' \in orb(X)$ enumerated by the branching tree will be such that:

$$X'(t_0, j) = 1, \ \forall j \in \{1, ..., n_{t_0}\}$$
$$X'(t_0, j) = 0, \ \forall j \in \{n_{t_0} + 1, ..., n\}$$

Indeed, dynamic orbitopal fixing is enforced in τ, thus any solution enumerated by τ must be lexicographically non-increasing with respect to ϕ_a.

Thus, if we denote by b_1 the son of a such that $x_{i_0,t_0} = 1$ and b_0 the son of a_0 such that $x_{i_0,t_0} = 0$, then the next node to be considered will be:

$$\begin{cases} b_1 & \text{if } i_0 \leq n_{t_0} \\ b_0 & \text{otherwise.} \end{cases}$$

Indeed, if $i_0 \leq n_{t_0}$ (resp. $i_0 > n_{t_0}$) then any $X' \in orb(X)$ enumerated by τ is such that $X'(i_0, t_0) = 1$ (resp. $X'(i_0, t_0) = 0$). Thus there is no representative of X in the branch $x_{i_0,t_0} = 0$ (resp. $x_{i_0,t_0} = 0$).

Now suppose $i_0 \leq n_{t_0}$, so the node considered is b_1. If the node considered were b_0, we could proceed similarly. Consider the branching disjunction at node b_1:

$$x_{i_1,t_1} = 0 \quad \vee \quad x_{i_1,t_1} = 1,$$

If $t_1 = t_0$ then, by the same arguments as at the root node, there is exactly one branch in which is enumerated any $X' \in orb(X)$, and this branch can be easily determined.

Otherwise $t_1 \neq t_0$, and $\phi_{a_1}(1) = t_0$, $\phi_{a_1}(2) = t_1$.

Let $n^1_{t_1}$ (resp. $n^0_{t_1}$) be the number of columns j such that $X(t_0, j) = 1$ (resp. $X(t_0, j) = 0$) and $X(t_1, j) = 1$. Since row t_1 is second with respect to lexicographical order ϕ_{a_1}, any $X' \in orb(X)$ enumerated by the branching tree will be such that:

$$X'(t_1, j) = 1, \ \forall j \in \{1, ..., n^1_{t_1}\}$$
$$X'(t_1, j) = 0, \ \forall j \in \{n^1_{t_1} + 1, ..., n_{t_0}\}$$
$$X'(t_1, j) = 1, \ \forall j \in \{n_{t_0} + 1, ..., n^0_{t_1}\}$$
$$X'(t_1, j) = 0, \ \forall j \in \{n^0_{t_1} + 1, ..., n\}$$

Thus, all $X' \in orb(X)$ enumerated by τ have the same value v in entry (i_1, t_1), and this value can be determined, as previously, by finding in which of the sets $\{1, ..., n^1_{t_1}\}$, $\{n^1_{t_1}+1, ..., n_{t_0}\}$, $\{n_{t_0}+1, ..., n^0_{t_1}\}$, $\{n^0_{t_1}+1, ..., n\}$ does index i_1 belong. Therefore, since for all $X' \in orb(X)$ enumerated by τ, $X'(i_1, t_1) = v$, there is exactly one branch in which any $X' \in orb(X)$ is enumerated: $x_{i_1,t_1} = v$. This defines the next node to consider. This process can be repeated until a leaf node a is reached. At that point, all entries of X' are determined. By construction, X' is lexicographically non-increasing with respect to ϕ_a, and $X' \in orb(X)$.

Furthermore, X' is the only element of $orb(X)$ enumerated by τ, since at each node we considered, there was always a unique branch leading to all elements of $orb(X)$. □

This result can be easily extended if one wishes to branch on a given inequality featuring only variables of a given row i, *i.e.* a branching disjunction of the form:

$$\sum_{j=1}^{p} a^i x_{i,j} \leq k \quad \vee \quad \sum_{j=1}^{p} a^i x_{i,j} > k.$$

