On the generation of symmetry breaking constraints for mathematical programs

Leo Liberti[1], James Ostrowski[2]

[1] *LIX, Ecole Polytechnique, 91128 Palaiseau, France*
Email:liberti@lix.polytechnique.fr

[2] *Decision and Information Sciences, Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL 60439, USA*
Email:jostrowski@anl.gov

July 15, 2011

Abstract

Mathematical programs whose formulation is symmetric often take a long time to solve using Branch-and-Bound type algorithms, because of the several symmetric optima. One of the techniques used to decrease the adverse effects of symmetry is adjoining symmetry breaking constraints to the formulation before solving the problem. These constraints aim to make some of the symmetric optima infeasible but guarantee that at least one optimum is feasible. Such constraints are usually associated to the different orbits of the action of the formulation group on the set of variable indices. In general, one cannot adjoin symmetry breaking constraints from more than one orbit. In previous work [14] we discussed some (restrictive) sufficient conditions for adjoining such constraints from several orbits. In this paper we present a new method for generating symmetry breaking constraints from several orbits. We show it is less restrictive than the existing method, and discuss some computational experiments.
Keywords: mathematical programming, static symmetry breaking, MILP, MINLP.

1 Introduction

It is well known that Mathematical Programs (MP) with nontrivial symmetry on the decision variables may take Branch-and-Bound (BB) type solvers a very long time to prove the optimality of a solution, due to the many symmetric optima in the leaves of the BB tree [16, 13]. One way to deal with this issue is to reformulate the MP in such a way as to make several symmetric optima infeasible, whilst guaranteeing that at least one optimum will remain feasible (such a reformulation is called a *narrowing* [12]). Two strategies are available: *static symmetric breaking*, which is very easy to deploy, and *dynamic symmetry breaking*, which is more difficult but, in general, also more effective. Both are discussed in [16] for Mixed-Integer Linear Programs (MILP). The occurrence of symmetry in MP is far from rare. The computational experiments in [14] show that 18% of the instances in the MIPLib3 [4], MIPLib2003 [17], GlobalLib [5], MINLPLib public instance libraries have a nontrivial formulation group.

Static symmetry breaking consists in adjoining some *symmetry breaking constraints* (SBC) to the problem formulation. Several SBCs can be found in the literature for specific problems (such as, e.g., the Quadratic Assignment Problem [8], the Kissing Number Problem [13], the problem of Packing Equal Circles in a Square [7]), or specific symmetry groups (such as the full symmetric group [19]). Several SBC classes for MILPs are surveyed in [16]. Some general-purpose SBCs that work in the full MINLP framework (which also includes the case of continuous Nonlinear Programs (NLP)) are discussed in [14].

Although there appears to be no clearly defined relation between how large a formulation group is and how hard it is to solve the MP in practice, there are two common sense arguments motivating the study of symmetries in MP, having to do with two different algorithmic classes. When the MP is solved

exactly (or approximately) using BB algorithms, such as [10] when it is MILP or [3] when it is a MINLP, and only one global optimum is required, then multiple symmetric global optima generally yield larger BB trees, and therefore longer solution processes. When heuristic or meta-heuristic algorithms (e.g. [15]) are used in order to find good solutions of P, the picture is often reversed: if the algorithm stochastically explores the neighbourhood of the most recent (or best) found local optimum, having several optima available in the neighbourhood prevents the algorithm from getting stuck early on in the search [13].

In this paper we address a limitation of the SBC generation method proposed in [14]. SBCs can be derived from each orbit of the action of the formulation group on the set of variable indices. In general, however, only SBCs referring to a single orbit can be adjoined to the formulation. If the formulation group has several orbits (as is often practically the case), however, the improvement on the BB solution time will be moderate if SBCs breaking symmetry for only one orbit are adjoined. This limitation was overcome in [14] by specifying sufficient conditions on subsets of orbits which makes it possible to simultaneously adjoin to the formulation SBCs from each orbit in the subset. These sufficient conditions, however, are too restrictive in practice. This paper discusses a different method for generating SBCs that can all simultaneously be adjoined to the formulation: we pick an orbit, generate the corresponding SBCs, then update the group with the orbit stabilizer; we repeat this procedure until the latter becomes trivial. We show empirically that the new method outperforms the old one on several MILP instances drawn from the literature.

