Automatic Generation of Symmetry-Breaking Constraints

Leo Liberti

LIX, École Polytechnique, F-91128 Palaiseau, France
liberti@lix.polytechnique.fr

Abstract. Solution symmetries in integer linear programs often yield long Branch-and-Bound based solution processes. We propose a method for finding elements of the permutation group of solution symmetries, and two different types of symmetry-breaking constraints to eliminate these symmetries at the modelling level. We discuss some preliminary computational results.

1 Introduction

We consider a Mixed Integer Linear Program (MILP) in the following form:

$$\left.\begin{array}{rl} \min c^\top x & \\ Ax & \leq b \\ x & \in [x^L, x^U] \\ \forall i \in Z \quad x_i & \in \mathbb{Z}. \end{array}\right\} \tag{1}$$

where $c, x, x^L, x^U \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, A is a real $m \times n$ matrix and $Z \subseteq \{1, \ldots, n\}$. Throughout the paper, elements of groups are represented by means of permutations of either the column or the row space; permutations on the row space are denoted by left multiplication, and permutations on the column space by right multiplication. Because a solution x of (1) has as many elements as the columns of A, a permutation π on x is likened to a permutation of the column space, and hence denoted by right multiplication $x\pi$.

Problems (1) having many symmetries are known to be very difficult to solve to global optimality with Branch-and-Bound (BB) techniques. These converge slowly in presence of symmetries because many leaf nodes in the BB tree may contain (symmetric) global optima: hence, no node in the paths leading from the root to these leaf nodes can ever be pruned. Despite the practical difficulties given by solution symmetries, there are relatively few group theory based methods in mathematical programming. These may be classified in three broad categories: (a) the abelian group approach proposed by Gomory to writing integer feasibility conditions for x; (b) symmetry-breaking techniques for specific problems, whose symmetry group can be computed in advance; (c) general-purpose symmetry group computations and symmetry-breaking techniques implemented via branching strategies and local cuts in a typical BB solution algorithm.

Category (a) was established by R. Gomory [7]: given a basis B of the constraint matrix A, it considers the (abelian) group $\mathscr{G} = \mathbb{Z}^n / \langle \mathrm{col}(B) \rangle$, where $\mathbb{Z}^n$ is

B. Yang, D.-Z. Du, and C.A. Wang (Eds.): COCOA 2008, LNCS 5165, pp. 328–338, 2008.

the additive group of integer n-sequences and $\langle \mathrm{col}(B) \rangle$ is the additive group generated by the columns of the (nonsingular) matrix B. We consider the natural group homomorphism $\varphi : \mathbb{Z}^n \to \mathscr{G}$ with $\ker \varphi = \langle \mathrm{col}(B) \rangle$: letting (x_B, x_N) be a basic/nonbasic partition of the decision variables, we apply φ to the standard form constraints $Bx_B + Nx_N = b$ to obtain $\varphi(Bx_B) + \varphi(Nx_N) = \varphi(b)$. Since $\varphi(Bx_B) = 0$ if and only if $x_B \in \mathbb{Z}^n$, setting $\varphi(Nx_N) = \varphi(b)$ is a necessary and sufficient condition for x_B to be integer feasible. Gomory's seminal paper gave rise to further research, among which [22,1]

Category (b) is possibly the richest in terms of number of published papers. Many types of combinatorial problems exhibit a certain amount of symmetry. Symmetries are usually broken by means of specific branching techniques (e.g. [16]), appropriate global cuts (e.g. [21]) or special formulations [11,2] based on the problem structure. The main limitation of the methods in this category is that they are difficult to generalize and/or to be rendered automatic.