4 Application to the Unit Commitment Problem

Given a discrete time horizon $\mathcal{T} = \{1, ..., T\}$, a demand for electric power D_t is to be met at each time period $t \in \mathcal{T}$. Power is provided by a set $\mathcal{N}$ of n production units. At each time period, unit $i \in \mathcal{N}$ is either down or up, and in the latter case, its production is within $[P^i_{min}, P^i_{max}]$. Each unit must satisfy minimum up-time (resp. down-time) constraints, *i.e.* each unit i must remain up (resp. down) during at least L^i (resp. ℓ^i) periods after start up (resp. shut down). Each unit i has *initial conditions* (e^i, τ^i), indicating unit i has commitment status e^i before time 1, and it should remain in the same status up to time τ^i (if $\tau^i = 0$, unit i is free to shut down or start up at time 1). Each unit i also features three different costs: a fixed cost c^i_f, incurred each time period the unit is up; a start-up cost c^i_0, incurred each time the unit starts up; and a cost c^i_p proportional to its production. The Min-up/min-down Unit Commitment Problem (MUCP) [1] is to find a production plan minimizing the total cost while satisfying the demand and the minimum up and down time constraints. The MUCP is strongly NP-hard [2].

For each unit $i \in \mathcal{N}$ and time period $t \in \mathcal{T}$, let us consider three variables: $x^i_t \in \{0, 1\}$ indicates if unit i is up at time t; $u^i_t \in \{0, 1\}$ indicates whether unit i starts up at time t; and $p^i_t \in \mathbb{R}$ is the quantity of power produced by unit i at time t. Without loss of generality we consider that $L^i, \ell^i \leq T$. The MUCP can be formulated as follows:

$$\min_{x,u,p} \quad \sum_{i=1}^{n} \sum_{t=1}^{T} c^i_f x^i_t + c^i_p p^i_t + c^i_0 u^i_t$$

$$\text{s. t.} \quad \sum_{t'=t-L^i+1}^{t} u^i_{t'} \leq x^i_t \qquad \forall i \in \mathcal{N},\ \forall t \in \{L^i, ..., T\} \tag{5}$$

$$\sum_{t'=t-\ell^i+1}^{t} u^i_{t'} \leq 1 - x^i_{t-\ell^i} \quad \forall i \in \mathcal{N},\ \forall t \in \{\ell^i, ..., T\} \tag{6}$$

$$u^i_t \geq x^i_t - x^i_{t-1} \qquad \forall i \in \mathcal{N},\ \forall t \in \{2, ..., T\} \tag{7}$$

$$\sum_{i=1}^{n} p^i_t \geq D_t \qquad \forall t \in \mathcal{T} \tag{8}$$

$$P^i_{min} x^i_t \leq p^i_t \leq P^i_{max} x^i_t \qquad \forall i \in \mathcal{N},\ \forall t \in \mathcal{T} \tag{9}$$

$$x^i_t, u^i_t \in \{0, 1\} \qquad \forall i \in \mathcal{N},\ \forall t \in \mathcal{T} \tag{10}$$

Symmetries in the MUCP (and in the UCP) arise from the existence of groups of identical units. Suppose there are H different types of units and n_h units of type h, $h \in \{1, ..., H\}$. The set of type h units is denoted by $\mathcal{N}_h$.

The solutions of the MUCP can be expressed as a series of 0/1 matrices. We introduce matrix $X^h \in \mathcal{P}(T, n_h)$ such that entry $X^h_{i,t}$ corresponds to variable x^i_t of the i^{th} unit of type h. Column i of matrix X^h corresponds to the up/down planning of the i^{th} unit of type h. Similarly, we introduce matrices U^h and P^h. Since all units of type h are identical, their production plans can be permuted, provided that the same permutation is applied to matrices X^h, U^h and P^h. Thus, the symmetry group $\mathcal{G}$ contains all column permutations applied to X, U and P. As binary variables U are uniquely determined by variables X, breaking the symmetry of the X variables will break the symmetry over U variables.

4.1 State of the art: modified orbital branching

The authors in [14] apply modified orbital branching alongside with various additional branching rules to break symmetries on the X variables of another variant of the UCP. Experimental results compare different approaches:

- *Default* Cplex
- *Branch & Cut Cplex*, *i.e* Cplex with advanced features turned off, mimicking what happens when callbacks are used
- OB, *i.e.* orbital branching
- MOB with no branching rules enforced (Cplex is free to choose the next branching variable)
- MOB with RMRI, the most flexible branching rule ensuring that only representatives are enumerated.

It is shown that MOB with RMRI is more efficient than MOB, OB and Branch & Cut Cplex in terms of CPU time. The difference between using MOB with RMRI and MOB alone is however not as significant as the difference between MOB and simple orbital branching. Because advanced Cplex features are turned off when callbacks are used, there is still a huge performance gap between Branch & Cut Cplex and default Cplex.