The rest of this paper is organized as follows. We present background material, formal defitions and notation in Sect. 2, discuss the new method in Sect. 3, and the corresponding computational results in 4.

2 Background and notation

First, we give a formal definition of the type of MPs we consider. This would not be strictly necessary for MILPs only. It is necessary, however, to characterize the type of nonlinear functions we allow in MINLPs. We then recall some group definitions, including the formulation group of a MP and of a graph, and finally formally introduce SBCs.

2.1 A formal definition for MP

Mathematical Programming (MP) is a formal language for describing optimization problems of the form:

$$\min\{f(x) \mid g(x) \leq 0 \wedge x \in X\}, \tag{1}$$

where $x \in \mathbb{R}^n$ is a vector of decision variables, $X \subseteq \mathbb{R}^n$ might include bounds and integrality constraints on subsequences of x, and $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^m$ are functions that can be written as strings of a formal language $\mathscr{E}$ on the alphabet $\mathscr{A} = \mathscr{O} \cup \mathbb{Q} \cup \mathscr{V}$, where $\mathscr{V} = \{\mathsf{x}_i \mid i \in \mathbb{N}\}$ and $\mathscr{O} = \{+, -, \times, \div, (\cdot)^{(\cdot)}, \log, \exp, (,)\}$. The strings of $\mathscr{E}$ are only and all those that can be obtained as follows: (a) $\forall \mathsf{s} \in \mathbb{Q} \cup \mathscr{V}$ ($\mathsf{s} \in \mathscr{E}$); (b) $\forall \otimes \in \mathscr{O}$ representing a k-ary operator and $\mathsf{e}_1, \ldots, \mathsf{e}_k \in \mathscr{E}$, $\otimes(\mathsf{e}_j \mid j \leq k)$ is in $\mathscr{E}$ [7]. If $P \in$ MP, let $\mathcal{F}(P) \subseteq X$ be the set of its feasible solutions, i.e. those $x \in X$ satisfying $g(x) \leq 0$, and $\mathcal{G}(P) \subseteq \mathcal{F}(P)$ the set of its globally optimal solutions.

The recognition process (or parsing) of a valid string $h \in \mathscr{E}$ naturally yields a directed graph $D(h)$ whose leaf nodes are elements of $\mathbb{Q} \cup \mathscr{V}$ and whose non-leaf nodes are elements of $\mathscr{O}$ [7]. Every $P \in$ MP has a Directed Acyclic Graph (DAG) representation $D(P)$ obtained as a minor of $D(f) \cup \bigcup_{i \leq m} D(g_i)$ by contracting all equal leaf nodes [7].

We let $\mathcal{F}(P)$ be the set of feasible solutions of a given MP, and $\mathcal{G}(P)$ be the set of globally optimal solutions.

2.2 Solution and formulation groups

Let $[n] = \{1, \ldots, n\}$. For any vector $v \in \mathbb{R}^n$ and a subset $B \subseteq [n]$, we let $v[B]$ be the vector consisting of the components of v indexed by elements of B.

A group G is generated by elements of a set S (denoted $G = \langle S \rangle$) when G is the closure of S with respect to the group product. For a group G acting on X, we let $Gx = \{gx \mid g \in G\}$ be the orbit of x in G for all $x \in X$; we let $\mathsf{stab}(Y, G) = \{g \in G \mid \forall y \in Y \ (gy \in Y)\}$ be the setwise stabilizer and $G^Y = \{g \in G \mid \forall y \in Y \ (gy = y)\}$ be the pointwise stabilizer of Y w.r.t. G for all $Y \subseteq X$. For a permutation $\pi \in S_n$ let $\Gamma(\pi)$ be the set of all the cycles in its (unique) disjoint cycle representation. For $N \subseteq [n]$ we define $\pi[N] = \prod_{\sigma \in \Gamma(\pi) \cap \mathsf{stab}(N, S_n)} \sigma$ to be the restriction of π to N. If $g_1, \ldots, g_k \in G \leq S_n$ are generators for G (i.e. $G = \langle g_j \mid j \leq k \rangle$) we define $G[N] = \langle g_j[N] \mid j \leq k \rangle$ to be the restriction of G to N. If N is an orbit of the action of G on $[n]$ then $G[N]$ is a *transitive constituent* of G. By [6], p. 35, for each orbit ω of G, the (right) map $\cdot[\omega] : G \to G[\omega]$ given by $\pi \to \pi[\omega]$ is a group homomorphism whose kernel is the pointwise stabilizer G^ω, which is therefore a normal subgroup of G [1].

