Noname manuscript No.
(will be inserted by the editor)

Exploiting symmetries in mathematical programming via orbital independence

Gustavo Dias · Leo Liberti

Received: date / Accepted: date

Abstract The presence of symmetries in the solution set of mathematical programs requires the exploration of symmetric subtrees during the execution of Branch-and-Bound type algorithms and yields increases in computation times. When some of the solution symmetries are evident in the formulation, it is possible to deal with symmetries as a preprocessing step. In this sense, implementation-wise, one of the simplest approaches is to break symmetries by adjoining Symmetry-Breaking Constraints (SBCs) to the formulation. Designed to remove some of the symmetric global optima, these constraints are generated from each orbit of the action of the symmetries on the variable index set. Incompatible SBCs however make all of the global optima infeasible. In this paper we introduce and test a new concept of Orbital Independence which we define as independent sets of orbits. We provide necessary conditions for characterizing independent sets of orbits and also prove that such sets embed sufficient conditions to exploit symmetries from two or more distinct orbits concurrently. The theory developed is used to devise an algorithm that potentially identifies the largest independent set of orbits of any mathematical program. Extensive numerical experiments are provided to validate the theoretical results.

Keywords combinatorial optimization · symmetry breaking · group theory · quadratic programming

1 Introduction

Mathematical Programming (MP) is a descriptive language used to formalize several types of optimization problems by defining a class of corresponding mathematical

This paper is an extension of the work presented in [5].

Gustavo Dias
CNRS LIX, École Polytechnique, 91128 Palaiseau, France
E-mail: dias@lix.polytechnique.fr

Leo Liberti
CNRS LIX, École Polytechnique, 91128 Palaiseau, France
E-mail: liberti@lix.polytechnique.fr

programs [15]. Let $\mathbb{MP}$ denote such a class. In this context, we consider problems $P \in \mathbb{MP}$ in the following general form:

$$\left.\begin{array}{rl} \min\limits_{x} & f(x) \\ \forall i \in \mathcal{I}_I & g_i(x) \leq 0, \\ \forall i \in \mathcal{I}_E & g_i(x) = 0, \\ & x \in B. \end{array}\right\} \qquad (P)$$

In problem P, $f, g_i : \mathbb{R}^n \to \mathbb{R}$ are functions for which we have closed form expressions. The set B might contain nonfunctional constraints such as ranges $[x^L, x^U]$ for the decision variables, and/or integrality constraints. For a mathematical program P we let $\mathscr{F}(P)$ denote its feasible region and $\mathscr{G}(P)$ the set of its global optima.

Quite often researchers and practitioners face formulations P which contain undesired mathematical properties. In such cases, casting the given problem P into a different one, say $P' \in \mathbb{MP}$, is a natural strategy: P' is called a *reformulation* of P. Any reformulation P' shares (numerical) properties with P (e.g. the set of global optima), but is in some sense better than the original program. There are indeed many types of reformulations such as relaxations, approximations, variable changes, and all of them play an essential role in MP [15]. In our context, recall that the Branch-and-Bound (BB) algorithm paradigm is the most widely used technique for solving optimization problems formulated as P. Briefly, a tree-based search for global optima is performed in $\mathscr{F}(P)$. These algorithms may however converge slowly on problems whose feasible region has many symmetric global optima because all the symmetric leaf nodes in the BB tree must be visited in order to assert convergence. In fact, it was found that roughly 18% of the MP instances in commonly employed public libraries have nontrivial symmetry [16]. Symmetries are therefore investigated in MP mainly to reduce the computation time of BB type algorithms. In this sense, we aim to derive reformulations P' whose sets $\mathscr{G}(P')$ contain less symmetric global optima than $\mathscr{G}(P)$, meaning that $\mathscr{G}(P') \subseteq \mathscr{G}(P)$; and P' will be derived by addjoining Symmetry-Breaking Constraints (SBCs) to P.

In general, any strategy designed to cope with symmetries in MP can be divided into two main phases: (a) symmetry *detection* and (b) symmetry *exploitation*. When both phases (detection and exploitation) are performed before running the solution algorithm, we call such strategy a Static Symmetry Breaking (SSB) approach. Otherwise, a Dynamic Symmetry Breaking (DSB) approach [22]. This work concerns a general-purpose automated SSB strategy that advocates the usage of SBCs. Overall, we can further describe our methodology in a subtle higher level of detail w.r.t. the exploitation phase as follows: (a) *detect* formulation symmetries, (b) *generate* new constraints and (c) *reformulate* the original problem. The main contribution of this paper relates to the second step: constraint generation. Symmetry-breaking devices are usually derived from orbits of the action of the formulation group on the set of decision variables; however, one cannot simply use such devices for all the orbits simultaneously because some orbits depend on each other in a very precise mathematical sense. We devise herein a concept of Orbital Independence (OI) called *independent set of orbits* which allow us to overcome this limitation. We provide necessary conditions for characterizing independent sets of orbits and also prove that such sets embed sufficient conditions to exploit symmetries from two or more distinct orbits concurrently.

More precisely, this paper extends the content of [5] in three fronts: first we characterize independent sets of orbits via direct product of groups (Lemma 3) and prove that the OI conditions provided by Corollary 1 (see Section 3.3) are not sufficient to characterize independent sets of orbits (Example 2); second, we introduce a family of highly symmetric Binary Quadratic Programs (denoted by $\mathbb{BQP}$); and lastly, besides the original tests using symmetric instances from MIPLIB2010, we enlarge our dataset with symmetric instances from MINLPLib2 and randomly generated symmetric Binary Quadratic Programs.

The rest of the paper is organized as follows: in Section 2 we introduce notation, recall concepts of Group Theory and review some previous work related to symmetries in MP; in Section 3 we present all the theoretical developments concerning the OI framework; in Section 4 we describe in details the SBCs generation algorithm devised based on the recently constructed theory; and finally, computational experiments are provided and analysed in Section 5.

2 Notation and previous work

2.1 Group Theory

We consider that permutation groups act on vectors in $\mathbb{R}^n$ by permuting its components and that permutations act on sets of vectors by acting on each vector individually. For a vector $v \in \mathbb{R}^n$ and a subset $N \subseteq [n] = [1, \dots, n]$, we let $v[N]$ denote the projection of v on the coordinates indexed by N.

Nomenclature-wise, S_n and C_n are the symmetric and cyclic group of order n, respectively. $\mathsf{Sym}(X)$ is the symmetric group on a set X (e.g. $S_n = \mathsf{Sym}([n])$). Throughout the paper, let H and G denote permutation groups and $\diamond$ denote the group operator. If H is a subgroup of G, we write $H \leq G$. If H is isomorphic to G, we write $H \cong G$. If H is a normal subgroup of G, we write $H \lhd G$. $\langle \Delta \rangle$ denotes the group generated by the set Δ of generators and $G \times H$ denotes the direct product of groups.

Consider a set X. We recall that an orbit is an equivalence class of the quotient set $X/\!\sim$, where $i \sim j$ if there is a permutation $g \in G$ such that $g(i) = j$. This way, the group G partitions X into a set Ω_G of orbits $\omega_1, \dots, \omega_p$, for $p \in [1, \dots, |\Omega_G|]$. Moreover, we let $\omega(\ell)$ denote the ℓ-th element of ω (stored as a list).

Let $Y \subseteq X$. The pointwise stabilizer of Y w.r.t. G is the subgroup of permutations of G fixing each element of Y, i.e., $G^Y = \{g \in G \mid \forall y \in Y \ (g \diamond y = y)\}$. The setwise stabilizer of Y w.r.t. G is the subgroup of those permutations of G under which Y is invariant, i.e., $\mathsf{stab}(Y, G) = \{g \in G \mid \forall y \in Y \ (g \diamond y \in Y)\}$.

A group action is *transitive* on a set X if $s \sim t$ for each $s, t \in X$. If a group G acts transitively on a set X, then X is an orbit of G.

2.2 Literature review

Apart from problem oriented breaking techniques [17], most of the work regarding symmetries in MP was dedicated to develop general-purpose symmetry group computations and breaking techniques embedded in BB-type algorithms. The work in this sense follows three main streams.

The first is devoted to DSB techniques and was established by Margot [20,21]. He defined the relaxation group of Binary Linear Programs (denoted $\mathbb{BLP}$) and used it to derive pruning strategies and cuts by means of isomorphism; this technique is known as *isomorphism prunning*. The idea was later extended and named *orbital branching* in [25] by using valid disjunctions to orbits of the formulation group to derive BB braching rules.

The second refers to SSB techniques and the usage of SBCs to tighten the search space. It was established by Kaibel and coworkers [14,6], with the introduction of the packing and partitioning orbitopes. Inspired by orbitopes, Friedman proposed a more general approach named *fundamental domains* [10]. Liberti then studied and extended the use of general purpose SBCs to Mixed-Integer Nonlinear Programs (denoted by $\mathbb{MINLP}$) in [17,16].

Developed at first to Mixed-Integer Linear Programs (denoted by $\mathbb{MILP}$), the third stream (named *orbital shrinking* by Fischetti and Liberti [7]) focus on deriving compact symmetry free relaxations by replacing whole orbits by single variables. The technique was extended to convex $\mathbb{MINLP}$s and some nonconvex $\mathbb{MINLP}$s having a special structure. A recent survey on the subject is available in [8].

As concerns symmetry detection, the formal description of P in the language $\mathcal{L}$ can be parsed into a Directed Acyclic Graph (DAG) data structure T using a fairly simple context-free grammar [2]; we refer to [4] for further details and an example about this procedure. In practice, one can write P using a modelling language such as AMPL [9] and use an unpublished AMPL API to derive T and its set of nodes $V(T)$ [11]. Since T is a labelled graph, we know how to compute the group $\mathcal{G}$ of its *label-invariant isomorphisms* [23,24]. Furthermore, it was shown in [16] that the action of $\mathcal{G}$ can be projected to the leaf nodes of $V(T)$, which represent the set of decision variables of P, and that this projection induces a group homomorphism ϕ mapping $\mathcal{G}$ to a certain group image known as the formulation group of P.

Definition 1 The formulation group G_P of P is the group of permutations which acts on the set of decision variables of P while keeping the objective function $f(x)$ and the feasible region $\{g_i(x) \mid i \in \mathcal{I}_I \cup \mathcal{I}_E\}$ unchanged.

The *solution group* of P is the group of permutations which keeps the set $\mathscr{G}(P)$ invariant, i.e. $G^*_P = \mathsf{stab}(\mathscr{G}_P, S_n)$. It is easy to show that $G_P \leq G^*_P$. It is impractical however to compute solution groups because it requires aprioristic knowledge of $\mathscr{G}_P$. On the other hand, if G_P is nontrivial, one can use the methodology proposed in [16] to computing generators for G_P and extract symmetries from P prior to solving it. In [16, §8.3] one can find many examples of formulation groups that operate in MP, such as symmetric, cyclic, dihedral and groups which are represented by means of the direct product operator. Moreover, it is important to note that formulation groups act on the set of decision variables only because mathematical programs are invariant under constraint-order permutations.

Symmetry is exploited in MP in a number of different ways, however their most efficient exploitation appears to be their usage within DSB strategies [20,21,25]. Such approaches are, unfortunately, difficult to implement, as each solver code must be addressed separately. Their simplest exploitation is the SSB approach [22, §8.2] which consists in adjoining some SBCs to the original formulation P in the hope of making all but one of the symmetric global optima infeasible. This procedure yields a reformulation of the narrowing type [15].