4.2 Orbitopal fixing applied to the MUCP

As the production plans of identical units can be permuted, each variable matrix X^h can be restricted to be in the full orbitope $\mathcal{P}_0(T, n_h)$. The fixing strategies developed in Sections 3.2 and 3.3 can then be applied to fix variables in each matrix X^h, in order to enumerate only solutions with lexicographically non-increasing X.

More precisely, the two possible approaches are:

- Static orbitopal fixing, where the order on the lines is decided before the branching process. In this case, the branching decisions remain completely free
- Dynamic orbitopal fixing, where the order on the lines is decided during the branching process. In this case, the branching disjunctions must be of the form $(x_{i,t} = 0 \ \vee \ x_{i,t} = 1)$, or more generally, featuring only one line i of matrix X.

As in the context of the UCP it may be useful to branch on U variables, the next section describes how dynamic orbitopal fixing can be adapted to branch on more general types of disjunctions.

4.3 Dynamic orbitopal fixing and branching on u variables

One might want to be able to branch on the start-up variables u as well when using dynamic orbitopal fixing, or in terms of X variables, to branch on a disjunction of the form:

$$x_t^i - x_{t-1}^i \leq 0 \quad \vee \quad x_t^i - x_{t-1}^i = 1$$

Actually, it is possible to slightly adapt lexicographical ordering ϕ, so that dynamic fixing remains valid when at each node, the branching disjunction has one of the following forms:

$$x_t^i = 0 \quad \vee \quad x_t^i = 1$$

$$u_t^i = 0 \quad \vee \quad u_t^i = 1.$$

We thus define lexicographical ordering $\widetilde{\phi}$. For a given node a, if the branching disjunction at a is

$$x_{t_a}^{i_a} = 0 \quad \vee \quad x_{t_a}^{i_a} = 1,$$

or if $t_a = 1$, then subset $\widetilde{T}_a \subset \mathcal{T}$, integer $\widetilde{e}_a \in \mathbb{N}$ and bijection $\widetilde{\phi}_a : \{1, ..., e_a\} \to T_a$ are defined as T_a, e_a and ϕ_a in Section 3.3.

If the branching disjunction at a is

$$u_{t_a}^{i_a} = 0 \quad \vee \quad u_{t_a}^{i_a} = 1,$$

then we define subset $\widetilde{T}_a \subset \mathcal{T}$, integer $\widetilde{e}_a \in \mathbb{N}$ and bijection $\widetilde{\phi}_a : \{1, ..., \widetilde{e}_a\} \to T_a$ as follows.

If a is the root node, then

$$\begin{cases} \widetilde{T}_a = \{t_a,\ t_a - 1\} \\ \widetilde{e}_a = 2 \\ \widetilde{\phi}_a(1) = t_a \\ \widetilde{\phi}_a(2) = t_a - 1 \end{cases}$$

Otherwise, let b be the father node of a and:

$$\begin{cases} \widetilde{T}_a = \widetilde{T}_b \cup \{t_a,\ t_a - 1\} & \\ \widetilde{e}_a = |\widetilde{T}_a| & \\ \forall i \in \{1, ..., \widetilde{e}_b\}, & \widetilde{\phi}_a(i) = \widetilde{\phi}_b(i) \\ \text{If } t_a \notin \widetilde{T}_b, & \widetilde{\phi}_a(t_a) = \widetilde{e}_b + 1 \\ \text{If } t_a - 1 \notin \widetilde{T}_b, & \widetilde{\phi}_a(t_a - 1) = \widetilde{e}_a \end{cases}$$

Thus, at each node a, dynamic orbitopal fixing is performed on lines $\widetilde{T}_a = \{\ \widetilde{\phi}_a(1),\ \widetilde{\phi}_a(2),\ ...,\ \widetilde{\phi}_a(\widetilde{e}_a)\ \}$, with respect to lexicographical ordering $\widetilde{\phi}_a$.