We remark that stabilizers are usually denoted by G_Y rather than G^Y, and transitive constituents by G^N [6, 18]. Our nonstandard notation allows us to disambiguate expressions such as G_P as concerns the two possible interpretations "the group indexed by symbol P" and "the stabilizer of the set P".

We consider the action of $G \leq S_n$ on X given by $\pi x = (x_{\pi(i)} \mid i \leq n)$. This action induces a right action $P \mapsto P\pi$ (where $\pi \in S_n$) on MP by replacing x with πx everywhere in (1). S_m also induces a left action $P \mapsto \sigma P$ (where $\sigma \in S_m$) given by replacing $g = (g_1, \ldots, g_m)$ by σg. Because optimization problems (1) are independent of the order of the constraints, $\sigma P = P$ for all $\sigma \in S_m$, which implies $\forall \sigma \in S_m, \pi \in S_n \ (\sigma P)\pi = \sigma(P\pi)$.

We define the *solution group* $G^*(P) = \mathsf{stab}(\mathcal{G}(P), S_n)$ and the *formulation group* $G_P = \langle \pi \in S_n \mid \exists \sigma \in S_m \ (\sigma P \pi = P) \rangle$ of a MP formulation P given by (1); it is easy to show that $G_P \leq G^*(P)$. Computing the solution group in general requires aprioristic knowledge of $\mathcal{G}(P)$, which is usually the ultimate aim when considering and solving MPs, and is therefore impractical. Since deciding whether two function encodings $h_1, h_2 \in \mathscr{E}$ are equal has linear complexity in $|D(P)|$, computing generators for G_P is a decidable problem [14] which can be solved once the formulation of P is known. By choosing an appropriate colouring $\gamma : D(P) \to \mathbb{N}$ of the vertices of $D(P)$ (in order to avoid permutations of nodes of different types, e.g., operator nodes with variable nodes), we show that $G_P = \mathsf{Aut}(D(P), \gamma)[N]$, where $\mathsf{Aut}(\mathcal{G}, \delta)$ is the group of automorphisms of the graph $\mathcal{G}$ which stabilizes each equivalence class given by the vertex colouring δ setwise [14].

2.3 Symmetry breaking constraints

Formally, a *symmetry breaking constraint* (SBC) for problem P and a group $G \leq G_P$ is a set of constraints $g(x) \leq 0$ (where $g : \mathbb{R}^n \to \mathbb{R}^p$ for some $p \in \mathbb{N}$) such that there exists an optimum $y \in \mathcal{G}(P)$ and $\pi \in G$ with $g(\pi y) \leq 0$ [14]. Strictly speaking, this definition does not force SBCs to actually break any symmetry, but only to keep at least one optimum feasible; this ensures the definition also works with optima y which are invariant to all permutations. In practice, however, SBCs are constructed in such a way that $g(\sigma y) > 0$ for as many $\sigma \in G$ as possible, so as to make as many symmetric optima in $\mathcal{G}(P)$ as possible infeasible in the narrowing (a definition in this sense was given in [11]). If $g(x) \leq 0$ are SBCs involving only variables x_j with j in a given set B, we emphasize this by writing $g[B](x) \leq 0$, and say $g(x) \leq 0$ are *SBCs with respect to* B. G is taken to be the whole of G_P unless specified otherwise.