Definition 2 Given a problem P, a narrowing P' of P is such that (a) there is a function $\eta : \mathscr{F}(P') \to \mathscr{F}(P)$ for which $\eta(\mathscr{G}(P')) \subseteq \eta(\mathscr{G}(P))$ and (b) P' is infeasible only if P is.

Following the usual trade-off between efficiency and generality, approaches which offer provable guarantees of removing symmetric optima are limited to special structures [14], whereas approaches which hold for any mathematical program in the large class P are mostly common-sense constraints designed to work in general [17].

As concerns symmetry exploitation via SBCs, the projection homomorphism ϕ defined above for $\mathcal{G}$ and the leaf nodes of the parsing tree can be restricted to act on G_P and generalized to project its action to any subset $Y \subseteq X$. Let ϕ_Y denote this generalized action projection homomorphism, which is defined as follows: for each $\pi \in G_P$, $\phi_Y(\pi)$ is the product of the cycles of π having all components in Y. If Y is some orbit $\omega \in \Omega_{G_P}$, then (a) the image of ϕ_Y is a group $G_P[\omega]$ called the *transitive constituent* of ω, (b) G^Y is the kernel of ϕ_Y and (c) $\mathsf{stab}(Y, G) = G$.

Now let $x^* \in \mathscr{G}(P)$. If $G_P[\omega] \cong \mathsf{Sym}(\omega)$ on the orbit, $\mathscr{G}(P)$ contains vectors which yield every possible order of $x^*[\omega]$ when projected onto ω. Thus we can arbitrarily choose one order, e.g.:

$$\forall \ell < |\omega| \quad x_{\omega(\ell)} \leq x_{\omega(\ell+1)}, \tag{1}$$

enforce this order by means of SBCs, and still be sure that at least one global optimum remains feasible. The constraints in Eq. (1) are called *strong SBCs*. If $G_P[\omega]$ has any other group structure, we observe that, by transitivity of the transitive constituent, at least one permutation in $G_P[\omega]$ will map the component having minimum value in $x^*[\omega]$ to the first component. This yields the *weak SBCs*:

$$\forall \ell \in \omega \smallsetminus \{\omega(1)\} \quad x_{\omega(1)} \leq x_{\omega(\ell)}. \tag{2}$$

Strong SBCs select one order out of $|\omega|!$ many, and hence are able to break all the symmetries in $G_P[\omega]$. Weak SBCs may unlikely achieve that. We let $g(x[N]) \leq 0$ denote SBCs involving only variables x_j with j in the set N.

Remark 1 The choice of minimum value and first components are arbitrary; alternative sets of SBCs can occur for other distinct choices.

Lastly, we refer readers to surveys [22, 17] for an assessment of the state of the art in symmetry handling methods in Mathematical Programming.

3 Orbital independence

In this section we start introducing our OI theoretical results. First we exemplify how incompatible SBCs cut global optima from a mathematical program; then we recall the OI conditions originally introduced in [16] and [19]; finally we introduce the concept of independent set of orbits and the conditions which we shall use to identify such sets within the algorithmic framework presented in Section 4.

3.1 Incompatible SBCs

In general, one may only adjoin to P the SBCs associated to one single orbit. Example 1 shows that adjoining SBCs from two or more orbits chosen arbitrarily may result in all global optima being infeasible.

Example 1 Let P be the following $\mathbb{MILP}$:

$$\min_{x\in\{0,1\}^4} \quad x_1 + x_2 + 2x_3 + 2x_4$$
$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ -1 & 0 & -1 & 0 \\ 0 & -1 & 0 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \leq \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix}.$$

This problem has as set of optima $\mathscr{G}(P) = \{(0,1,1,0),(1,0,0,1)\}$. In addition, it has formulation group $G_P = \langle(1\ 2)(3\ 4)\rangle$ and orbits $\Omega_{G_P} = \{\omega_1, \omega_2\} = \{\{1,2\},\{3,4\}\}$. Valid SBCs for ω_1 (resp. ω_2) are $x_1 \leq x_2$ (resp. $x_3 \leq x_4$). By simple inspection of the optima set, whereas adjoining either of the two SBCs yields valid narrowings, adjoining both simultaneously leads to an infeasible mathematical program.

3.2 Some existing OI conditions

Sufficient conditions to concurrently combine sets of SBCs generated by two different orbits, say $\omega, \theta \in \Omega_{G_P}$, into a valid narrowing of $\mathbb{MINLP}$s are provided in [16]: (a) there is a subgroup $H \leq G_P[\omega \cup \theta]$ such that $H[\omega] \cong C_{|\omega|}$ and $H[\theta] \cong C_{|\theta|}$ and (b) $\mathsf{gcd}(|\omega|, |\theta|) = 1$. Two orbits with these properties are called *coprime*. The author proposes an algorithm that interatively builds a set Ω_C of pairwise coprime orbits: at iteration k, an orbit ω^k is randomly picked and tested against coprime orbits found in previous iterations; should ω^k pairwisely satisfy the conditions above w.r.t. all previously found coprime orbits, it is added to Ω_C. Once Ω_C is built, SBCs are generated for all of the orbits within it. The coprime conditions however are very restrictive and occur relatively rarely in practice, meaning that in general, most of information regarding the orbits remains unexploited.

A strategy based on chains of stabilizers is derived to overcome the limitations of the coprime narrowing in [19]. Based on the result that the map $\cdot[\omega] : G \to G[\omega]$ is a group homomorphism whose kernel is the pointwise stabilizer G^ω, which is therefore a normal subgroup of G, the authors propose an algorithm that, at each iteration k, identifies the set of orbits resulting from the action of a group G_k and randomly picks one of them to generate SBCs. The idea underlying this approach is how G_k is updated in between iterations: if ω^k is the orbit used to generate SBCs at iteration k, $G_{k+1} = G^{\omega^k} \lhd G_k$. Provided that G_{k+1} stabilizes ω^k pointwise and is a normal subgroup of G_k, ω^k is permanently factored out during the remaining iterations. The algorithm iterates until G_k becomes trivial, meaning that the stabilizers'chain was totally exploited.

Remark 2 In both methods, the orbits are arbitrarily chosen at each iteration; for the same mathematical program, different orbit choices lead to different sets of SBCs.

3.3 New conditions for OI

We will now build the concept of independent set of orbits and provide necessary conditions for characterizing such sets.

First, let $\omega, \theta \in \Omega_{G_P}$. We look at what happens to ω when θ is pointwise stabilized: either G^θ fixes ω, or a subset of ω, or it does not fix any element of ω at all. We can thus state the following binary *dependence* relations on the set Ω_{G_P}.

Definition 3 The orbit ω is:

(a) dependent of θ (denoted by $\omega \to \theta$) if, for any subset $\sigma \subseteq \omega$, $\sigma \notin \Omega_{G^\theta}$;
(b) semi-dependent of θ (denoted by $\omega \rightsquigarrow \theta$) if there is at least one subset $\sigma \subsetneq \omega$ such that $\sigma \in \Omega_{G^\theta}$;
(c) independent of θ (denoted by $\omega \Rsh \theta$) if $\omega \in \Omega_{G^\theta}$.

Next, let Γ^ω be the set of permutations of G_P which move elements of the orbit ω nontrivially. By definition, Γ^ω does not contain the identity permutation e of G_P and thus it is not itself a group. Moreover, the following properties trivially hold: (a) $G^\omega \cap \Gamma^\omega = \varnothing$, (b) $\mathsf{stab}(\omega, G_P) = G^\omega \cup \Gamma^\omega = G_P$ and (c), for $\omega \in \Omega_{G_P}$, $G_P[\omega] = \phi_\omega(\Gamma^\omega) \cup \{e\}$.

Theorem 1 establishes the dependence relation between two orbits $\omega, \theta \in \Omega_{G_P}$ by comparing the sets Γ^ω and Γ^θ.

Theorem 1 *The following statements are true:*

(1) If $\Gamma^\theta = \Gamma^\omega$ then $\theta \to \omega$ and $\omega \to \theta$;
(2) If $\Gamma^\theta \subset \Gamma^\omega$ then $\theta \to \omega$ and either $\omega \Rsh \theta$ or $\omega \rightsquigarrow \theta$;
(3) If $\Gamma^\theta \cap \Gamma^\omega \neq \varnothing$ then $(\theta \Rsh \omega$ or $\theta \rightsquigarrow \omega)$ and $(\omega \Rsh \theta$ or $\omega \rightsquigarrow \theta)$;
(4) If $\Gamma^\theta \cap \Gamma^\omega = \varnothing$ then $\theta \Rsh \omega$ and $\omega \Rsh \theta$.

Proof (1) Assume $\Gamma^\theta = \Gamma^\omega$ and consider ω. Then $G^\omega = G_P \smallsetminus \Gamma^\omega \Rightarrow G^\omega \cap \Gamma^\theta = \varnothing \Rightarrow \theta \notin \Omega_{G^\omega}$ and $\theta \to \omega$. Since the same argument holds if we consider θ, we also have $\omega \to \theta$.

(2) Assume $\Gamma^\theta \subset \Gamma^\omega$ and consider ω. Then $G^\omega = G_P \smallsetminus \Gamma^\omega \Rightarrow G^\omega \cap \Gamma^\theta = \varnothing \Rightarrow \theta \notin \Omega_{G^\omega}$ and $\theta \to \omega$. Considering θ, we have that $G^\theta = G_P \smallsetminus \Gamma^\theta \Rightarrow G^\theta \cap \Gamma^\omega \neq \varnothing$. If the action of G^θ is transitive on ω, we have $\omega \Rsh \theta$. Otherwise, we have $\omega \rightsquigarrow \theta$.

(3) Assume $\Gamma^\theta \cap \Gamma^\omega \neq \varnothing$ but neither set is wholly contained in the other, and consider ω. Then $G^\omega = G_P \smallsetminus \Gamma^\omega \Rightarrow G^\omega \cap \Gamma^\theta \neq \varnothing$. If the action of G^ω is transitive on θ, we have $\theta \Rsh \omega$. Otherwise, we have $\theta \rightsquigarrow \omega$. The same argument holds if we consider θ.

(4) Assume $\Gamma^\theta \cap \Gamma^\omega = \varnothing$ and consider ω. Then $G^\omega = G_P \smallsetminus \Gamma^\omega \Rightarrow G^\omega \supset \Gamma^\theta \Rightarrow \theta \in \Omega_{G^\omega}$ and $\theta \Rsh \omega$. The same holds if we consider θ, thus $\omega \Rsh \theta$. □

Lemma 1 *The premise $\Gamma^\theta \cap \Gamma^\omega = \varnothing$ to condition (4) in Theorem 1 never holds.*

Proof Let Δ be a set of generators of G_P. If there is a permutation $g \in \Delta$ such that $g[\omega]$ and $g[\theta]$ are nontrivial, then $g \in \Gamma^\theta \cap \Gamma^\omega$. Otherwise, let $\Delta^\theta = \{g \in \Delta \mid g[\omega] = e\}$ and $\Delta^\omega = \{g \in \Delta \mid g[\theta] = e\}$. Because every element of G_P can be expressed as the combination (under the group operation) of finitely many elements of Δ, there is $g \in G_P$ such that $g = g_\omega \diamond g_\theta$ where $g_\omega \in \Delta^\omega$ and $g_\theta \in \Delta^\theta$. Thus $g \in \Gamma^\theta \cap \Gamma^\omega$. □

Based on the above definitions and results, the following lemma holds.

Lemma 2 *The following statements are true:*

(1) The relation $\to$ is reflexive and the relations $\rightsquigarrow$ and $\Rsh$ are irreflexive;
(2) The relation $\to$ is symmetric iff $\Gamma^\theta = \Gamma^\omega$ and asymmetric iff $\Gamma^\theta \subset \Gamma^\omega$;

(3) The relation $\rightarrow$ is transitive.