Let τ be a branch & bound tree of MILP (UCP), in which dynamic orbitopal fixing is used and at each node, the branching disjunction has one of the following forms:

$$x_t^i = 0 \quad \vee \quad x_t^i = 1$$

$$u_t^i = 0 \quad \vee \quad u_t^i = 1.$$

Theorem 5. *For each feasible solution X to MILP (UCP), there is exactly one solution $X' \in orb(X)$ enumerated in tree τ.*

Proof. The proof can be derived as in Theorem 4: it suffices to see that at each node considered from the root, there is exactly one branch which contains all the elements of $orb(X)$ enumerated in tree τ. □

This result can be easily extended if one wishes to branch on an arbitrary disjunction, if at each node a, any time period featured in the disjunction is ordered by $\widetilde{\phi}_a$.

4.4 Numerical experiments

All experiments were performed using one thread of a PC with a 64 bits Intel Core i7-6700 processor running at 3.4GHz, and 32 GB of RAM memory. The problems are solved until optimality (defined within 10^{-6} of relative optimality tolerance) or until the time limit of 3600 seconds is reached.

4.4.1 Instances

In order to determine which symmetry-breaking method performs best with respect to the number of rows and columns of matrix X, we consider various couples (n, T), namely: (60, 48), (30, 96), (20, 192) and (15, 288).

For each couple (n, T), we generate a set of UCP instances as follows.

For each instance, we generate a "2-peak per day" type demand with a large variation between peak and off-peak values: during one day, the typical demand in energy during one day has two peak periods, one in the morning and one in the evening. The amplitudes between peak and off-peak periods have similar characteristics to those in the dataset from [3].

For all instances we randomly generate initial conditions e^i and set $\tau^i = 0$, for each unit i. We consider the parameters (P_{min}, P_{max}, L, ℓ, c_f, c_0, c_p) of each unit from the dataset presented in [3]. We draw a correlation matrix between these characteristics and define a possible range for each characteristic. In order to introduce symmetries in our instances, some units are randomly generated based on the parameters correlations and ranges. Each unit generated is duplicated d times, where d is randomly selected in $[1, \frac{n}{F}]$ in order to obtain a total of n units. The parameter F is called symmetry factor, and can vary from 2 to 4 depending on the value of n.

Table 1 provides some statistics on the instances. For each instance, a group is a set of two or more units with same characteristics. Each unit which has not been duplicated is a singleton. The first and second entries column-wise are the number of singletons and groups. The third entry is the mean group size and the fourth entry is the maximum group size. Each entry row-wise corresponds to the average value obtained over the 20 instances with same size (n, T) and same symmetry factor F.

Size	Sym. factor	Nb of singletons	Nb of groups	Mean size of groups	Group max size
(15,288)	F = 3	1.5	4.2	3.27583	4.55
	F = 2	0.9	3.45	4.23167	6.15
(20,192)	F = 4	2.15	5.35	3.40726	4.75
	F = 3	1.35	5.1	3.7875	5.5
	F = 2	0.65	3.7	5.38333	8.05
(30,96)	F = 4	1.3	6.45	4.51718	6.7
	F = 3	0.4	5.25	5.96786	8.65
	F = 2	0.55	4.05	7.58667	11.4
(60,48)	F = 4	0.8	7.7	7.86369	13.15
	F = 3	0.55	5.8	10.8604	17.8
	F = 2	0.2	4.75	13.9317	23.75

Table 1: Instances

In the following experiments, we compare various resolution methods. The experimental results show there is an important variability in the computation time within groups of instances with

same size (n, T) and same symmetry factor F. We thus introduce the CPU time improvement score, which is a performance ratio comparing the CPU times of two methods. The *improvement score* for two given methods m_1 and m_2 is defined as:

$$I_{CPU}(m_1, m_2) = 2\frac{CPU(m_2)-CPU(m_1)}{CPU(m_1)+CPU(m_2)}$$

We similarly define the node improvement score and the gap improvement score:

$$I_{nodes}(m_1, m_2) = 2\frac{nodes(m_2)-nodes(m_1)}{nodes(m_1)+nodes(m_2)}$$

$$I_{gap}(m_1, m_2) = 2\frac{gap(m_2)-gap(m_1)}{gap(m_1)+gap(m_2)}$$

For any indicator ind and any two methods m_1 and m_2, the considered improvement score $I_{ind}(m_1, m_2)$ provides a symmetric comparison between the two methods m_1 and m_2. Indeed, the improvement score is a performance ratio, where the reference used is the average of the indicator values obtained from m_1 and m_2. Using this average value as reference yields the following key property: $I_{ind}(m_1, m_2) = -I_{ind}(m_2, m_1)$. In particular, $I_{ind}(m_1, m_2) \in [-2, 2]$, while the standard relative error calculated as $\frac{ind(m_1)-ind(m_2)}{ind(m_1)} \in [-\infty, 1]$ would be non-symmetric and unbounded.