In [14], new methods were presented in order to break symmetries with two types of general-purpose SBC derived from the set Ω of orbits of the action of G_P on the variable index set $[n]$. Specifically, for a nontrivial orbit $\omega \in \Omega$, if the transitive constituent $G_P[\omega]$ can be ascertained to be isomorphic to the full symmetric group $\mathsf{Sym}(\omega)$ on ω, then a unique order can be imposed on the variables indexed by ω

by adjoining the following linear inequalities to P:

$$\forall j \in \omega \smallsetminus \{\max \omega\} \quad x_j \leq x_{j^+}, \tag{2}$$

where j^+ is the successor of j in ω. Otherwise, for any structure $G_P[\omega]$ might have, one can always choose a variable (for example $x_{\min \omega}$) that should have minimum values among all those indexed by ω:

$$\forall j \in \omega \smallsetminus \{\min \omega\} \quad x_{\min \omega} \leq x_j. \tag{3}$$

In general, if $\omega, \theta \in \Omega$ with $\omega \neq \theta$ and $g^\omega(x) \leq 0$ and $g^\theta(x) \leq 0$ are SBCs with respect to, respectively, ω and θ, adjoining both $g^\omega(x) \leq 0$ and $g^\theta(x) \leq 0$ to P may not yield a valid narrowing of P, as Example 2.1 shows.

2.1 Example
Let P be the following MILP:

$$\min_{x \in \{0,1\}^4} \left\{ x_1 + x_2 + x_3 + x_4 \;\middle|\; \begin{pmatrix} -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \geq \begin{pmatrix} -1 \\ -1 \\ 1 \\ 1 \end{pmatrix}. \right\}$$

Its formulation group is $G_P = \langle (1,2)(3,4) \rangle \cong C_2$, with two orbits $\omega_1 = \{1,2\}$ and $\omega_2 = \{3,4\}$ and optima $\mathcal{G}(P) = \{(0,1,1,0),(1,0,0,1)\}$. Valid SBCs for ω_1 (resp. ω_2) are $x_1 \leq x_2$ (resp. $x_3 \leq x_4$). Whereas adjoining either of the two SBCs leads to a valid narrowing, adjoining both at the same time results in an infeasible problem.

2.4 Coprime orbits narrowing

We showed in [14] some sufficient conditions by which SBCs originating from different orbits could be combined into a valid narrowing of P. By Cor. 14 in [14], this holds if:

- $G_P[\omega \cup \theta]$ contains a subgroup H such that $H[\omega] \cong C_{|\omega|}$ and $H[\theta] \cong C_{|\theta|}$ (where C_p is the cyclic group of order p for all $p \in \mathbb{N}$);
- $\mathsf{gcd}(|\omega|, |\theta|) = 1$.

Adjoining such SBCs to the original formulation results into a reformulation which we call the *coprime orbits narrowing*.

We remark that these conditions are restrictive but not necessary, as Example 2.2 shows.

2.2 Example
The problem $P \equiv \min\{\sum_{j \leq 6} x_j \mid x_1 + x_2 + 2\sum_{3 \leq j \leq 6} x_j \geq 3\}$ has formulation group $G_P = \langle (1,2),(3,4),(4,5),(5,6) \rangle$. The action of G on $\{1,\ldots,6\}$ has the two orbits $\omega_1 = \{1,2\}$ and $\omega_2 = \{3,4,5,6\}$, yielding SBCs $x_1 \leq x_2$ and, respectively, $x_3 \leq x_4$, $x_3 \leq x_5$, $x_3 \leq x_6$. These two sets of SBCs can both be adjoined to P at the same time, even though $\mathsf{gcd}(|\omega_1|, |\omega_2|) = 2 \neq 1$.

3 A new method for SBC generation

Consider the SBCs generated by the following procedure.

1. Let $G = G_P$ and $C = \varnothing$;
2. Let Ω be the set of orbits of the action of G on $[n]$;
3. Choose an orbit $\omega \in \Omega$;
4. Let $g[\omega](x) \leq 0$ be some SBCs for P and G w.r.t. ω;
5. Let $C = C \cup \{g[\omega](x) \leq 0\}$;
6. If the (pointwise) stabilizer G^ω is nontrivial:
   (a) replace G with G^ω
   (b) adjoin $g[\omega](x) \leq 0$ to P
   (c) go to Step 2;
7. Return C.