Proof The proof of statements (1) and (2) follows directly from Theorem 1. As concerns (3), let $\theta, \omega, \tau \in \Omega_{G_P}$ be distinct orbits satisfying $\theta \rightarrow \omega$ and $\omega \rightarrow \tau$. From Theorem 1, $\theta \rightarrow \omega$ implies that either $\Gamma^\theta = \Gamma^\omega$ or $\Gamma^\theta \subset \Gamma^\omega$. Similarly, $\omega \rightarrow \tau$ implies that either $\Gamma^\omega = \Gamma^\tau$ or $\Gamma^\omega \subset \Gamma^\tau$. Then:

(a) $\Gamma^\theta = \Gamma^\omega \wedge \Gamma^\omega = \Gamma^\tau \Rightarrow \Gamma^\theta = \Gamma^\tau \Rightarrow \theta \rightarrow \tau$;
(b) $\Gamma^\theta = \Gamma^\omega \wedge \Gamma^\omega \subset \Gamma^\tau \Rightarrow \Gamma^\theta \subset \Gamma^\tau \Rightarrow \theta \rightarrow \tau$;
(c) $\Gamma^\theta \subset \Gamma^\omega \wedge \Gamma^\omega = \Gamma^\tau \Rightarrow \Gamma^\theta \subset \Gamma^\tau \Rightarrow \theta \rightarrow \tau$;
(d) $\Gamma^\theta \subset \Gamma^\omega \wedge \Gamma^\omega \subset \Gamma^\tau \Rightarrow \Gamma^\theta \subset \Gamma^\tau \Rightarrow \theta \rightarrow \tau$. $\square$

Whenever the dependence relations are symmetric, we write $\omega \leftrightarrow \theta$ or $\omega \leftrightsquigarrow \theta$ or $\omega \text{ ↲↳ } \theta$. Using this notation, we set forth that:

Definition 4 Two orbits $\omega, \theta \in \Omega_{G_P}$ are dependent if $\omega \leftrightarrow \theta$, semi-dependent if $\omega \leftrightsquigarrow \theta$ and independent if $\omega \text{ ↲↳ } \theta$.

Now we describe the independence relation (↲↳) by means of the direct product of groups. Let H_ω denote a group that acts transitively on ω and $G \diamond H$ the set of permutations generated by multiplying each permutation (except the identity) of G by each permutation (except the identity) of H.

Lemma 3 *For $\omega, \theta \in \Omega_{G_P}$, if there is a subgroup $H \leq G_P[\omega \cup \theta]$ such that $H = H_\omega \times H_\theta$, then $\omega \text{ ↲↳ } \theta$.*

Proof Assume that there is such a group H. Applying the definition of direct product of groups, we obtain that

$$H_\omega \cup H_\theta \cup H_\omega \diamond H_\theta \leq G_P[\omega \cup \theta].$$

Moreover, we know that $G_P[\omega \cup \theta] = \phi_{\omega\cup\theta}(\Gamma^\omega \cup \Gamma^\theta) \cup \{e\}$. Using elementary set theory, we can write that $\Gamma^\theta \cup \Gamma^\omega = (G^\theta \cap \Gamma^\omega) \cup (G^\omega \cap \Gamma^\theta) \cup (\Gamma^\omega \cap \Gamma^\theta)$. Thus $\phi_{\omega\cup\theta}(\Gamma^\omega \cup \Gamma^\theta) = \phi_{\omega\cup\theta}(G^\theta \cap \Gamma^\omega) \cup \phi_{\omega\cup\theta}(G^\omega \cap \Gamma^\theta) \cup \phi_{\omega\cup\theta}(\Gamma^\omega \cap \Gamma^\theta) = \phi_\omega(G^\theta \cap \Gamma^\omega) \cup \phi_\theta(G^\omega \cap \Gamma^\theta) \cup \phi_{\omega\cup\theta}(\Gamma^\omega \cap \Gamma^\theta)$ since $G^\theta \cap \Gamma^\omega$ stabilizes θ and $G^\omega \cap \Gamma^\theta$ stabilizes ω; we then have that

$$G_P[\omega \cup \theta] = \phi_\omega(G^\theta \cap \Gamma^\omega) \cup \phi_\theta(G^\omega \cap \Gamma^\theta) \cup \phi_{\omega\cup\theta}(\Gamma^\omega \cap \Gamma^\theta) \cup \{e\}.$$

Comparing both expressions involving $G_P[\omega \cup \theta]$, we get by inclusion that $H_\omega \leq \phi_\omega(G^\theta \cap \Gamma^\omega) \cup \{e\}$, $H_\theta \leq \phi_\theta(G^\omega \cap \Gamma^\theta) \cup \{e\}$ and $H_\omega \diamond H_\theta \leq \phi_{\omega\cup\theta}(\Gamma^\omega \cap \Gamma^\theta)$. Thus $G^\theta \cap \Gamma^\omega \neq \varnothing$, $G^\omega \cap \Gamma^\theta \neq \varnothing$ and $\Gamma^\omega \cap \Gamma^\theta \neq \varnothing$ since H_ω and H_θ are nontrivial groups by assumption, meaning that premise (3) in Theorem 1 holds. Moreover, $G^\theta \cap \Gamma^\omega$ acts transitively on ω and $G^\omega \cap \Gamma^\theta$ on θ, which means that the action of the stabilizers G^θ and G^ω is also transitive on ω and θ, respectively, since $G^\theta \cap \Gamma^\omega \leq G^\theta$ and $G^\omega \cap \Gamma^\theta \leq G^\omega$. Thus $\omega \text{ ↲↳ } \theta$. $\square$

Note the similarity between the conditions presented in Lemma 3 and the coprime conditions (Section 3.2), the latter being more restrictive. However, from a computational point of view, to the best of our knowledge, there is no method available in the literature capable of efficiently finding a subgroup $H \leq G[\omega \cup \theta]$ satisfying Lemma 3 for given orbits $\omega, \theta \in \Omega_{G_P}$. This is why we resort to characterize the OI conditions via pointwise stabilizers.

Following, we extend the dependence relations presented above to sets of orbits. In this sense, consider a set $\Omega \subseteq \Omega_{G_P}$ and let $\Omega^\omega = \Omega \setminus \omega$ for $\omega \in \Omega$. The pointwise stabilizer of a set Ω of orbits is denoted as G^Ω hereafter. We look at what happens to ω when the set Ω^ω is pointwise stabilized (i.e. when all the orbits in Ω^ω are simultaneously pointwise stabilized) and, as previously, state suitable dependence definitions.

Definition 5 The orbit ω is:

(a) dependent of Ω^ω (denoted by $\omega \hookrightarrow \Omega^\omega$) if , for any subset $\sigma \subseteq \omega$, $\sigma \notin \Omega_{G^{\Omega^\omega}}$;
(b) semi-dependent of Ω^ω (denoted by $\omega \rightsquigarrow \Omega^\omega$) if there is at least one subset $\sigma \subsetneq \omega$ such that $\sigma \in \Omega_{G^{\Omega^\omega}}$;
(c) independent of Ω^ω (denoted by $\omega \not\hookrightarrow \Omega^\omega$) if $\omega \in \Omega_{G^{\Omega^\omega}}$.

Lemma 4 establishes necessary conditions to have $\omega \not\hookrightarrow \Omega^\omega$.

Lemma 4 *If $\omega \not\hookrightarrow \Omega^\omega$, then $\omega \Rsh \theta$ for all $\theta \in \Omega^\omega$.*

Proof By definition, $\omega \not\hookrightarrow \Omega^\omega$ implies that the action of G^{Ω^ω} on ω is transitive. Since G^{Ω^ω} is a subgroup of G^θ for every $\theta \in \Omega^\omega$, G^θ also acts transitively on ω and thus $\omega \Rsh \theta$. $\square$

Now we can define an independent set of orbits. Note that similar definitions can be laid down as for dependent and semi-dependent sets of orbits.

Definition 6 A set Ω of orbits is said to be independent if $\omega \not\hookrightarrow \Omega^\omega$ for all $\omega \in \Omega$.

Corollary 1 provides necessary conditions so as to a set of orbits be independent.

Corollary 1 *If the set Ω of orbits is independent, then $\omega \Lsh\!\Rsh \theta$ for all $\omega, \theta \in \Omega$.*

Proof By Definition 6 and Lemma 4. $\square$

And Example 2 proves that the conditions in Corollary 1 are not sufficient to guarantee that a set Ω of orbits is independent.

Example 2 Let P be the following $\mathbb{MILP}$:

$$\min_{x \in \{0,1\}^6} \quad x_1 + x_2 + 2x_3 + 2x_4 + 3x_5 + 3x_6$$
$$\begin{pmatrix} 1&1&0&0&0&0 \\ 0&0&1&1&0&0 \\ 0&0&0&0&1&1 \\ 1&0&1&0&1&0 \\ 0&1&0&1&1&0 \\ 1&0&0&1&0&1 \\ 0&1&1&0&0&1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} \leq \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}.$$

The set $\mathcal{G}(P) = \{(1,0,1,0,0,1), (0,1,0,1,0,1), (0,1,1,0,1,0), (1,0,0,1,1,0)\}$ contains its optima. It has formulation group $G_P = \langle (1\ 2)(3\ 4), (3\ 4)(5\ 6) \rangle$, which induces the orbits $\Omega_{G_P} = \{\omega_1, \omega_2, \omega_3\} = \{\{1,2\}, \{3,4\}, \{5,6\}\}$. It is easy to see that the elements in Ω_{G_P} are pairwise independent ($\omega_1 \Lsh\!\Rsh \omega_2 \ \wedge\ \omega_1 \Lsh\!\Rsh \omega_3 \ \wedge\ \omega_2 \Lsh\!\Rsh \omega_3$).

Consider for instance a restriction of the coefficient matrix to the columns indexed by orbits ω_1 and ω_2:

$$R = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

The action of permutation (1 2) on the columns of R is equivalent to the action of permutation (1 4)(2 3) on the rows of R, both resulting in the following matrix:

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}.$$

Since mathematical programs are invariant under constraint-order permutations, we conclude that $\omega_1 \not\hookrightarrow \omega_2$. The same argument is valid for the other cases. Nonetheless, $\omega_1 \hookrightarrow \{\omega_2, \omega_3\}$, $\omega_2 \hookrightarrow \{\omega_1, \omega_3\}$ and $\omega_3 \hookrightarrow \{\omega_1, \omega_2\}$. In order to see this, we consider the full coefficient matrix (i.e. the three orbits simultaneously) and let the permutation (1 2) act on its columns; it yields the matrix:

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Note that in this case, there is no permutation of the rows of the coefficient matrix that produces the same matrix above, which means that we obtain a different set of constraints (i.e. a different mathematical program) when permuting the columns indexed by ω_1 only. On the other hand, the action of (1 2)(5 6) on the columns of the coefficient matrix is equivalent to the action of (1 4)(2 3) on its rows, which means that we must permute the columns indexed by ω_3 and ω_1 simultaneously in order to obtain the original set of constraints; thus $\omega_1 \hookrightarrow \{\omega_2, \omega_3\}$. The same argument is valid for the other cases.

3.4 SBCs from independent sets

Let Ω_I denote an independent set of orbits. Similarly to the results presented in [16], the following propositions set appropriate conditions to build weak and strong SBCs, respectively, from independent sets of orbtis.