Category (c) contains two main research streams. The first was established by F. Margot in the early 2000s [14,15], and is applicable to problems in general form (1) where $x^L = 0, x^U = 1$, i.e. Binary Linear Programs (BLPs). Margot defines the *symmetry group* of a BLP as:

$$\{\pi \in S_n \mid c^\top \pi = c^\top \wedge \exists \sigma \in S_n \ (\sigma b = b \wedge \sigma A \pi = A)\}, \tag{2}$$

or, in other words, all relabellings of problem variables for which the objective function and constraints are the same. The symmetry group (2) is used to derive effective BB pruning strategies by means of isomorphism pruning and isomorphism cuts local to some selected BB tree nodes (Margot extended his work to general integer variables in [17]). Stronger results of the same type can be obtained for covering and packing problems [20], for these have an objective function vector $c = (1, \ldots, 1)$ and a RHS vector $b = (1, \ldots, 1)$ fixed by all elements of S_n and S_m respectively, and their constraint matrix is 0-1.

The second was established by V. Kaibel and M. Pfetsch in 2007 [9]. Symmetries in the column space (i.e. permutations of decision variables) of binary ILPs having 0-1 constraint matrices are shown to affect the quality of the linear programming bound. Limited only to permutations in cyclic and symmetric group, complete descriptions of *orbitopes* are provided by means of linear inequalities. Let x' be a point in $\{0,1\}^n$ (the solution space), with $n = pq$, so that we can arrange the components of x' in a matrix C. Given a group G and $\pi \in G$, for all 0-1 $p \times q$ matrices C let $C\pi$ be the matrix obtained by permuting the columns of C according to π. Let $G \cdot C$ be the orbit of C under the action of all $\pi \in G$, $\overline{G \cdot C}$ be the lexicographically maximal matrix in $G \cdot C$ (ordering matrices by rows first) and $\mathcal{M}^{\max}_{pq}(G)$ be the set of all $\overline{G \cdot C}$. Then the *full orbitope* associated with G is $\mathrm{conv}(\mathcal{M}^{\max}_{pq}(G))$. Inspired by the work on orbitopes, E. Friedman very recently proposed a similar but extended approach leading to *fundamental domains* [5]: given a feasible polytope $X \subseteq [0,1]^n$ with integral extreme points and a group G acting as an affine transformation on X (i.e. for all $\pi \in G$ there is a matrix $A \in GL(n)$ and an n-vector d such that $\pi x = Ax + d$ for all $x \in X$), a fundamental domain is a subset $F \subset X$ such that $GF = X$.

The Constraint Programming (CP) community is also concerned with symmetries and some of the results in the CP literature can be extended to mathematical programming (see [3] for a good introduction).

The present work belongs to category (c): it proposes general-purpose methods for identifying (some) solution symmetries and restrict the feasible region so that it does not contain all representatives per equivalence class. This paper contributes two ideas: (i) breaking the same type of symmetries described in [16,14,15,17] at the modelling instead of the algorithmic level (whereas Margot proposes symmetry breaking methods local to each node of the BB tree, we discuss global symmetry breaking constraints which can be added to the original formulation); and (ii) computing symmetries automatically instead of assuming them as given. In Section 2 we propose symmetry breaking constraints derived from cycles; Section 3 describes a mathematical program whose solution encodes a permutation of the symmetry group; in Section 4 we discuss some practical strategies for exploiting the proposed symmetry breaking constraints and some preliminary computational results.

1.1 Notation

For a group $G \leq S_n$ and a set X of row vectors, $XG = \{xg \mid x \in X \wedge g \in G\}$; if Y is a set of column vectors, $GY = \{gy \mid y \in Y \wedge g \in G\}$. If $X = \{x\}$, we denote XG by xG (and similarly for Y). For a mathematical program P we let $\mathcal{F}(P)$ be the feasible region of P and $\mathcal{G}(P)$ be the set of (global) optima of P. For $x \in \mathbb{R}^n$, we let $\mathrm{ran}(x) = \{a \in \mathbb{R} \mid \exists j \leq n\ (x_j = a)\}$ be the range of x. All groups considered in this paper are finite.