4.4.2 Static orbitopal fixing vs dynamic orbitopal fixing

The average improvement score between static orbitopal fixing (SOF) and dynamic orbitopal fixing (DOF) in given in Table 2, for each group of 20 instances of same size and symmetry factor. It is clear that DOF outperforms SOF on each group.

(15, 288)		((20, 192)			(30, 96)			(60, 48)		
$F = 3$	$F = 2$	$F = 4$	$F = 3$	$F = 2$	$F = 4$	$F = 3$	$F = 2$	$F = 4$	$F = 3$	$F = 2$
72.7 %	102 %	79.3 %	116 %	65.1 %	52.7 %	84 %	52.5 %	50.3 %	43.3 %	39.5 %

Table 2: $I_{CPU}(SOF, DOF)$

4.4.3 Comparison of Cplex, MOB and DOF

We compare four different resolution methods for the MUCP:

- *Default* Cplex (DC)
- *Callback Cplex* (CC) , *i.e* Cplex with an empty branch callback
- MOB with no branching rules enforced (Cplex is free to choose the next branching variable)
- Dynamic orbitopal fixing (DOF)

On instances from [14], the average CPU time improvement score between MOB and MOB+RMRI is 9%. Even though MOB+RMRI is slightly better than MOB with no branching rules, we choose to compare our methods to MOB, because its implementation is straightforward, thus leaving no room to interpretation.

Table 3 compares MOB and DOF to Callback Cplex and Default Cplex, with respect to their average gap, node and CPU time improvement scores on groups of 20 instances of same size (n, T)

and same symmetry factor F. For each method m_1, Table 3 indicates N, the number of instances solved to optimality by m_1, and compares m_1 to another method m_2 with respect to:

N_Δ = number of instances solved by m_1 - number of instances solved by m_2

I_{gap}: average gap improvement score, computed for each instance on which neither m_1 nor m_2 reaches optimality.

I_{nodes}: average node improvement score, computed for each instance on which both m_1 and m_2 reached optimality.

I_{CPU}: average CPU time improvement score, computed on every instance.

Instance				$m2$ = Default Cplex				m_2 = Callback Cplex			
(n,T)	Sym	m_1	N	N_Δ	I_{gap}	I_{nodes}	I_{cpu}	N_Δ	I_{gap}	I_n	I_{cpu}
(15,288)	$F=3$	MOB	13	-6	-134 %	-87.4 %	-88.4 %	-1	14.7 %	0.15 %	4.73 %
		DOF	15	-4	-171 %	-23.6 %	-42.4 %	1	72.4 %	96.4 %	63.3 %
	$F=2$	MOB	17	-3	0 %	-61.1 %	-57.5 %	0	54.3 %	43.7 %	15.2 %
		DOF	18	-2	0 %	-21.5 %	-30.4 %	1	61.8 %	113 %	57.9 %
(20,192)	$F=4$	MOB	17	-2	-153 %	-123 %	-111 %	1	-37.4 %	64.2 %	20.3 %
		DOF	19	0	-144 %	-77.2 %	-40.7 %	3	29.8 %	132 %	102 %
	$F=3$	MOB	13	-5	-183 %	-62.5 %	-96.8 %	1	29.1 %	69.3 %	32.9 %
		DOF	14	-4	-185 %	-68.7 %	-79.5 %	2	59.9 %	100 %	58.9 %
	$F=2$	MOB	12	-6	-82.3 %	-31.5 %	-75.9 %	0	4.84 %	36.1 %	27.3 %
		DOF	13	-5	-61.7 %	0.00326 %	-55.6 %	1	43.3 %	104 %	52.2 %
(30,96)	$F=4$	MOB	14	-6	0 %	-72.3 %	-107 %	2	-3.64 %	48.1 %	16.5 %
		DOF	19	-1	0 %	-38.3 %	-18.5 %	7	14.1 %	143 %	113 %
	$F=3$	MOB	12	-4	-124 %	-34.2 %	-51.1 %	4	16.2 %	149 %	69 %
		DOF	13	-3	-90 %	-58.3 %	-47.8 %	5	82.6 %	121 %	84.8 %
	$F=2$	MOB	12	-5	-46.3 %	-12.7 %	-63.2 %	1	85.4 %	113 %	45.8 %
		DOF	14	-3	-10.5 %	58.5 %	-15.1 %	3	125 %	138 %	89.6 %
(60,48)	$F=4$	MOB	17	0	47.7 %	66.7 %	15 %	9	143 %	91.1 %	105 %
		DOF	16	-1	-7.35 %	65.5 %	28.7 %	8	103 %	106 %	110 %
	$F=3$	MOB	13	0	1.62 %	42.6 %	14.3 %	6	49.9 %	114 %	80.2 %
		DOF	15	2	-3.55 %	89.4 %	48.9 %	8	99.3 %	120 %	110 %
	$F=2$	MOB	17	0	44.1 %	-17.9 %	7.57 %	6	44.5 %	97.7 %	89.4 %
		DOF	19	2	0 %	14.1 %	61.4 %	8	195 %	75.6 %	112 %