Proposition 1 *The constraints* (2) *are SBCs for P and $G^{\Omega_I^\omega}$ with respect to $\omega \in \Omega_I$.*

Proof Let $y \in \mathscr{G}(P)$. Since $G^{\Omega_I^\omega}$ acts transitively on ω, there exists $\pi \in G^{\Omega_I^\omega}$ mapping $\min y[\omega]$ to $y_{\omega(1)}$. □

Proposition 2 *Provided that $G^{\Omega_I^\omega}[\omega] = \mathsf{Sym}(\omega)$, the constraints* (1) *are SBCs for P and $G^{\Omega_I^\omega}$ with respect to $\omega \in \Omega_I$.*

Proof Let $y \in \mathscr{G}(P)$. Since $G^{\Omega_I^\omega}[\omega] = \mathsf{Sym}(\omega)$, there exists $\pi \in G^{\Omega_I^\omega}$ such that $(\pi y)[\omega]$ is ordered by $\leq$. Thus πy is feasible w.r.t. contraints (1). □

4 Orbital independence algorithm

In this section we show how to solve the problem of finding an independent set of orbits of a mathematical program via a classical combinatorial optimization problem, and describe the algorithm proposed to build SBCs from this set. We also conclude our theoretical development by proving our main result concerning independent sets of orbits.

4.1 Independence graph

Our goal is to find the largest possible $\Omega_I \subseteq \Omega_{G_P}$. So far we do not have theoretical results providing sufficient conditions to find such a set. Yet we can use the necessary conditions provided by Corollary 1 and search for the largest set $\Omega_K \subseteq \Omega_{G_p}$ whose elements are pairwise independent. Having obtained Ω_K, we can then search for the largest $\Omega_I \subseteq \Omega_K$. We propose to find Ω_K by solving the problem of finding the maximum clique in the (as of now called) independence graph of P.

Definition 7 The independence graph of P is an undirected graph that encodes the independence relation between orbits of G_P, i.e., an undirected graph $G_I = (V, E)$ where $V = \Omega_{G_P}$ and E is the set of pairwise independent (⫝) orbits of Ω_{G_P}.

4.2 OI reformulations

We expect that the larger the number of SBCs adjoined to the original formulation, the stronger their computational impact. The larger the number of strong SBCs, the better. In fact it remains an open question what is the best trade-off in terms of computational impact: to add many weak, few strong or a mix of both SBCs to the same original formulation? In trying to shed some light on this matter, we propose two reformulations based on the concept of OI: the first prioritizing the total number of SBCs generated and the second prioritizing the total number of strong SBCs generated. In this sense, we look for cliques in G_I that either involve large orbits or mostly orbits which may satisfy the conditions to build strong SBCs.

In order to find such cliques, we associate a weight function $w : V \to W$ to $G_I = (V, E, w)$ and solve the Maximum Weight Clique Problem (MWCP) for G_I using the MP formulation described in [3]. In the first reformulation, which we call *orbital independence narrowing*, we have $W = \{|\omega_1|, \ldots, |\omega_{|V|}|\}$ and $w(\omega_i) = |\omega_i|$ for all $\omega_i \in V$. In the second, which we call *strong orbital independence narrowing*, we consider two different weight values $W = \{w_1, w_2\}$ with $w_2 > w_1 > 0$, and assign w_1 to orbits which generate weak SBCs and w_2 to orbits which generate strong SBCs.

4.3 Algorithm description

The Algorithm 1 generates a system C of compatible SBCs derived from the largest independent set of orbits of P. It takes as inputs a nontrivial formulation group (parameter G_P) and a reformulation strategy (parameter ς). The following functions simplify the pseudocode of Alg. 1: $\mathsf{computeOrbits}(G_P)$ returns the orbits of the group G_P; $\mathsf{computePointStab}(\omega)$ returns the pointwise stabilizer of orbit ω; $\mathsf{pos}(\omega)$ returns the position of orbit ω in the list Ω_{G_P}; $\mathsf{isTransitive}(G, \omega)$ returns true if the action of the group G is transitive on the orbit ω and false otherwise; $\mathsf{buildGraph}(V, E, \varsigma)$ returns a graph with vertices V, edges E and weights appropriate to the strategy ς; $\mathsf{solveMWCP}(G_I)$ returns a solution of the MWCP for the graph G_I. We remark that the functions $\mathsf{computeOrbits}(G_P)$, $\mathsf{computePointStab}(\omega)$ and $\mathsf{isTransitive}(G, \omega)$ are built-in functions available in the software package we use to carry out all group-related computations (see Section 5.3).

Algorithm 1 Orbital Independence SBC generator

```
Require: nontrivial G_P and reformulation strategy ς
Let C = ∅ and Ω_I = ∅
Let Ω_{G_P} = computeOrbits(G_P)
if |Ω_{G_P}| > 1 then
    for ω ∈ Ω_{G_P} do
        Let G^ω = computePointStab(ω)
        for θ ∈ Ω_{G_P} such that pos(θ) > pos(ω) do
            Let G^θ = computePointStab(θ)
            if isTransitive(G^ω, θ) ∧ isTransitive(G^θ, ω) then
                Let E = E ∪ {{ω, θ}, {θ, ω}}
            end if
        end for
    end for
    if |E| ≥ 2 then
        Let G_I = buildGraph(Ω_{G_P}, E, ς)
        Let Ω_K = Ω_I = solveMWCP(G_I)
        for ω ∈ Ω_K do
            if not isTransitive(G^{Ω_I^ω}, ω) then
                Let Ω_I = Ω_I ∖ ω
            end if
        end for
        for ω ∈ Ω_I do
            Let g(x[ω]) ≤ 0 be SBCs satisfying either Proposition 1 or 2
            Let C = C ∪ {g(x[ω]) ≤ 0}
        end for
    end if
end if
return C
```

If G_P has more than one orbit ($|\Omega_{G_P}| > 1$), the algorithm first iteratively looks for all the pairs of independent orbits to build the set E. Provided that the premise

(3) in Theorem 1 is not sufficient to ascertain whether two orbits $\omega, \theta \in \Omega_{G_P}$ satisfy $\omega \mathrel{↲\!↳} \theta$, the algorithm does not compare the sets Γ^{ω} and Γ^{θ} but rather directly checks whether the action of the stabilizers G^{ω} and G^{θ} is transitive on θ and ω, respectively. Testing transitivity is essential since it allows us to identify whether a set X is an orbit of a group G or not (see Section 2.1): in our context, if G^{ω} acts transitively on θ, then $\theta \mathrel{↳} \omega$ and if G^{θ} acts transitively on ω, then $\omega \mathrel{↳} \theta$. In this case, we can add the edge (ω, θ) to E since $\omega \mathrel{↲\!↳} \theta$.

Following the first loop, if at least one pair of independent orbits is found ($|E| \geq 2$), the algorithm builds the independence graph G_I according to the reformulation strategy ς and calls a third party Mixed-Integer Linear Programming solver to solve the MWCP for G_I. Once Ω_K is known, the algorithm converges to a set Ω_I by iteratively removing (from a copy of Ω_K stored as Ω_I) the orbits that do not satisfy $\omega \looparrowright \Omega_I^{\omega}$.

Remark 3 Our approach here is not optimal since the resulting Ω_I may not be the largest one: evaluating all possible $\Omega_I \subseteq \Omega_k$ would require a huge computational effort owing to many stabilizer computations.

Then, for each orbit in Ω_I, the algorithm builds and adds SBCs to the set C. It is relevant to emphasize that if $|\Omega_{G_P}| = 1$ (unique orbit) or $|E| = 0$ (no pair of independent orbits in Ω_{G_P}), no reformulation is carried out.

Theorem 2 proves that any system of SBCs generated by Algorithm 1 for a given G_P is a system of compatible SBCs for problem P, or in other words, it proves that independent sets of orbits embed sufficient conditions to exploit symmetries from two or more distinct orbits simultaneously.

Theorem 2 *The constraint set* $C_{\Omega_I} = \{g(x[\omega_k]) \leq 0 \mid \omega_k \in \Omega_I\}$ *is a system of compatible SBCs for* P.

Proof If P is infeasible then adjoining the constraints in C_{Ω_I} to P does not change its infeasibility, so assume P is feasible. Since $g(x[\omega_k]) \leq 0$ are SBCs for P and $G^{\Omega_I^{\omega_k}}$ with respect to ω_k (i.e. satisfying either Proposition 1 or 2), there exist $y \in \mathscr{G}(P)$ and $\pi_{\omega_k} \in G^{\Omega_I^{\omega_k}}$ such that $\pi_{\omega_k} y$ satisfies $g((\pi_{\omega_k} y)[\omega_k]) \leq 0$. But $\pi_{\omega_k} \in G_P$ for all $\omega_k \in \Omega_I$ and, due to the closure of the group operation, there exists $\pi \in G_P$ such that $\pi = \prod \pi_{\omega_k}$. So $\pi y \in \mathscr{G}(P)$. But $\pi[\omega_k] = \pi_{\omega_k}[\omega_k]$ since $\pi_{\omega_{k'}}$ stabilizes ω_k pointwise for every $k' \neq k$ and thus $(\pi y)[\omega_k] = (\pi_{\omega_k} y)[\omega_k]$. Therefore πy satisfies $g((\pi y)[\omega_k]) \leq 0$ for all $\omega_k \in \Omega_I$. □

Finally, we would like to remark that the algorithms presented in [16] (coprime), [19] (stabilizer's chain) and Algorithm 1, with high probability, generate different sets of SBCs for the same mathematical program. The main reason for this is the fact that the coprime and the stabilizer's chain algorithms perform arbitrary orbit picks on every execution. As pointed out in Remark 2, for the same mathematical program, different runs of the same algorithm may result in different sets of SBCs. Moreover, for a given orbit, the order used to generate the SBCs is also randomly choosen by each algorithm, each choice leading to a different set of SBCs (see Remark 1). An exception would be the case where the stabilizer's chain algorithm luckily picks, at every iteration, the orbits that constitute the independent set found by Algorithm 1 (on top of the respective order for each orbit), what is unlike to happen as the number of orbits $|\Omega_{G_P}|$ increases.

5 Computational experiments

In this section we show the computational impact on the resolution of symmetric $\mathbb{MILP}$s, $\mathbb{MINLP}$s and $\mathbb{BQP}$s when adjoining SBCs from independent sets of orbits. We describe the computational environment involved and analyze the results obtained from the conducted experiments.

5.1 Symmetric BQP

First we define the symmetric Binary Quadratic Programs used in our computational experiments. We are interested in $\mathbb{BQP}$s in the form:

$$\left.\begin{array}{rl} \min\limits_{x} & x^\top A_0 x \\ \forall i \in \mathcal{I}_E & {a_i}^\top x = b_i, \\ & x \in \{0,1\}^n. \end{array}\right\} \tag{3}$$

where A_0 denotes a $n \times n$ real (possibly indefinite) symmetric matrix, a_i denotes a vector of dimension n for all $i \in \mathcal{I}_E$, b denotes a vector of dimension $|\mathcal{I}_E|$ and x represents a n dimensional vector of binary decision variables.

Let $\mathbf{1}$ denote the n dimensional all-ones vector. The first definition relates to the feasible region.

Definition 8 The feasible region has one single equality constraint of the type $\mathbf{1}^\top x = \lceil n/2 \rceil$.

Being invariant to permutations, the constraint in Definition 8 allows us to specify the structure of the formulation groups by controlling the strutucture of the matrix A_0 alone, which is defined next.

Definition 9 A_0 is a block diagonal matrix.