2 Theoretical Results

2.1 Efficiency of Symmetry Breaking Constraints

We let S_n be the symmetric group of order $n \in \mathbb{N}$. For a set $X \subseteq \mathbb{R}^n$, a group $G \leq S_n$ and $x, y \in X$, we define an equivalence relation $x \sim_G y \Leftrightarrow \exists \pi \in G\ (x\pi = y)$. The relation $\sim_G$ partitions X into a set $\mathcal{E}(G, X)$ of equivalence classes (each of finite cardinality) such that $X = \bigcup_{Y \in \mathcal{E}(G,X)} Y$ (the cardinality of $\mathcal{E}(G, X)$ itself need not be finite or even countable).

Definition 2.1
A linear constraint $dx \leq d_0$ with $(d, d_0) \in \mathbb{R}^{n+1}$ is symmetry breaking with respect to G and X if for all $Y \in \mathcal{E}(G, X)$ there are $\bar{x} \neq \bar{y} \in Y$ s.t. $d\bar{x} \leq d_0$ and $d\bar{y} > d_0$. The constraint is symmetry breaking of order ℓ if there is at least one equivalence class $Y \in \mathcal{E}(G, X)$ in which there are exactly $\ell - 1$ points y s.t. $d\bar{y} > d_0$. The constraint is maximally symmetry breaking if there is at least one equivalence class $Y \in \mathcal{E}(G, X)$ for which it is symmetry breaking of order $|Y|$.

Defn. 2.1 can easily be extended to systems of constraints. Supposing $X \subseteq \mathbb{Z}^n$, symmetry breaking constraints are not in general valid cuts for any linear polyhedron containing X, because they may also cut off some integral points. However, they guarantee feasibility of at least one integral point per equivalence

class. Adding appropriate symmetry breaking constraints to P results in a reformulation of the *narrowing* type [13], i.e. a reformulation Q of P with a map $\psi : \mathcal{F}(Q) \to \mathcal{F}(P)$ such that $\psi(\mathcal{G}(Q)) \subseteq \mathcal{G}(P)$ [12]. Notice that symmetry breaking constraints of order $\ell \leq 1$ do not break any symmetry at all, as they do not separate any point in any equivalence class of $\mathcal{E}(G, X)$.

2.2 Symmetry Groups Associated to a MILP

We consider symmetries that leave various properties of P invariant.

Definition 2.2
The set

$$G^* = \{\pi \in S_n \mid \forall x \in \mathcal{G}(P)\ (x\pi \in \mathcal{G}(P))\} \tag{3}$$

of automorphisms of $\mathcal{G}(P)$ is called the solution symmetry group of P. The set

$$\tilde{G} = \{\pi \in S_n \mid \forall x \in \mathcal{F}(P)\ (x\pi \in \mathcal{F}(P))\} \tag{4}$$

of automorphisms of $\mathcal{F}(P)$ is called the feasible symmetry group of P

It is easy to show that $\tilde{G}, G^*$ are both subgroups of S_n, and that $G^* \leq \tilde{G}$. Next, we extend (2) to formulation (1).

Definition 2.3
The set

$$\begin{aligned} G_P = \{\pi \in S_n \mid \forall a \in \{c, x^L, x^U\} \\ (a^\top \pi = a^\top) \wedge Z\pi = Z \wedge \exists \sigma \in S_n\ (\sigma b = b \wedge \sigma A\pi = A)\} \end{aligned} \tag{5}$$

of permutations that fix the problem formulation is called the problem symmetry group of P.

It is equally easy to show that G_P is a subgroup of S_n. The following useful result states that problem symmetries are solution symmetries.

Proposition 2.4
$G_P \leq G^*$.

2.3 Symmetry Breaking Constraints from Disjoint Cycles

Let R be the relaxation of P obtained by removing the constraints $Ax \leq b$, $\bar{X} = \mathcal{F}(R)$ and $X^* = \mathcal{G}(R)$. Let $\sigma = (\sigma_1, \ldots, \sigma_k)$ be a cycle of length $1 < k \leq n$ in S_n. For any $x \in \mathbb{R}^n$, let $x[\sigma] = (x_{\sigma_1}, \ldots, x_{\sigma_k})$, and assume that $x[\sigma]$ are constrained to be integer. Let $\bar{G} \leq S_n$ be the group of all permutation automorphisms of $\bar{X}$. If $\sigma \in \bar{G}$, for all $x \in \bar{X}$ and $j \leq k$ we have $x^L_{\sigma_1} \leq x[\sigma]_j \leq x^U_{\sigma_1}$, i.e. there is a unique number of values χ that all variables in $x[\sigma]$ can take. We let $\bar{\chi}$ be the