Table 3: MOB and dynamic orbitopal fixing - average improvement scores for various instances

In terms of CPU time, both MOB and DOF greatly outperform B&C Cplex, but the improvement is much larger with DOF.

As observed in [14], there is a huge performance gap between B&C Cplex and Default Cplex. In particular, on instances with $n \leq 30$, Default Cplex manages symmetries very efficiently. Thus, even if DOF and MOB greatly improve B&C Cplex, it is usually not enough to close this performance gap. Globally, even if it does not catch up Default Cplex, the performance loss is less important with DOF than with MOB.

When n increases ($n = 60$), Default Cplex sometimes struggles to reach optimality within the time limit. On the opposite, DOF still performs well in this context and solves to optimality some instances Default Cplex cannot. Thus, in case $n = 60$, DOF outperforms both B&C Cplex and Default Cplex. MOB does not improve Default Cplex as much as DOF.

Table 4 provides average number of nodes, number of fixings (for MOB, it is the total number of variables fixed during the branching process) and CPU time for each group of 20 instances.

Instances		method	nodes	number of fixings	cpu time
(15,288)	$F = 3$	DC	72723.7	0	237.434
		CC	278722	0	1166.08
		MOB	259414	5374.2	1353.14
		DOF	215412	57826.9	923.255
	$F = 2$	DC	16480.5	0	20.9985
		CC	207097	0	576.868
		MOB	218786	1464.35	583.012
		DOF	138925	40262.1	418.916
(20,192)	$F = 4$	DC	24734.8	0	197.672
		CC	386446	0	978.943
		MOB	163565	9150.55	706.585
		DOF	121659	23574.9	347.097
	$F = 3$	DC	141456	0	392.782
		CC	509161	0	1624.79
		MOB	342974	22621.8	1363.59
		DOF	374525	204167	1256.53
	$F = 2$	DC	70283.7	0	413.073
		CC	569132	0	1574.98
		MOB	310299	37422	1493.3
		DOF	446119	254436	1379.42
(30,96)	$F = 4$	DC	34741.5	0	35.242
		CC	1.23224e+06	0	1541.24
		MOB	734868	48313.3	1222.69
		DOF	312632	131181	353.692
	$F = 3$	DC	800230	0	879.658
		CC	1.47284e+06	0	2338.72
		MOB	683773	188132	1583.97
		DOF	782985	627884	1343.78
	$F = 2$	DC	350610	0	608.481
		CC	947007	0	1762.22
		MOB	904914	617226	1530.35
		DOF	531426	661735	1088.96
(60,48)	$F = 4$	DC	1.02977e+06	0	841.093
		CC	1.92451e+06	0	2286.65
		MOB	326837	194657	647.311
		DOF	628777	656787	767.224
	$F = 3$	DC	1.25776e+06	0	1284.29
		CC	1.57958e+06	0	2386.88
		MOB	872257	730178	1321.21
		DOF	448037	911962	923.128
	$F = 2$	DC	589062	0	580.998
		CC	1.21218e+06	0	1805.79
		MOB	436945	50496	590.528
		DOF	166956	336218	198.01

Table 4: Average number of nodes, fixing and cpu time values for each method and each instance group

Conclusion

In this paper, we define a linear time orbitopal fixing algorithm for the full orbitope. This algorithm is proven to be optimal, in the sense that at any node a in the search tree, any variable that can be fixed (i.e. a variable set to the same value in all lexicomax feasible solutions to subproblem a) is fixed by the algorithm. From the simple observation that when more variables on small-index rows are already fixed at node a, more fixings can be performed, we propose a dynamic version of the orbitopal fixing algorithm, where the lexicographical order at node a is defined with respect to the branching decisions leading to a. For MUCP instances, our experimental results show that the proposed approach is competitive with state of the art methods like modified orbital branching.