Recall that the action of the formulation group on the index set defines a partition; and every member of the partition which has two or more indices is an orbit. We use this observation in our (quite simple) generation procedure: it first divides the indices of the decision variables into a partition $\mathcal{P}$, and then randomly decides whether each subset $s \in \mathcal{P}$ shall become an orbit or not (all according to some input data provided by the user). Each s corresponds to a block in the matrix A_0. If s is an orbit, the entries of the block B_s are computed by sampling a pair (z_1, z_2) of natural numbers and defining

$$B_s = \begin{cases} z_1 + (|s|-1)z_2 & \text{if } i = j, \\ -z_2 & \text{if } i \neq j. \end{cases} \tag{4}$$

These blocks are Diagonal Dominant matrices. Otherwise (s is not an orbit), the entries of the block B_s are computed by sampling a $(|s| \times |s|)$-matrix M_s and defining

$$B_s = {M_s}^\top M_s. \tag{5}$$

These blocks are Gram matrices. Since all blocks of A_0 are Positive Semidefinite (PSD), the matrix A_0 is PSD as well and the continuous relaxations of the $\mathbb{BQP}$s are convex.

Definition 10 The blocks of A_0 are build according to Eq. (4) for orbits and according to Eq. (5) otherwise.

When the three definitions above are put together, they induce the following symmetry properties on the formulation group of the $\mathbb{BQP}$s: (a) $\Omega_I = \Omega_{G_P}$ and (b) $G_P[\omega] = \mathsf{Sym}(\omega)$ for every orbit $\omega \in \Omega_{G_P}$. These conditions allow us to concurrently use SBCs derived from all orbits of these $\mathbb{BQP}$s. Example 3 illustrates one of these programs.

Example 3 Let P be the following $\mathbb{BQP}$:

$$\min_{x\in\{0,1\}^9} \quad x^\top A_0 x$$
$$\mathbf{1}^\top x = 5,$$

where

$$A_0 = \begin{pmatrix} 6 & -3 & -3 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & 6 & -3 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & -3 & 6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 12 & -6 & -6 & 0 & 0 & 0 \\ 0 & 0 & 0 & -6 & 12 & -6 & 0 & 0 & 0 \\ 0 & 0 & 0 & -6 & -6 & 12 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 1 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 5 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & -2 & 5 \end{pmatrix}.$$

Since $n = 9$, it is easy to see that it satisfies Def. 8: $\lceil 9/2 \rceil = 5$. It is also clear that Def. 9 is satisfied since A_0 has a block diagonal shape. The indices are partioned into $\mathcal{P} = \{\{1,2,3\},\{4,5,6\},\{7,8,9\}\}$. The first two subsets $(\{1,2,3\},\{4,5,6\})$ are choosen to become orbits, and the pairs $(0,3)$ and $(0,6)$ are used to build the blocks associated to them, respectively. As concerns subset $\{7,8,9\}$, the matrix

$$M_s = \begin{pmatrix} 1 & 1 & -1 \\ 2 & 0 & 1 \\ 0 & 1 & -2 \end{pmatrix}$$

is used to build its block. As a result, $\Omega_{G_P} = \{\omega_1, \omega_2\} = \{\{1,2,3\},\{4,5,6\}\}$, $\omega_1 \perp\!\!\!\perp \omega_2$ holds, and the transitive constituent of both orbits is isomorphic to S_3.

These BQP instances are particular cases of the Binary Quadratic Knapsack Problem (BQKP) [12] where (a) each orbit represents a set of identical objects, (b) all objects have the same size, (c) the knapsack has size $\lceil n/2 \rceil$, and (d) a pick within an orbit (variable set to 1) does not affect a pick within a different orbit cost-wise (because of the block-diagonal structure of matrix A_0). As an example, for $n = 23$ and $|\Omega_{G_P}| = 3$, orbit ω_1 could represent ten apples, orbit ω_2 seven oranges and orbit ω_3 six pears; all apples, oranges and pears would have size one in our case, and the knapsack would have size $\lceil 23/2 \rceil = 12$. However, since we are not using any of the classical BQKP formulations, we prefer to present them simply as symmetric BQP instances.

Moreover, we could not find BQP instances in public libraries where Definitions 8, 9 and 10 occur simultaneously. Since we could not assume that such an application does not exist in practice, we designed these highly symmetric instances to at least provide guidelines and help readers to decide whether to employ or not the OI theory when solving the application they have in hands. By explicitly showcasing the symmetry properties under which the OI theory performs usefully, we highlight what sort of characteristics should be looked for in mathematical programs.

5.2 Datasets

Our test bed consists of three groups of instances: (a) symmetric $\mathbb{MILP}$s found in MIPLIB2010; (b) symmetric $\mathbb{MINLP}$s found in MINLPLib2 and (c) symmetric $\mathbb{BQP}$s generated via procedure described previously.

We found 89 instances within the public libraries, 47 from MIPLIB2010 and 42 from MINLPLib2. We refer to [26] for a detailed description of the presence of symmetries in public MP instances. The third group contains a total of 74 convex medium-sized $\mathbb{BQP}$s, named as bqp_n_oxs, where n represents the number of variables, o the number of orbits and s the orbits' size (R means random sizes).

5.3 Environment

The reformulations were obtained on a 4-CPU Intel Xeon at 2.66GHz with 24Gb RAM. Automatic group detection is carried out using ROSE [18] and TRACES [24]. Other group computations are carried out using GAP v. 4.7.4 [27]. The MP results were obtained on a 24-CPU Intel Xeon at 2.53GHz with 48Gb RAM. We used CPLEX 12.6 [13] to solve the $\mathbb{MILP}$s and the $\mathbb{BQP}$s, and SCIP 3.0.1 [1] to solve the $\mathbb{MINLP}$s, all under the AMPL [9] environment. The $\mathbb{BQP}$ generator was coded in Python 2.7.

The computation time was limited to 7200 seconds of user cpu time. In order to try and provide a fair assessment of our methodology, we disabled the symmetry handling methods built into CPLEX and ran it in single thread mode. SCIP does not contain internal symmetry handling methods.

5.4 MILP and MINLP Results

We first comment the results of the reformulation process. Tables 1 and 2 report, per instance, the number of variables (n) and orbits ($|\Omega_{G_P}|$) of the original formulation, and the number of variables indexed by orbits Ω_{G_P} (#svar); for each OI narrowing type, they report the maximum clique ($|\Omega_K|$) and the largest independent set ($|\Omega_I|$) sizes, the number of variables indexed by orbits Ω_I (#var), the number of weak (#wea) and strong (#str) SBCs, and the parameters σ, ρ and υ (described later on).

	Original formulation			OI-narrowing							
Instance	n	$\|\Omega_{G_P}\|$	#svar	$\|\Omega_K\|$	$\|\Omega_I\|$	#var	#wea	#str	σ	ρ	υ
bab5	21600	1936	3872	4	4	8	0	4	.17	0*	0*
blp-ar98	16017	2	4	2	2	4	0	2	0*	1.00	1.00
blp-ic97	8445	2	4	2	2	4	0	2	0*	1.00	1.00
core2536-691	15288	88	187	12	12	29	3	14	.01	.13	.15
core4872-1529	24605	505	1046	46	46	96	0	50	.04	.09	.09
gmu-35-40	842	40	111	4	4	13	0	9	.13	.10	.11

gmu-35-50	1177	40	111	4	4	13	0	9	.09	.10	.11
gmut-75-50	36164	64	242	6	6	19	0	13	0*	.09	.07
gmut-77-40	13140	70	280	7	7	26	0	19	.02	.10	.09
iis-bupa-cov	345	2	7	2	2	7	0	5	.02	1.00	1.00
lectsched-4-obj	3513	267	557	17	17	36	0	19	.15	.06	.06
macrophage	2260	251	566	18	18	42	5	19	.25	.07	.07
map06	46015	107	245	10	10	20	0	10	0*	.09	.08
map10	46015	107	245	10	10	20	0	10	0*	.09	.08
map14	46015	107	245	10	10	20	0	10	0*	.09	.08
map18	46015	107	245	10	10	20	0	10	0*	.09	.08
map20	46015	107	245	10	10	20	0	10	0*	.09	.08
mcsched	1669	45	90	15	15	30	0	15	.05	.33	.33
mzzv11	10240	155	310	16	16	32	0	16	.03	.10	.10
neos-1311124	1092	52	1092	4	4	84	0	80	1.00	.07	.07
neos-1426635	520	52	520	4	4	40	0	36	1.00	.07	.07
neos-1426662	832	52	832	4	4	64	0	60	1.00	.07	.07
neos-1436709	676	52	676	4	4	52	0	48	1.00	.07	.07
neos-1440460	468	52	468	4	4	36	0	32	1.00	.07	.07
neos-1442119	728	52	728	4	4	56	0	52	1.00	.07	.07
neos-1442657	624	52	624	4	4	48	0	44	1.00	.07	.07
neos-555424	3815	132	3810	8	8	190	107	75	.99	.06	.04
neos-826841	5516	156	5436	3	3	200	191	6	.98	.01	.03
neos-849702	1737	128	1737	2	2	36	34	0	1.00	.01	.02
neos-911880	888	259	888	7	7	24	0	17	1.00	.02	.02
neos-952987	31329	37	81	4	4	8	0	4	0*	.10	.09
neos18	963	53	248	5	5	26	0	21	.25	.09	.10
ns1631475	22696	105	210	11	11	22	0	11	0*	.10	.10
ns2081729	661	300	600	3	3	6	0	3	.90	.01	.01
p2m2p1m1p0n100	100	25	92	3	3	12	0	9	.92	.12	.13
protfold	1835	558	1800	2	2	4	0	2	.98	0*	0*
rocII-4-11	3409	2	27	2	2	27	0	25	0*	1.00	1.00
rococoC10-001000	2566	41	82	4	4	8	0	4	.03	.09	.09
rvb-sub	33765	113	226	12	12	24	0	12	0*	.10	.10
satellites1-25	9013	200	400	20	20	40	0	20	.04	.10	.10
seymour-disj-10	1209	49	106	5	5	12	0	7	.08	.10	.11
seymour	1255	55	156	5	5	41	29	7	.12	.09	.26
swath	6404	21	163	2	2	8	0	6	.02	.09	.04
transportmoment	9099	85	189	17	17	38	0	21	.02	.20	.20
toll-like	2883	386	1091	26	26	91	44	21	.37	.06	.08
uc-case3	36921	2687	5374	2	2	4	0	2	.14	0*	0*
uct-subprob	2236	136	306	7	7	14	0	7	.13	.05	.04

	Original formulation			SOI-narrowing							
Instance	n	$\lvert\Omega_{G_P}\rvert$	#svar	$\lvert\Omega_K\rvert$	$\lvert\Omega_I\rvert$	#var	#wea	#str	σ	ρ	υ
core2536-691	15288	88	187	12	12	27	0	15	.01	.13	.14
macrophage	2260	251	566	18	18	39	0	21	.25	.07	.06
neos-555424	3815	132	3810	8	8	145	58	79	.99	.06	.03
neos-826841	5516	156	5436	4	4	46	0	42	.98	.02	0*
neos-849702	1737	128	1737	2	2	9	0	7	1.00	.01	0*
toll-like	2883	386	1091	26	26	59	0	33	.37	.06	.05

Table 1: OI-narrowings of symmetric instances from MIPLIB2010. 0* indicates values of $O(10^{-3})$ or less.