row vector whose j-th component is χ^{k-j} for all $j \leq k$. Consider the following constraints, often cited in the literature for symmetry breaking purposes [21,10]:

$$\forall 1 \leq h \leq k-1 \quad \bar{\chi}(x[\sigma] - x^L[\sigma]) \leq \bar{\chi}(x[\sigma] - x^L[\sigma])\sigma^h \tag{6}$$
$$\Rightarrow \sum_{j=1}^{k} \chi^{k-j}(x_{\sigma_j} - x^L_{\sigma_j}) \leq \sum_{j=1}^{k} \chi^{k-j}(x_{\sigma^h(\sigma_j)} - x^L_{\sigma^h(\sigma_j)})).$$

Proposition 2.5
Let $\sigma = (\sigma_1, \ldots, \sigma_k)$ be a cycle of length $k \leq n$ in $\bar{G}$. Then constraints (6) are maximally symmetry breaking w.r.t. $\langle\sigma\rangle$, $\bar{X}$.

The practical trouble with (6) is their well-known poor scaling, as the values of the coefficients are of different orders of magnitude.

Next, we consider some well-scaled (though less effective) symmetry breaking constraints.

Proposition 2.6
Let $\sigma = (\sigma_1, \ldots, \sigma_k)$ be a cycle of length $k \leq n$ in $\bar{G}$. For all $x \in \bar{X}$ with $|ran(x[\sigma])| = \ell$,

$$\forall 2 \leq j \leq k \quad x_{\sigma_1} \leq x_{\sigma_j} \tag{7}$$

are symmetry breaking constraints of order ℓ.

Notice that if $\ell = k$, then by Prop. 2.6 (7) are symmetry-breaking constraints of order k; furthermore, if $\ell = k$ the vector $x \in \mathbb{R}^n$ having distinct components $x[\sigma]$ gives rise to an equivalence class $x\langle\sigma\rangle$ of cardinality k, so (7) are maximally symmetry breaking. If $\ell = 1$ then (7) do not break any symmetry (remark after Defn. 2.1) but then again if $\ell = 1$ it means that $x[\sigma] = (a, \ldots, a)$ for some $a \in \mathbb{R}$, so $|x\langle\sigma\rangle| = 1$, which means there are no symmetric solutions complicating the solution process. The most likely case is that $x[\sigma] \in \{0,1\}^k$ and $\ell = 2$: this is unfortunate as this situation provides the weakest case of Prop. 2.6. We stress, however, that symmetry breaking constraint of order ℓ will cut away *at least* (not *exactly*) $\ell - 1$ symmetric solutions.

Example 2.7
Let $x = (0,1,1,1)$, and $\sigma = (1,2,3,4)$. Then $x\langle\sigma\rangle = \{(0,1,1,1), (1,0,1,1), (1,1,0,1), (1,1,1,0)\}$. Since $|ran(x)| = |\{0,1\}| = 2$, constraints (7) are symmetry breaking of order 2. However, exactly 3 elements of $x\langle\sigma\rangle$ are cut off by (7) (i.e. all elements of $x\langle\sigma\rangle \setminus \{x\}$). Taking $x = (0,0,0,1)$, on the other hand, results in (7) only cutting off $x\sigma = (1,0,0,0)$ according to Prop. 2.6.

The main insight given by Example 2.7 is that if we make the assumption that optimal solutions of binary problems will contain on average as many 0s as 1s on components indexed by σ, we can expect (7) to cut away $\lfloor k/2 \rfloor$ symmetric solutions even though $\ell = 2$. Another insight is that if we suspect optimal solutions to have a large number of components attaining small values of the range, we might want to change the $\leq$ relation in (7) to a $\geq$ relation to increase the number of cut-off symmetric solutions (modifying the inequality relation in (7) to $x_{\sigma_1} \geq x_{\sigma_j}$ only requires a trivial change to the proof).