This algorithm recursively builds a sequence $\omega_1, \ldots, \omega_k$ of subsets of $[n]$ with associated SBCs $g_\ell[\omega_\ell](x) \leq 0$, and a chain of normal subgroups

$$G_P = G_1 \rhd G_1^{\omega_1} = G_2 \rhd \ldots \rhd G_{k-1}^{\omega_{k-1}} = G_k \rhd G_k^{\omega_k} = \{e\}.$$

It is easy to see that $G_\ell = G_P^{\bigcup_{i<\ell} \omega_i}$ for each $\ell \leq k$, i.e. permutations in G_ℓ stabilize all elements of the orbits $\omega_1, \ldots, \omega_{\ell-1}$ pointwise.

3.1 Theorem
The constraint set $C_k = \{g_\ell[\omega_\ell](x) \leq 0 \mid \ell \leq k\}$ is an SBC system for P.

Proof. We prove this by induction on k. When $k = 1$ the result trivially holds because $g_1[\omega_1](x) \leq 0$ are SBCs for P by construction. Let P' be like P with all the constraints in C_{k-1} adjoined to it. By the induction hypothesis C_{k-1} is an SBC set for P, so $\mathcal{G}(P')$ is non-empty; and since $k-1$ is not the last iteration of the algorithm, G_k is a nontrivial group. Since $g_k[\omega_k](x) \leq 0$ are SBCs for P' and G_k with respect to ω_k, there exist $y \in \mathcal{G}(P')$ and $\pi \in G_k$ such that $g_k[\omega_k](\pi y) \leq 0$. Since $\pi \in G_k$, it stabilizes the orbits $\omega_1, \ldots, \omega_{k-1}$ pointwise, which implies that $(\pi y)[\omega_\ell] = y[\omega_\ell]$ for all $\ell < k$. Since $y \in \mathcal{G}(P')$, y is feasible in P', so it satisfies all constraints in C_{k-1}: this means $\forall \ell < k\ g[\omega_\ell](\pi y) \leq 0$, which concludes the proof. □

Adjoining C to the original formulation yields a reformulation which we call the *stabilizer narrowing*.

We remark that there is a choice of ω in Step 3 which guarantees that the stabilizer narrowing is always "stronger" than the coprime orbits one: i.e., by choosing coprime orbits first. The notion of strength we consider here is based on the $\subseteq$ relation on the sets of SBCs generated by the two narrowings: a set S is stronger than S' if $S' \subseteq S$. It was observed empirically, however, that best results are obtained when ω is chosen as the largest orbit in Ω, which is the policy we follow below.

4 Computational results

We compare the effect that the coprime orbits narrowing (Sect. 2.4) and the stabilizer narrowing (Sect. 3) have on BB. Our test set includes MILPs (taken from François Margot's website, from MIPLib3 and from MIPLib2003), (nonconvex) NLPs (taken from GlobalLib) and MINLPs (taken from MINLPLib): we solve linear instances using CPLEX 12.2 [9] and nonlinear ones using Couenne [3, 2]. The test set consists of 118 instances having nontrivial formulation group. Out of these, 76 were discarded, as the coprime orbits

and the stabilizer narrowing procedures yielded the same reformulation (this happens, for example, on transitive groups, since they only have one orbit). Five instances caused AMPL/COUENNE errors. The remaining set contains 38 instances partitioned as follows. 27 MILPs:

`air03 arki001 cod105r cod83r cod93r flosn52 flosn60 jgt18 jgt30 mas74 mas76 mered misc06 mitre oa66234 oa67233 oa68233 oa76234 oa77233 of5_14_7 of7_18_9 ofsub9 p0201 p2756 protfold qiu rgn`

7 NLPs:

`ex8_3_1 ex8_3_2 ex8_3_3 ex8_3_4 ex8_3_5 ex9_2_6 st_rv9`

and 3 MINLPs:

`cecil_13 gastrans hmittelman`

The results are given in Table 1 for MILPs, Table 2 for NLPs and Table 3 for MINLPs. For both coprime orbits and stabilizer narrowings, we report: value of the incumbent, solution time in seconds, optimality gap and status ("opt" when the optimum was found, "limit" when the time limit was reached, "infeas" when the instance was proved infeasible). All results were obtained on a quad-CPU Intel Xeon at 2.66 GHz with 24GB RAM. The solvers were allowed a maximum of 1800 seconds of solution time. The solution time corresponds to "real time" for CPLEX 12.2, which is by default a parallel solver and exploits all CPUs (this is why the time limit is hit with different values of *Time* column in Table 1). For COUENNE, solution time corresponds to seconds of user time on one CPU.