	Original formulation			OI-narrowing							
Instance	n	$\lvert\Omega_{G_P}\rvert$	#svar	$\lvert\Omega_K\rvert$	$\lvert\Omega_I\rvert$	#var	#wea	#str	σ	ρ	υ
arki0002	2456	384	2304	2	2	12	0	10	.93	0*	0*
arki0005	2370	9	18	9	9	18	0	9	0*	1.00	1.00
arki0006	2370	9	18	9	9	18	0	9	0*	1.00	1.00
autocorr_bern25-03	26	12	24	2	2	4	0	2	.92	.16	.16
carton7	230	49	162	3	3	13	8	2	.70	.06	.08
carton9	266	83	266	3	3	13	8	2	1.00	.03	.04
cecil_13	733	18	36	9	9	18	0	9	.04	.50	.50
chp_partload	2080	82	164	5	5	10	0	5	.07	.06	.06
crudeoil_li21	1236	134	268	2	2	4	0	2	.21	.01	.01
ex9_2_6	16	7	16	2	2	6	3	1	1.00	.28	.37
gastrans	89	6	12	2	2	4	0	2	.13	.33	.33
hmittelman	16	3	6	3	3	6	0	3	.37	1.00	1.00
kport20	98	25	55	5	5	11	0	6	.56	.20	.20
kport40	217	48	150	8	8	28	0	20	.69	.16	.18
lop97ic	1626	3	127	3	3	127	0	124	.07	1.00	1.00
lop97icx	986	8	777	8	8	777	0	769	.78	1.00	1.00
mbtd	210	61	210	2	2	12	9	1	1.00	.03	.05

	Original formulation			SOI-narrowing							
Instance	n	$\lvert\Omega_{G_P}\rvert$	#svar	$\lvert\Omega_K\rvert$	$\lvert\Omega_I\rvert$	#var	#wea	#str	σ	ρ	υ
netmod_kar1	456	48	132	3	3	9	0	6	.28	.06	.06
netmod_kar2	456	48	132	3	3	9	0	6	.28	.06	.06
powerflow2383wpr	15882	12	24	3	3	6	0	3	0*	.25	.25
powerflow2383wpp	15882	12	24	3	3	6	0	3	0*	.25	.25
risk2bpb	434	12	72	12	12	72	0	60	.16	1.00	1.00
routingdelay_bigm	1115	18	36	12	12	24	0	12	.03	.66	.66
routingdelay_proj	1115	18	36	12	12	24	0	12	.03	.66	.66
sepasequ_complex	485	5	27	5	5	27	9	13	.05	1.00	1.00
st_rv9	50	10	20	10	10	20	0	10	.40	1.00	1.00
super1	1263	12	26	12	12	26	0	14	.02	1.00	1.00
super2	1274	11	24	11	11	24	0	13	.01	1.00	1.00
super3	1281	11	24	11	11	24	0	13	.01	1.00	1.00
super3t	1032	11	24	11	11	24	0	13	.02	1.00	1.00
syn15m	55	2	5	2	2	5	0	3	.09	1.00	1.00
torsion100	5004	2	4	2	2	4	0	2	0*	1.00	1.00
torsion25	1254	2	4	2	2	4	0	2	0*	1.00	1.00
torsion50	2504	1227	2454	3	3	6	0	3	.98	0*	0*
torsion75	3754	2	4	2	2	4	0	2	0*	1.00	1.00
transswitch2383wpr	18768	15	30	3	3	6	0	3	0*	.20	.20
transswitch2383wpp	18768	15	30	3	3	6	0	3	0*	.20	.20
turkey	512	4	8	4	4	8	0	4	.01	1.00	1.00
unitcommit1	738	2	30	2	2	30	0	28	.04	1.00	1.00
unitcommit2	738	2	30	2	2	30	0	28	.04	1.00	1.00
waste	1425	30	76	15	15	38	0	23	.05	.50	.50
waterund28	760	106	216	2	2	4	0	2	.28	.01	.01
carton7	230	49	162	3	3	8	0	5	.70	.06	.04
carton9	266	83	266	3	3	8	0	5	1.00	.03	.03

Table 2: OI-narrowings of symmetric instances from MINLPLib2. 0* indicates values of $O(10^{-3})$ or less.

Both reformulation strategies yielded the same narrowings for the most part of the instances. In these cases, we do not present results concerning the SOI reformulation. Recall that Algorithm 1 can yield suboptimal independent sets in terms of size (see Remark 3 in Sect. 4.3). The reformulation results show that the size of the maximum cliques is equal to the size of the largest independent sets for all instances; we thus judge that Algorithm 1 yields good results on average as concerns symmetry detection.

	Original formulation		OI-narrowing		SOI-narrowing	
Dataset	# Best	Time (h)	# Best	Time (h)	# Best	Time (h)
MIPLIB2010	22	49.52	20	48.16	4	48.15
MINLPLib2	14	52.00	17	51.3	2	51.29
Total	36	101.52	37	99.46	6	99.44

Table 3: Aggregated solution statistics for datasets MIPLIB2010 and MINLPLib2.

Table 3 provides aggregated solution statistics. Per dataset and for each formulation, the table reports the number of best performances and the total time comsumed in hours to solve all instances. The statistics are more expressive regarding the MIPLIB2010 library.

Finally, Tables 4 and 5 report details of the optimization results. Per instance and for each formulation, the table exhibits the best solution found, the user cpu time (in seconds), the gap (%), the number of BB nodes and the solver status at termination (opt = optimum found, lim = time limit reached, inf = infeasible instance). Best values are emphasized in boldface. Two intances (namely powerflow2383wpp and transswitch2383wpp) do not appear in Table 5 due to SCIP technical limitations.

We observe that the total computation time of the OI-narrowings is 2 hours inferior (Table 3), which means that we improved overall, despite the factors that play against us (as explained below). Instance-wise the results may not be significant in some cases, but this goes both ways (original formulation vs oi-narrowings): the SBCs slightly helped to improve the performance of the solvers in 43 cases and were detrimental in 36 cases out of 87. Despite of providing good results, the SOI-narrowings did not achieve outstanding performances.

Our investigation indicates that strong OI reformulations occur seldomly in practice (it was found in 9% of the symmetric public instances tested), yet we encourage its use since the computational experiments also show that such narrowings helped to improve the solver's performance in 75% of the cases (6 out of 8 instances). This is a fairly good percentual when compared to the overall performance of the OI reformulations.

Overall, we think that two facts contribute to explain the average-to-poor computational results we have achieved with the public instances. First, apart from the structure of the group G_P, the ratio $\sigma = (\#\text{svar}/n)$ may also indicate how symmetric a formulation P is. Similarly, the ratios $\rho = (|\Omega_I|/|\Omega_{G_P}|)$ and $\upsilon = (\#\text{var}/\#\text{svar})$ may indicate how extensively one has exploited the symmetries of P. All together, we expect SBCs to make a strong computational impact whenever the triplet (σ, ρ, υ) tends to $(1, 1, 1)$. However, Tables 1 and 2 show the two patterns in which the majority of the instances fit into: either the instance is highly symmetric ($\sigma \approx 1$) and we cannot explore much of its symmetries $(\rho, \upsilon) \approx (0, 0)$, or it does not exhibit many symmetries ($\sigma \approx 0$) and we explore almost all of them $(\rho, \upsilon) \approx (1, 1)$. Second, recall that BB type algorithms are complex systems whose performance depend on many factors (Linear Programming (LP) solutions, branching policies, cut generation schemes and so on). The presence of SBCs may change LP solutions computed in the nodes of the BB tree, which means that SBCs can also unduly impact on branching policies and on cut generation schemes. Since there are elements of arbitrary choice regarding the generation of SBCs (recall Remarks 1 and 2), forcing these choices may, in some cases, prevent the BB algorithm to take the correct decisions.

5.5 BQP Results

Again we start off commenting the results related to the OI reformulation process. As the content of Table 6 indicates, the $\mathbb{BQP}$s are highly symmetric. We observe that $(\sigma, \rho, \upsilon) = ([0.5, 1], 1, 1)$ holds for all cases. Moreover, per $\mathbb{BQP}$ generated, every orbit satisfies the conditions in Proposition 2 and thus we could build nothing but strong SBCs for all instances.

	Original formulation			OI-narrowing							
Instance	n	$\|\Omega_{G_P}\|$	#svar	$\|\Omega_K\|$	$\|\Omega_I\|$	#var	#wea	#str	σ	ρ	υ
bqp_70_2xR	70	2	49	2	2	49	0	47	.70	1.00	1.00
bqp_70_3xR	70	3	45	3	3	45	0	42	.64	1.00	1.00
bqp_70_4x14	70	4	56	4	4	56	0	52	.80	1.00	1.00
bqp_70_4xR	70	4	70	4	4	70	0	66	1.00	1.00	1.00
bqp_70_5xR	70	5	58	5	5	58	0	53	.82	1.00	1.00
bqp_70_6x10	70	6	60	6	6	60	0	54	.85	1.00	1.00
bqp_70_7xR	70	7	63	7	7	63	0	56	.90	1.00	1.00
bqp_70_9x7	70	9	63	9	9	63	0	54	.90	1.00	1.00