3 Finding Symmetries

Let

$$\hat{G}_P = \{(\sigma,\pi) \in S_m \times S_n \mid \forall a \in \{c, x^L, x^U\} \\ (a^\top \pi = a^\top) \wedge Z\pi = Z \wedge (\sigma b = b \wedge \sigma A \pi = A)\}. \tag{8}$$

It is easy to see that the projection of $\hat{G}_P$ on the second component S_n is equal to G_P. For $q \in \mathbb{N}$, Let $\vartheta : S_q \to GL(q)$ be the regular (faithful) permutation matrix representation of elements of S_q, i.e. for $\pi \in S_q$, $\vartheta(\pi)$ is a doubly stochastic invertible matrix with entries in $\{0,1\}$, such that for any row vector $v \in \mathbb{R}^n$, $v\pi = v\theta(\pi)$. We can then write the condition of (8) in terms of products of vectors and matrices.

We consider decision variables: σ_{ih}, the (i,h)-th element of the matrix $\vartheta(\sigma)$ for all $i,h \leq m, \sigma \in S_m$; and π_{jk}, the (j,k)-th element of the matrix $\vartheta(\pi)$ for all $j,k \leq n, \pi \in S_n$. Let $z \in \{0,1\}^n$ be the indicator vector of Z, such that $z_j = 1 \Leftrightarrow j \in Z$, and let $\Gamma(P)$ be the set of binary values of σ, π defined by the following constraints:

$$\forall j \leq n \quad \sum_{k \leq n} x^L_k \pi_{jk} = x^L_j \quad \wedge \quad \sum_{k \leq n} x^U_k \pi_{jk} = x^U_j \tag{9}$$

$$\forall j \leq n \quad \sum_{k \leq n} c_k \pi_{jk} = c_j \quad \wedge \quad \sum_{k \leq n} z_k \pi_{jk} = z_k \tag{10}$$

$$\forall i \leq m, j \leq n \quad \sum_{h \leq m} \sigma_{ih} A_{hj} = \sum_{k \leq n} A_{ik} \pi_{kj} \tag{11}$$

$$\forall i \leq m \quad \sum_{h \leq m} \sigma_{ih} b_h = b_i \tag{12}$$

$$\forall j \leq n \quad \sum_{k \leq n} \pi_{kj} = 1 \quad \wedge \quad \sum_{k \leq n} \pi_{jk} = 1 \tag{13}$$

$$\forall i \leq m \quad \sum_{h \leq m} \sigma_{ih} = 1 \quad \wedge \quad \sum_{h \leq m} \sigma_{hi} = 1 \tag{14}$$

$$\forall j,k \leq n \quad \pi_{kj} \in \{0,1\} \tag{15}$$

$$\forall i,h \leq m \quad \sigma_{ih} \in \{0,1\}. \tag{16}$$

It is easy to show that $\Gamma(P) = \{(\vartheta(\sigma), \vartheta(\pi)) \mid (\sigma,\pi) \in \hat{G}_P\}$. In order to exclude the identity from $\Gamma(P)$, we also add the constraint:

$$\sum_{j \leq n} \pi_{jj} \leq n - 1 \tag{17}$$

Since by Sect. 4 we look for long cycles, we arbitrarily choose an index $j' \leq n$ which is likely to belong to a long cycle in some permutation of G_P (this choice should be based on the block structure of A) and minimize the following objective function, which ensures that we select a permutation moving j':

$$\min \pi_{j'j'}. \tag{18}$$

We call the problem of minimizing (18) subject to (9)-(17) the FEASIBLE PERMUTATION PROGRAM associated to P w.r.t. j', denoted by FPP(P, j').

Proposition 3.1
If FPP(P, j') is infeasible, then $G_P = \{e\}$.