In the past, the complete linear description of partitioning and packing orbitopes helped to design an orbitopal fixing algorithm for these orbitopes. Likewise in the future, maybe the orbitopal fixing algorithm for the full orbitope, by improving our understanding of this polyhedron, might help to find a complete linear description of the full orbitope. Moreover, it would be interesting to extend orbitopal fixing to full orbitopes under other group actions, for example the cyclic group.

Finally, there is a wide range of problems featuring all column permutation symmetries, in particular many variants of the UCP, on which it would be desirable to analyze the effectiveness of our approach. Other examples of such problems can be found among covering problems, whose solution matrix has at least one 1-entry per row. Even though computing the exact fixing has been shown NP-hard in this case, our orbitopal fixing algorithm, designed for full orbitopes, can be used to compute fixings in a covering orbitope. In this case, there is no guarantee that fixings are done as early as possible in the tree, because the special structure of covering orbitopes may induce possible fixings that would not be correct in a full orbitope. Nevertheless, this fixing algorithm breaks all column-permutation related symmetries at some point in the branching tree, which may be sufficient to overcome the computational difficulties arising from the highly symmetric nature of these problems.

References

[1] P. Bendotti, P. Fouilhoux, and C. Rottner. The min-up/min-down unit commitment polytope. *Optimization Online*, 2016. `http://www.optimization-online.org/DB_HTML/2016/12/5750.html`.

[2] P. Bendotti, P. Fouilhoux, and C. Rottner. Complexity of the min-up/min-down unit commitment problem. *Optimization Online*, 2017. `http://www.optimization-online.org/DB_HTML/2017/06/6061.html`.

[3] M. Carrion and J. M. Arroyo. A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem. *IEEE Transactions on Power Systems*, 21, 2006.

[4] E. J. Friedman. *Fundamental Domains for Integer Programs with Symmetries*, pages 146–153. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[5] L C Grove and C T Benson. *Finite reflection groups*, volume 99. Springer Science & Business Media, 1996.

[6] C. Hojny and M. E. Pfetsch. Polytopes associated with symmetry handling. *Optimization Online*, 2017. `http://www.optimization-online.org/DB_HTML/2017/01/5835.html`.

[7] V. Kaibel and A. Loos. Branched polyhedral systems. In *Proceedings of the 14th International IPCO Conference on Integer Programming and Combinatorial Optimization*. Springer-Verlag, 2010.

[8] V. Kaibel, M. Peinhardt, and M. E. Pfetsch. Orbitopal fixing. *Discrete Optimization*, 8(4):595 – 610, 2011.

[9] V. Kaibel and M. Pfetsch. Packing and partitioning orbitopes. *Mathematical Programming*, 114(1):1 – 36, 2008.

[10] A. Loos. *Describing Orbitopes by Linear Inequalities and Projection Based Tools*. PhD thesis, Universität Magdeburg, 2011.

[11] F. Margot. Pruning by isomorphism in Branch-and-Cut. In *Proceedings of the 8th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 304–317, London, UK, 2001. Springer-Verlag.

[12] F. Margot. Exploiting orbits in symmetric ILP. *Mathematical Programming*, 98(1):3–21, 2003.

[13] F. Margot. *Symmetry in Integer Linear Programming*, pages 647–686. Springer, Berlin, Heidelberg, 2010.

[14] J. Ostrowski, M.F. Anjos, and A. Vannelli. Modified orbital branching for structured symmetry with an application to unit commitment. *Mathematical Programming*, 150(1):99 – 129, 2015.

[15] J. Ostrowski, J. Linderoth, F. Rossi, and S. Smriglio. *Constraint Orbital Branching*, pages 225–239. Springer Berlin Heidelberg, 2008.

[16] J. Ostrowski, J. Linderoth, F. Rossi, and S. Smriglio. Orbital branching. *Mathematical Programming*, 126(1):147–178, 2011.

[17] C. C. Sims. Computational methods in the study of permutation groups. In *Computational Problems in Abstract Algebra*, pages 169 – 183. Pergamon, Oxford, 1970.