It appears clear from the results in Tables 1-3 that the stabilizer narrowing reformulation has a remarkably positive effect when solving MILPs using CPLEX 12.2; its impact on NLP and MINLP remains questionable. However, the NLP and MINLP test sets have excessively small sizes for us to draw meaningful conclusions.

Acknowledgments

The first author was partially supported by the Digiteo Chair grant 2009-14D "RMNCCO".

References

[1] R. Allenby. *Rings, Fields and Groups: an Introduction to Abstract Algebra*. Edward Arnold, London, 1991.

[2] P. Belotti. COUENNE*: a user's manual*. Dept. of Mathematical Sciences, Clemson University, 2011.

[3] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software*, 24(4):597–634, 2009.

[4] R. Bixby, S. Ceria, C. McZeal, and M. Savelsbergh. An updated mixed integer programming library: Miplib 3. Technical Report TR98-03, Rice University, 1998.

[5] M. Bussieck. Globallib — a collection of nonlinear programming problems, 2004. (`http://www.gamsworld.org/global/globallib.htm`).

[6] P. Cameron. Polynomial aspects of codes, matroids and permutation groups, online. Lecture notes.

MILP	Coprime orbits				Stabilizer			
Instance	*Incumbent*	*Time*	*Gap*	*Status*	*Incumbent*	*Time*	*Gap*	*Status*
air03	340160	1.66	0%	opt	340160	**1.06**	0%	opt
arki001	7.58122e+06	30.35	0%	opt	7.58122e+06	**13.73**	0%	opt
cod105r	-11	34.04	0%	opt	-11	**27.59**	0%	opt
cod83r	-19	53.14	0%	opt	-19	**27.04**	0%	opt
cod93r	-39	3315.69	9.31%	limit	-39	4699.36	0%	**opt**
flosn52	$+\infty$	19.3	NA	infeas	$+\infty$	**0**	NA	infeas
flosn60	$+\infty$	69.79	NA	infeas	$+\infty$	**0**	NA	infeas
jgt18	$+\infty$	2.09	NA	infeas	$+\infty$	**0.95**	NA	infeas
jgt30	$+\infty$	419.56	NA	infeas	$+\infty$	**131.7**	NA	infeas
mas74	11857.4	46.1	6.6876%	limit	11801.2	458.81	0%	**opt**
mas76	40005.4	62.29	0%	opt	40005.4	**48.73**	0%	opt
mered	$+\infty$	4.7	NA	infeas	$+\infty$	**0.99**	NA	infeas
misc06	12850.8	0.48	0%	opt	12850.8	**0.29**	0%	opt
mitre	115155	0.88	0%	opt	115155	**0.64**	0%	opt
oa66234	48	**0**	0%	opt	48	0.02	0%	opt
oa67233	48	**0**	0%	opt	48	0.01	0%	opt
oa68233	48	0.42	0%	opt	48	**0.2**	0%	opt
oa76234	56	0.02	0%	opt	56	0.02	0%	opt
oa77233	56	**0.03**	0%	opt	56	0.04	0%	opt
of5_14_7	$+\infty$	0.28	NA	infeas	$+\infty$	**0.19**	NA	infeas
of7_18_9	$+\infty$	0.17	NA	infeas	$+\infty$	**0.08**	NA	infeas
ofsub9	$+\infty$	2394.37	NA	infeas	$+\infty$	**288.64**	NA	infeas
p0201	7615	0.71	0%	opt	7615	**0.33**	0%	opt
p2756	3124	**0.87**	0%	opt	3124	1.16	0%	opt
protfold	-15	3313.45	147.778%	limit	-26	6753.95	**39.32%**	limit
qiu	-132.873	**46.35**	0%	opt	-132.873	64.89	0%	opt
rgn	82.2	0.34	0%	opt	82.2	**0.22**	0%	opt

Table 1: MILP results obtained using CPLEX.