Although solving the full FPP(P, j') may be more CPU-intensive than solving the original problem, various improvements based on the block structure of the constraints in $\Gamma(P)$ are possible. A promising one consists in solving relaxations of the FPP where (11) only fix certain rows or blocks, and verifying later than the solution is valid in the general problem. Computational experience shows that although the linear relaxation of the FPP may be fractional, solutions to the FPP are mostly found at the root node of the CPLEX [8] BB tree after cuts addition.

4 Practical Solution Strategies

Our strategy for solving (1) consists in seeking permutations of G_P having long cycles in their disjoint cycle representation, and add symmetry breaking constraints (6) or (7). In this section we restrict the discussion to the well-scaled constraints (7) but the same ideas can be (and were) applied to (6) too.

Ideally, we would like to be able to add symmetry-breaking constraints (7) for all disjoint cycles in all generators of G_P. This, however, may lead to infeasibility, as Example 4.1 shows.

Example 4.1
Suppose $\mathcal{G}(P) = \{(0,1,1,0),(1,0,0,1)\}$ and $G_P = \{e,(1,2)(3,4)\}$. Then both (6) and (7) would imply $x_1 \leq x_2$ and $x_3 \leq x_4$, which are satisfied by no point in $\mathcal{G}(P)$.

The trouble arises because for a cycle σ, constraints (7) arbitrarily decide that x_{σ_1} is the component of $x[\sigma]$ having minimum value. At the modelling level, this is similar to the main drawback of the algorithm proposed in [14]: "the branching variable cannot be chosen freely, but always has to be the non-fixed variable with smallest index" ([15], p. 3-4). However, since at the modelling level there is no knowledge of what variables are fixed, (7) can only be imposed for one single cycle. A promising strategy is that of selecting the longest cycle σ from the set of all disjoint cycles in all permutations of G_P. In general, we can change the arbitrary choice of minimum component for any $i \leq |\sigma|$ (the cycle length), and we denote BREAKSYMM2(P, i) the reformulated problem P with (7) adapted to σ and i as added constraints. It is easy to show that BREAKSYMM2(P, i) is a valid narrowing for all cycles σ and $i \leq |\sigma|$, albeit one that is still subject to an arbitrary choice. We circumvent this by introducing continuous variables $y_i^\sigma \geq 0$ whose value is exactly 0 only if x_{σ_i} is the minimum element of σ, and reformulating (7) as follows:

$$\forall i, j \leq |\sigma|, j \neq i \quad x_{\sigma_i} - x_{\sigma_j} \leq y_i^\sigma \tag{19}$$

$$\sum_{i \leq |\sigma|} y_i^\sigma \leq |\sigma| - 1. \tag{20}$$

Constraints (19) express the fact that there may be indices i for which x_{σ_i} is minimum in $x[\sigma]$, and (20) say that there is at least one such i, thus yielding a narrowing BreakSymm2(P) that is independent of of the choice of i. We remark that an aggregated version of (19), when combined with (20), produces a narrowing Q' such that $\mathcal{F}(Q') = \mathcal{F}(\text{BreakSymm2}(P))$:

$$\forall i \leq k \quad (|\sigma| - 1)x_{\sigma_i} - \sum_{j \neq i} x_{\sigma_j} \leq (|\sigma| - 1)y_i^\sigma. \tag{21}$$

Although in general aggregated constraints tend to produce slacker linear relaxations [19], we mention (21) here because in the tested instances they usually improve CPU times.

Since (19)-(20) together simply express the fact that there is a minimum component in each $x[\sigma]$, and this sentence is true for each disjoint cycle and each permutation in G_P, it follows that (19)-(20) can be added to P for each cycle σ appearing in the set of disjoint cycles over all permutations of G_P, yielding a valid narrowing denoted by BreakSymm2All(P). This, however, adds several variables and constraints to P, which implies that the size of the solution set increases, and each linear relaxation costs more in terms of CPU time; moreover, although BreakSymm2All(P) is a valid narrowing of P, the relaxation of BreakSymm2All(P) need not be strictly tighter than that of P.