bqp_75_2x25	75	2	50	2	2	50	0	48	.66	1.00	1.00
bqp_75_2xR	75	2	39	2	2	39	0	37	.52	1.00	1.00
bqp_75_3xR	75	3	60	3	3	60	0	57	.80	1.00	1.00
bqp_75_4x15	75	4	60	4	4	60	0	56	.80	1.00	1.00
bqp_75_4xR	75	4	54	4	4	54	0	50	.72	1.00	1.00
bqp_75_5x15	75	5	75	5	5	75	0	70	1.00	1.00	1.00
bqp_75_5xR	75	5	66	5	5	66	0	61	.88	1.00	1.00
bqp_75_6xR	75	6	67	6	6	67	0	61	.89	1.00	1.00
bqp_75_7xR	75	7	63	7	7	63	0	56	.84	1.00	1.00
bqp_75_8xR	75	8	66	8	8	66	0	58	.88	1.00	1.00
bqp_80_2x20	80	2	40	2	2	40	0	38	.50	1.00	1.00
bqp_80_2xR	80	2	55	2	2	55	0	53	.68	1.00	1.00
bqp_80_3x20	80	3	60	3	3	60	0	57	.75	1.00	1.00
bqp_80_3xR	80	3	64	3	3	64	0	61	.80	1.00	1.00
bqp_80_4x16	80	4	64	4	4	64	0	60	.80	1.00	1.00
bqp_80_4xR	80	4	65	4	4	65	0	61	.81	1.00	1.00
bqp_80_5x16	80	5	80	5	5	80	0	75	1.00	1.00	1.00
bqp_80_5xR	80	5	70	5	5	70	0	65	.87	1.00	1.00
bqp_80_6xR	80	6	72	6	6	72	0	66	.90	1.00	1.00
bqp_80_7x10	80	7	70	7	7	70	0	63	.87	1.00	1.00
bqp_80_8xR	80	8	68	8	8	68	0	60	.85	1.00	1.00
bqp_85_12x5	85	12	60	12	12	60	0	48	.70	1.00	1.00
bqp_85_16x5	85	16	80	16	16	80	0	64	.94	1.00	1.00
bqp_85_2x17	85	2	34	2	2	34	0	32	.40	1.00	1.00
bqp_85_2xR	85	2	59	2	2	59	0	57	.69	1.00	1.00
bqp_85_3xR	85	3	68	3	3	68	0	65	.80	1.00	1.00
bqp_85_4x17	85	4	68	4	4	68	0	64	.80	1.00	1.00
bqp_85_4xR	85	4	67	4	4	67	0	63	.78	1.00	1.00
bqp_85_5xR	85	5	64	5	5	64	0	59	.75	1.00	1.00
bqp_85_6xR	85	6	75	6	6	75	0	69	.88	1.00	1.00
bqp_85_7xR	85	7	76	7	7	76	0	69	.89	1.00	1.00
bqp_85_8xR	85	8	80	8	8	80	0	72	.94	1.00	1.00
bqp_85_9xR	85	9	85	9	9	85	0	76	1.00	1.00	1.00
bqp_90_2x30	90	2	60	2	2	60	0	58	.66	1.00	1.00
bqp_90_2xR	90	2	65	2	2	65	0	63	.72	1.00	1.00
bqp_90_3x30	90	3	90	3	3	90	0	87	1.00	1.00	1.00
bqp_90_3xR	90	3	75	3	3	75	0	72	.83	1.00	1.00
bqp_90_4x18	90	4	72	4	4	72	0	68	.80	1.00	1.00
bqp_90_4xR	90	4	73	4	4	73	0	69	.81	1.00	1.00
bqp_90_5x15	90	5	75	5	5	75	0	70	.83	1.00	1.00
bqp_90_5xR	90	5	77	5	5	77	0	72	.85	1.00	1.00
bqp_90_6xR	90	6	76	6	6	76	0	70	.84	1.00	1.00
bqp_90_7xR	90	7	70	7	7	70	0	63	.77	1.00	1.00
bqp_90_8x10	90	8	80	8	8	80	0	72	.88	1.00	1.00
bqp_90_9x9	90	9	81	9	9	81	0	72	.90	1.00	1.00
bqp_95_18x5	95	18	90	18	18	90	0	72	.94	1.00	1.00
bqp_95_2xR	95	2	51	2	2	51	0	49	.53	1.00	1.00
bqp_95_3xR	95	3	77	3	3	77	0	74	.81	1.00	1.00
bqp_95_4x19	95	4	76	4	4	76	0	72	.80	1.00	1.00
bqp_95_4xR	95	4	90	4	4	90	0	86	.94	1.00	1.00
bqp_95_5xR	95	5	88	5	5	88	0	83	.92	1.00	1.00
bqp_95_6xR	95	6	89	6	6	89	0	83	.93	1.00	1.00
bqp_95_7xR	95	7	95	7	7	95	0	88	1.00	1.00	1.00
bqp_95_8xR	95	8	95	8	8	95	0	87	1.00	1.00	1.00
bqp_95_9xR	95	9	86	9	9	86	0	77	.90	1.00	1.00
bqp_100_2xR	100	2	70	2	2	70	0	68	.70	1.00	1.00
bqp_100_3x25	100	3	75	3	3	75	0	72	.75	1.00	1.00
bqp_100_3xR	100	3	77	3	3	77	0	74	.77	1.00	1.00
bqp_100_4x20	100	4	80	4	4	80	0	76	.80	1.00	1.00
bqp_100_4xR	100	4	81	4	4	81	0	77	.81	1.00	1.00
bqp_100_5x20	100	5	100	5	5	100	0	95	1.00	1.00	1.00
bqp_100_5xR	100	5	93	5	5	93	0	88	.93	1.00	1.00
bqp_100_6xR	100	6	96	6	6	96	0	90	.96	1.00	1.00
bqp_100_7xR	100	7	88	7	7	88	0	81	.88	1.00	1.00
bqp_100_8xR	100	8	89	8	8	89	0	81	.89	1.00	1.00
bqp_100_9x10	100	9	90	9	9	90	0	81	.90	1.00	1.00

Table 6: OI narrowings of symmetric $\mathbb{BQP}$s. 0* indicates values of $O(10^{-3})$ or less.

Table 7 presents the aggregated statistics for the BQP dataset. In the majority of the cases, 51 out of 74, the narrowings performed better, against 21 of the original formulations. Note however the large difference in terms of execution time: more than 11 hours in total for the original problems against less than 17 seconds for

	Original formulation					OI-narrowing				
Instance	Best	Time (s)	Gap (%)	Nodes	St.	Best	Time (s)	Gap (%)	Nodes	St.
bab5	-106412	3879.57	0	86526	opt	-106412	**930.53**	0	19756	opt
blp-ar98	6205.21	**1319.87**	0	100570	opt	6205.21	3736.05	0	242538	opt
blp-ic97	4025.02	5517.71	0	544235	opt	4025.02	**3122.49**	0	292192	opt
core4872-1529	1459	7200.11	**1.70**	11013	lim	1467	7200.14	2.21	13372	lim
gmu-35-40	-2406600	63.59	0	114177	opt	-2406600	**53.12**	0	114177	opt
gmu-35-50	-2607780	7208.25	0.01	**9602254**	lim	-2607780	7209.47	0.01	9649343	lim
gmut-75-50	-14178800	7200.61	0.01	659846	lim	-14179300	6597.76	0	1044318	**opt**
gmut-77-40	-14170700	**3439.64**	0	813820	opt	-14170700	3582.83	0	813820	opt
iis-bupa-cov	36	7200.08	4.34	300463	lim	36	6017.99	0	291341	**opt**
lectsched-4-obj	4	9.10	0	1122	opt	4	**8.05**	0	1508	opt
map06	-289	**705.17**	0	1166	opt	-289	806.19	0	1274	opt
map10	-495	**616.85**	0	1146	opt	-495	699.47	0	1570	opt
map14	-674	684.74	0	1828	opt	-674	**664.73**	0	1781	opt
map18	-847	**322.72**	0	854	opt	-847	328.57	0	1232	opt
map20	-922	**147.98**	0	593	opt	-922	169.07	0	651	opt
mcsched	211913	**317.08**	0	39393	opt	211913	364.21	0	37954	opt
mzzv11	-21718	**20.27**	0	195	opt	-21718	34.15	0	153	opt
neos-1311124	-181	7200.79	0.55	**3726812**	lim	-181	7200.49	0.55	4218649	lim
neos-1426635	-176	7201.01	1.14	6151363	lim	-176	7200.77	**0.57**	5479189	lim
neos-1426662	-44	7200.67	14.40	2044958	lim	-44	7201.45	**12.94**	2643304	lim
neos-1436709	-128	7200.38	0.78	**2371927**	lim	-128	7200.40	0.78	2828927	lim
neos-1440460	-179.25	7200.59	0.42	4026340	lim	-179.25	7200.37	**0.05**	4870972	lim
neos-1442119	-181	7200.35	0.55	1896571	lim	-181	7200.34	0.55	**1808334**	lim
neos-1442657	-154.5	7200.59	0.97	**2672729**	lim	-154.5	7200.40	0.97	2729172	lim
neos-911880	54.76	7.61	0	9755	opt	54.76	**7.12**	0	10045	opt
neos-952987	∞	7200.37	∞	**518621**	lim	∞	7200.32	∞	583213	lim
neos18	13	25.43	0	7125	opt	13	**16.00**	0	4451	opt
ns1631475	21450	7200.10	91.03	534	lim	∞	7200.06	∞	229	lim
ns2081729	9	**391.90**	0	399139	opt	9	815.13	0	848789	opt
p2m2p1m1p0n100	0	0.00	0	0	opt	0	0.00	0	0	opt
protfold	-26	7200.04	37.68	33112	lim	-27	7200.04	**32.19**	34311	lim
rocII-4-11	-5.65564	400.50	0	92664	opt	-5.65564	**385.86**	0	92664	opt
rococoC10-001000	11460	140.67	0	11973	opt	11460	**138.33**	0	11973	opt
rvb-sub	27.4683	7200.48	58.58	266959	lim	27.4683	7200.39	58.56	**201344**	lim
satellites1-25	-5	**191.84**	0	1447	opt	-5	421.96	0	3223	opt
seymour-disj-10	287	7200.08	**1.22**	98205	lim	288	7200.10	1.56	112785	lim
seymour	306	7200.11	**1.35**	201025	lim	307	7200.11	1.69	223602	lim
swath	467.408	7200.37	10.14	1929236	lim	467.408	7200.42	**9.95**	2335854	lim
transportmoment	∞	2.68	∞	0	inf	∞	**2.50**	∞	0	inf
uc-case3	6931.2	7200.36	0.05	**34255**	lim	6931.2	7200.38	0.05	46469	lim
uct-subprob	315	7200.20	**3.29**	356615	lim	315	7200.21	5.02	401113	lim

	Original formulation					OI-narrowing					SOI-narrowing				
Instance	Best	Time (s)	Gap (%)	Nodes	St.	Best	Time (s)	Gap (%)	Nodes	St.	Best	Time (s)	Gap (%)	Nodes	St.
core2536-691	683	56.90	0	1127	opt	683	65.38	0	782	opt	683	**50.90**	0	690	opt
macrophage	374	868.50	0	77147	opt	374	372.68	0	38511	opt	374	**271.38**	0	14009	opt
neos-555424	1286800	**5.76**	0	682	opt	1286800	6.79	0	776	opt	1286800	6.55	0	782	opt
neos-826841	29.0082	7200.13	3.45	881318	lim	29.0082	7200.18	3.45	718724	lim	29.0082	7200.14	3.45	**687082**	lim
neos-849702	0	730.88	0	53661	opt	0	**8.65**	0	140	opt	0	90.03	0	4136	opt
toll-like	614	7200.07	18.52	79283	lim	611	7200.08	17.02	106330	lim	613	7200.07	**16.76**	99629	lim

Table 4: MIPLIB2010 results obtained with CPLEX 12.6.

Instance	Original formulation					OI-narrowing				
	Best	Time (s)	Gap (%)	Nodes	St.	Best	Time (s)	Gap (%)	Nodes	St.
arki0002	12.97	7200.07	∞	1	lim	12.97	7200.06	∞	1	lim
arki0005	10434.4	7200.31	∞	1	lim	10434.4	7200.14	∞	1	lim
arki0006	116.82	7200.22	∞	1	lim	116.82	7200.54	∞	1	lim
autocorr_bern25-03	-92	0.02	0	1	opt	-92	0.02	0	1	opt
cecil_13	-115656	7304.46	2.66	**1750370**	lim	-115656	7300.49	2.66	1815455	lim
chp_partload	∞	7202.05	∞	**11005**	lim	∞	7202.55	∞	11090	lim
crudeoil_li21	∞	7203.18	∞	693902	lim	∞	7202.59	∞	**633908**	lim
ex9_2_6	-1	0.02	0	1	opt	-1	0.02	0	1	opt
gastrans	89.09	0.15	0	1	opt	89.09	**0.11**	0	20	opt
hmittelman	13	0.02	0	1	opt	13	0.02	0	1	opt
kport20	26.91	6489.09	0	5221763	opt	26.91	**4002.44**	0	3376405	opt
kport40	**32.37**	7443.03	35.35	1851112	lim	32.55	7436.03	35.09	2277122	lim
lop97ic	4973.17	7200.83	94.03	164143	lim	4830.18	7201.11	**88.70**	121559	lim
lop97icx	4306	7208.88	49.98	1838898	lim	4323.03	7205.98	**46.45**	2002183	lim
mbtd	10.17	7201.61	306.67	1	lim	8.50	7206.04	**240.00**	1	lim
netmod_kar1	-0.42	10.51	0	651	opt	-0.42	**7.76**	0	316	opt
netmod_kar2	-0.42	10.59	0	651	opt	-0.42	**7.79**	0	316	opt
powerflow2383wpr	∞	7204.88	∞	1781	lim	∞	7204.91	∞	**1384**	lim
risk2bpb	-55.88	**0.11**	0	2	opt	-55.88	0.15	0	11	opt
routingdelay_bigm	146.63	**12.27**	0	30	opt	146.63	13.76	0	38	opt
routingdelay_proj	∞	7201.98	∞	**631303**	lim	∞	7201.87	∞	635105	lim
sepasequ_complex	578.74	7229.26	106.34	363949	lim	492.42	7235.63	**70.75**	272573	lim
st_rv9	-120.15	0.32	0	951	opt	-120.15	**0.22**	0	594	opt
super1	∞	7211.69	∞	53457	lim	∞	7209.83	∞	**52513**	lim
super2	∞	7209.98	∞	**53171**	lim	∞	7212.45	∞	54086	lim
super3	∞	7224.12	∞	65944	lim	∞	7209.58	∞	**64678**	lim
super3t	∞	7209.06	∞	**61355**	lim	∞	7208.77	∞	76369	lim
syn15m	-853.28	0.16	0	9	opt	-853.28	0.16	0	9	opt
torsion25	-0.34	7200.11	**10069.16**	1	lim	0	7200.13	∞	1	lim
torsion50	-0.33	7200.32	**16362.80**	1	lim	0	7200.32	∞	1	lim
torsion75	-0.34	7200.61	**25863.95**	1	lim	0	7200.44	∞	1	lim
torsion100	-0.34	7200.57	**39539.62**	1	lim	0	7200.67	∞	1	lim
transswitch2383wpr	∞	7205.02	∞	674	lim	∞	7205.78	∞	**543**	lim
turkey	1766.82	7200.18	∞	76	lim	1766.82	7200.20	∞	**57**	lim
unitcommit1	578177	2.79	0	5	opt	578177	**2.66**	0	3	opt
unitcommit2	578177	**6.89**	0	33	opt	578177	7.62	0	38	opt
waste	609.13	7216.32	101.88	7448194	lim	609.13	7217.47	101.88	**4805522**	lim
waterund28	∞	7200.37	∞	**465**	lim	∞	7200.84	∞	646	lim