NLP	Coprime orbits				Stabilizer			
Instance	*Incumbent*	*Time*	*Gap*	*Status*	*Incumbent*	*Time*	*Gap*	*Status*
ex8_3_1	-0.814115	**10.0375**	0%	opt	-0.814115	10.1645	0%	opt
ex8_3_2	-0.41233	1807.79	2325.24%	limit	-0.41233	1807.41	2325.24%	limit
ex8_3_3	-0.416603	2.6616	0%	opt	-0.416603	**1.01884**	0%	opt
ex8_3_4	-3.57998	1805.81	179.331%	limit	-3.57998	1806.31	179.331%	limit
ex8_3_5	-0.0691197	**3.48947**	0%	opt	-0.0691197	7.59384	0%	opt
ex9_2_6	-1	**0.038993**	0%	opt	-1	0.043993	0%	opt
st_rv9	-120.153	4.81027	0%	opt	-120.153	**3.75543**	0%	opt

Table 2: NLP results obtained using COUENNE.

MINLP	Coprime orbits				Stabilizer			
Instance	*Incumbent*	*Time*	*Gap*	*Status*	*Incumbent*	*Time*	*Gap*	*Status*
cecil_13	$+\infty$	118.764	0%	infeas	$+\infty$	**118.692**	0%	infeas
gastrans	89.0858	**2.58461**	0%	opt	89.0858	2.6426	0%	opt
hmittelman	13	**0.178972**	0%	opt	13	0.184971	0%	opt

Table 3: MINLP results obtained using COUENNE.

[7] A. Costa, P. Hansen, and L. Liberti. Formulation symmetries in circle packing. In R. Mahjoub, editor, *Proceedings of the International Symposium on Combinatorial Optimization*, volume 36 of *Electronic Notes in Discrete Mathematics*, pages 1303–1310, Amsterdam, 2010. Elsevier.

[8] M. Fischetti, M. Monaci, and D. Salvagnin. Three ideas for the quadratic assignment problem. *Operations Research*, submitted.

[9] IBM. *ILOG CPLEX 12.2 User's Manual*. IBM, 2010.

[10] ILOG. *ILOG CPLEX 11.0 User's Manual*. ILOG S.A., Gentilly, France, 2008.

[11] L. Liberti. Automatic generation of symmetry-breaking constraints. In B. Yang, D.-Z. Du, and C.A. Wang, editors, *Combinatorial Optimization, Constraints and Applications (COCOA08)*, volume 5165 of *LNCS*, pages 328–338, Berlin, 2008. Springer.

[12] L. Liberti. Reformulations in mathematical programming: Definitions and systematics. *RAIRO-RO*, 43(1):55–86, 2009.

[13] L. Liberti. Symmetry in mathematical programming. In J. Lee and S. Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume IMA. Springer, New York, accepted.

[14] L. Liberti. Reformulations in mathematical programming: Automatic symmetry detection and exploitation. *Mathematical Programming*, DOI 10.1007/s10107-010-0351-0.

[15] L. Liberti, N. Mladenović, and G. Nannicini. A good recipe for solving MINLPs. In V. Maniezzo, T. Stützle, and S. Voß, editors, *Hybridizing metaheuristics and mathematical programming*, volume 10 of *Annals of Information Systems*, pages 231–244, New York, 2009. Springer.

[16] F. Margot. Symmetry in integer linear programming. In M. Jünger, T. Liebling, D. Naddef, G. Nemhauser, W. Pulleyblank, G. Reinelt, G. Rinaldi, and L. Wolsey, editors, *50 Years of Integer Programming*, pages 647–681. Springer, Berlin, 2010.

[17] A. Martin, T. Achterberg, and T. Koch. Miplib 2003. Technical Report 05-28, ZIB, 2005.

[18] A. Seress. *Permutation Group Algorithms*. Cambridge University Press, Cambridge, 2003.

[19] H. Sherali and C. Smith. Improving discrete model representations via symmetry considerations. *Management Science*, 47(10):1396–1407, 2001.