4.1 Computational Experiments

The results we report should be treated as preliminary experiments rather than full computational results. Although the whole software structure is in place and the process of finding symmetries and then solutions of MILPs has been made fully automatic (by means of several different software packages, some developed on purpose and some off-the-shelf such as AMPL [4], CPLEX [8] and GAP [6]), finding elements of G_P automatically is still too costly in terms of CPU time. Moreover, because of the small size of problems for which it was possible to automatically compute symmetries, the addition of constraints and variables and consequent higher cost at each BB node offset the advantages of the narrowing as regards CPU time.

We employed the test set as described in Table 1. *Instance* is the instance name, *Source* lists the instance library or citation where the instance appears, *Integers* and *Constraints* report the number of integer variables and constraints respectively, *Size* lists the size of the instance in bytes, and *Infeasible* is 1 if the instance is infeasible. Table 2 reports the results. Column *Group* contains the subgroup of G_P found automatically by repeatedly solving a version of the FPP with random coefficients on the objective function (we call this the "randomized FPP procedure"); the group descriptions are non-unique, as there are many possible semi-direct product types $\ltimes$, see e.g. `sts27` and `stein27`. *Longest* contains the size of the longest cycle in the group generators, which is used to formulate SymmBreak2(P). N is the number of nodes in the BB tree of the original problem P, and N' is the number of nodes in the BB tree of BreakSymm2(P). The

Table 1. Test set description

Instance	*Source*	*Integers*	*Constraints*	*Size*	*Infeasible*
`enigma`	[18]	100	21	5616	0
`jgt18`	[11], p. 413	132	105	9399	1
`oa66234`	[17], Table 1	64	42	5176	0
`oa67233`	[17], Table 1	128	64	12153	0
`oa76234`	[17], Table 1	64	42	5176	0
`ofsub9`	[11], p. 413	203	92	13157	1
`stein27`	[18]	27	118	5789	0
`sts27`	[15], p. 17	27	117	5612	0

Table 2. Results of the computational experiments. The last column refers to SymmBreak2(P). The group descriptions were computed by GAP [6].

Instance	*Group*	*Longest*	N	N'
`enigma`	C_2	2	3321	**269**
`jgt18`	$C_2 \times S_4$	6	**573**	1300
`oa66234`	S_3	2	0	0
`oa67233`	$C_2 \times S_4$	6	6	**0**
`oa76234`	S_3	2	0	0
`ofsub9`	$C_3 \times S_7$	21	1111044	**980485**
`stein27`	$((C_3 \times C_3 \times C_3) \ltimes PSL(3,3)) \ltimes C_2$	24	**1084**	1843
`sts27`	$((C_3 \times C_3 \times C_3) \ltimes PSL(3,3)) \ltimes C_2$	26	1317	**968**

instances were solved by CPLEX 10.1 [8] on one core of a 32 bit Intel Core Duo 1.2GHz with 1.5GB RAM running Linux.
Remarks on the experiments.

- Although the instance set is definitely still too small to draw significant conclusions (work is ongoing to enlarge it), the encouraging result is that there was an improvement on the only really difficult instance (`ofsub9`): even more so as, it being infeasible, BB performance is poor because there are many fewer prunings than with feasible ones (no upper bounding objective function value is ever available).
- The extensions of Sect. 4 applied to (6) did not yield good results due to the increased constraint matrix density and bad scaling.
- Using the straight version of (6) and (7) (with the arbitrary choice on the chosen longest cycle orbit representative) sometimes decreases N' so that even the CPU times are improved, depending on the arbitrary choice; in such cases, (6) were better than (7) as expected.
- Instance `stein27` is like `sts27` but with an added cardinality constraint (sum of all variables ≥ 13, a constraint which is inactive on all optimal solutions); this (small) difference in problem formulation caused the randomized FPP procedure to find different permutations (although leading to the same group description, the exact group structure is different) and hence to different performance.

– Sometimes long "easy" cycles are overlooked, such as in the case of `oa66234` and `oa76234`: the group is S_3, yet the cycle only has length 2. This happens because we select disjoint cycles from the group generators instead of the group elements themselves to avoid listing all permutations of a group. At the moment we let GAP select the generators automatically, but an improved implementation should take care of selecting generators having long cycles.

5 Conclusion and Future Work

We proposed an automatic way to compute permutations of the symmetry group of a MILP in general form (1) and derived two types of global symmetry breaking constraints designed to reduce the number of symmetric solutions. We exhibited a few preliminary experimental results indicating a positive trend. Future work will concentrate on: (i) reducing the computational effort taken to find permutations by means of exploitation of the block structure of the MILP constraint matrix; (ii) finding a method to reduce the influence of the arbitrary choice of orbit representative in (6), (7) *not* based on adding variables to the problem; (iii) extend the computational results to a more significant instance test set.

References

1. Bell, D.: Constructive group relaxations for integer programs. SIAM Journal on Applied Mathematics 30(4), 708–719 (1976)
2. Boulle, M.: Compact mathematical formulation for graph partitioning. Optimization and Engineering 5, 315–333 (2004)
3. Cohen, D., Jeavons, P., Jefferson, C., Petrie, K., Smith, B.: Symmetry definitions for constraint satisfaction problems. In: van Beek, P. (ed.) CP 2005. LNCS, vol. 3709, pp. 17–31. Springer, Heidelberg (2005)
4. Fourer, R., Gay, D.: The AMPL Book. Duxbury Press, Pacific Grove (2002)
5. Friedman, E.J.: Fundamental domains for integer programs with symmetries. In: Dress, A.W.M., Xu, Y., Zhu, B. (eds.) COCOA. LNCS, vol. 4616, pp. 146–153. Springer, Heidelberg (2007)
6. The GAP Group. GAP – Groups, Algorithms, and Programming, Version 4.4.10 (2007)
7. Gomory, R.: Some polyhedra related to combinatorial problems. Linear Algebra and Its Applications 2(4), 451–558 (1969)
8. ILOG. ILOG CPLEX 10.1 User's Manual. ILOG S.A., Gentilly, France (2006)
9. Kaibel, V., Pfetsch, M.: Packing and partitioning orbitopes. Mathematical Programming (to appear)
10. Lee, J.: All-different polytopes. Journal of Combinatorial Optimization 6, 335–352 (2002)
11. Lee, J., Margot, F.: On a binary-encoded ILP coloring formulation. INFORMS Journal on Computing 19(3), 406–415 (2007)
12. Liberti, L.: Reformulation techniques in mathematical programming, Thèse d'Habilitation à Diriger des Recherches (November 2007)

13. Liberti, L.: Reformulations in mathematical programming: Definitions. In: Aringhieri, R., Cordone, R., Righini, G. (eds.) Proceedings of the 7th Cologne-Twente Workshop on Graphs and Combinatorial Optimization, Crema, Università Statale di Milano (2008)
14. Margot, F.: Pruning by isomorphism in branch-and-cut. Mathematical Programming 94, 71–90 (2002)
15. Margot, F.: Exploiting orbits in symmetric ILP. Mathematical Programming B 98, 3–21 (2003)
16. Margot, F.: Small covering designs by branch-and-cut. Mathematical Programming B 94, 207–220 (2003)
17. Margot, F.: Symmetric ILP: coloring and small integers. Discrete Optimization 4, 40–62 (2007)
18. Martin, A., Achterberg, T., Koch, T.: MIPLIB 2003. A library of pure and mixed-Integer programs (2003)
19. Nemhauser, G.L., Wolsey, L.A.: Integer and Combinatorial Optimization. Wiley, New York (1988)
20. Ostrowski, J., Linderoth, J., Rossi, F., Smriglio, S.: Orbital branching. In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 104–118. Springer, Heidelberg (2007)
21. Sherali, H., Smith, C.: Improving discrete model representations via symmetry considerations. Management Science 47(10), 1396–1407 (2001)
22. Wolsey, L.: Group representation theory in integer programming. Technical Report Op. Res. Center 41, MIT (1969)