Instance	Original formulation					OI-narrowing					SOI-narrowing				
	Best	Time (s)	Gap (%)	Nodes	St.	Best	Time (s)	Gap (%)	Nodes	St.	Best	Time (s)	Gap (%)	Nodes	St.
carton7	191.73	98.15	0	108287	opt	191.73	92.05	0	94612	opt	191.73	**84.58**	0	83480	opt
carton9	205.137	103.60	0	118816	opt	205.137	97.24	0	108145	opt	205.137	**42.99**	0	47563	opt

Table 5: MINLPLib2 results obtained with SCIP 3.0.1.

	Original formulation		OI-narrowing	
Dataset	# Best	Time (s)	# Best	Time (s)
BQP	21	42054.31	51	16.66

Table 7: Aggregated solution statistics for the BQP dataset.

the OI narrowings. Table 8 exhibits detailed results. All narrowings were solved to optimality. As a side note, these results somehow support out claim that SBCs may eventually prevent BB algorithms to take correct decisions since some narrowings (21 in total) performed worse even under favorable conditions.

	Original formulation					OI-narrowing				
Instance	Best	Time (s)	Gap (%)	Nodes	St.	Best	Time (s)	Gap (%)	Nodes	St.
bqp_70_2xR	580	7212.02	14.69	55721277	lim	580	**0.14**	0	0	opt
bqp_70_3xR	1132	2.04	0	4644	opt	1132	**0.58**	0	613	opt
bqp_70_4x14	427	2.85	0	21397	opt	427	**0.13**	0	79	opt
bqp_70_4xR	648	16.13	0	121652	opt	648	**0.12**	0	120	opt
bqp_70_5xR	197	**0.12**	0	245	opt	197	0.17	0	47	opt
bqp_70_6x10	155	0.68	0	3065	opt	155	**0.13**	0	116	opt
bqp_70_7xR	112	**0.05**	0	163	opt	112	0.09	0	55	opt
bqp_70_9x7	70	**0.02**	0	49	opt	70	0.05	0	29	opt
bqp_75_2x25	763	**0.07**	0	141	opt	763	0.08	0	35	opt
bqp_75_2xR	646	**0.05**	0	55	opt	646	0.11	0	0	opt
bqp_75_3xR	651	**0.06**	0	111	opt	651	0.17	0	66	opt
bqp_75_4x15	949	14.61	0	126111	opt	949	**0.20**	0	71	opt
bqp_75_4xR	981	0.26	0	442	opt	981	**0.11**	0	42	opt
bqp_75_5x15	931	93.00	0	720167	opt	931	**0.15**	0	186	opt
bqp_75_5xR	604	0.84	0	3901	opt	604	**0.20**	0	36	opt
bqp_75_6xR	210	**0.07**	0	132	opt	210	0.12	0	58	opt
bqp_75_7xR	172	**0.06**	0	176	opt	172	0.09	0	54	opt
bqp_75_8xR	100	**0.06**	0	174	opt	100	0.08	0	42	opt
bqp_80_2x20	500	**0.01**	0	0	opt	500	0.02	0	0	opt
bqp_80_2xR	1760	59.90	0	350488	opt	1760	**0.06**	0	0	opt
bqp_80_3x20	100	**0.01**	0	0	opt	100	0.02	0	0	opt
bqp_80_3xR	853	0.25	0	371	opt	853	**0.24**	0	137	opt
bqp_80_4x16	976	75.01	0	549538	opt	976	**0.18**	0	61	opt
bqp_80_4xR	715	3.42	0	20751	opt	715	**0.19**	0	29	opt
bqp_80_5x16	936	27.07	0	206700	opt	936	**0.40**	0	143	opt
bqp_80_5xR	305	0.66	0	1696	opt	305	**0.22**	0	80	opt
bqp_80_6xR	78	**0.04**	0	80	opt	78	0.12	0	30	opt
bqp_80_7x10	170	**0.04**	0	27	opt	170	0.07	0	12	opt
bqp_80_8xR	100	**0.06**	0	59	opt	100	0.09	0	22	opt
bqp_85_12x5	147	1.25	0	5825	opt	147	**0.66**	0	1561	opt
bqp_85_16x5	69	1.76	0	10181	opt	69	**1.06**	0	3740	opt
bqp_85_2x17	2924	0.04	0	24	opt	2924	0.04	0	24	opt
bqp_85_2xR	5189	3.98	0	19265	opt	5189	**0.06**	0	8	opt
bqp_85_3xR	695	649.55	0	4322081	opt	695	**0.10**	0	22	opt
bqp_85_4x17	714	27.94	0	156926	opt	714	**0.15**	0	44	opt
bqp_85_4xR	1374	61.84	0	477419	opt	1374	**0.21**	0	60	opt
bqp_85_5xR	827	1.75	0	1942	opt	827	**0.86**	0	260	opt
bqp_85_6xR	233	1.65	0	2463	opt	233	**0.17**	0	202	opt
bqp_85_7xR	141	0.07	0	172	opt	141	**0.06**	0	64	opt
bqp_85_8xR	160	0.65	0	346	opt	160	**0.06**	0	37	opt
bqp_85_9xR	52	0.14	0	339	opt	52	**0.10**	0	110	opt
bqp_90_2x30	9420	7212.55	35.51	35350170	lim	9420	**0.11**	0	23	opt
bqp_90_2xR	1872	7212.53	32.07	36978133	lim	1872	**0.08**	0	3	opt
bqp_90_3x30	6585	7212.67	34.19	33351114	lim	6585	**0.16**	0	58	opt
bqp_90_3xR	2735	4.49	0	19974	opt	2735	**0.18**	0	12	opt
bqp_90_4x18	576	40.24	0	191424	opt	576	**0.11**	0	23	opt
bqp_90_4xR	1047	3.22	0	14661	opt	1047	**0.22**	0	70	opt
bqp_90_5x15	225	**0.01**	0	0	opt	225	0.03	0	0	opt
bqp_90_5xR	215	0.61	0	372	opt	215	**0.26**	0	89	opt
bqp_90_6xR	183	**0.12**	0	168	opt	183	0.17	0	49	opt
bqp_90_7xR	283	2.90	0	15893	opt	283	**0.55**	0	283	opt
bqp_90_8x10	925	65.20	0	411100	opt	925	**1.17**	0	2345	opt
bqp_90_9x9	117	**0.04**	0	53	opt	117	0.06	0	44	opt
bqp_95_18x5	95	2.22	0	16397	opt	95	**1.40**	0	4329	opt
bqp_95_2xR	4226	3.26	0	14109	opt	4226	**0.11**	0	10	opt
bqp_95_3xR	1843	8.24	0	43460	opt	1843	**0.12**	0	38	opt
bqp_95_4x19	636	29.29	0	188938	opt	636	**0.09**	0	69	opt

bqp_95_4xR	480	0.08	0	9	opt	480	**0.07**	0	12	opt
bqp_95_5xR	468	**0.07**	0	65	opt	468	0.37	0	50	opt
bqp_95_6xR	220	0.05	0	1	opt	220	**0.04**	0	0	opt
bqp_95_7xR	468	1.33	0	232	opt	468	**0.17**	0	27	opt
bqp_95_8xR	1425	3194.67	0	23845041	opt	1425	**0.12**	0	0	opt
bqp_95_9xR	209	1.06	0	2046	opt	209	**0.29**	0	243	opt
bqp_100_2xR	8606	7211.99	43.86	28145015	lim	8606	**0.32**	0	8	opt
bqp_100_3x25	700	0.12	0	71	opt	700	**0.07**	0	0	opt
bqp_100_3xR	6942	186.10	0	1036046	opt	6942	**0.49**	0	67	opt
bqp_100_4x20	4230	1238.27	0	7701248	opt	4230	**0.22**	0	99	opt
bqp_100_4xR	400	1.36	0	5260	opt	400	**0.16**	0	61	opt
bqp_100_5x20	1280	161.01	0	971381	opt	1280	**0.79**	0	167	opt
bqp_100_5xR	884	**0.16**	0	246	opt	884	0.31	0	33	opt
bqp_100_6xR	725	**0.09**	0	61	opt	725	0.23	0	5	opt
bqp_100_7xR	358	**0.10**	0	105	opt	358	0.17	0	25	opt
bqp_100_8xR	294	0.16	0	184	opt	294	**0.14**	0	71	opt
bqp_100_9x10	70	0.02	0	0	opt	70	0.02	0	0	opt

Table 8: BQP results obtained with CPLEX 12.6.

Lastly, we ran a second round of tests restricted to the instances whose original formulations were not solved to optimality, now forcing CPLEX to unleash its full power in terms of symmetry breaking (parameter *symmetry* of CPLEX's API set to level 5). The results are presented in Table 9. Remarkably, there is no significant change in the final gaps, meaning that these instances are indeed hard to solve; except if one employs, for instance, the OI reformulations.

	Original formulation			
Instance	Best	Time (s)	Gap (%)	St.
bqp_70_2xR	580	7212.03	14.62	lim
bqp_90_2x30	9420	7212.37	35.51	lim
bqp_90_2xR	1872	7212.13	32.07	lim
bqp_90_3x30	6585	7212.88	34.19	lim
bqp_100_2xR	8606	7212.29	43.86	lim

Table 9: Extended results obtained with CPLEX 12.6 for hard BQP instances.

6 Conclusions

In this paper we discussed the notion of Orbital Independence by presenting theoretical conditions that allow us to break symmetries from different orbits of mathematical programs concurrently: we introduced the concept of independent sets of orbits. An algorithm that potentially identifies the largest independent set of orbits of a mathematical program and generates SBCs to all orbits of such set was also presented. We evaluated the impact of our algorithm by conducting experiments with symmetric instances taken from the libraries MIPLIB2010 and MINLPLib2. We observed that the computational results were coherent in theoretical terms but average-to-poor in practical terms. We conjecture why the results are not expressive, but we take them mainly as an evidence of reaching the limit of what we can do in terms of SSB: no significant impact (for the general case) despite exploiting as many orbits as possible. It seems like determining automatically (prior to exploring the BB tree) a set of SBCs capable of producing a strong computational impact timewise is as hard as solving the original problem itself. Yet we consider the exploitation of OI ideas dynamically (e.g. by means of branching rules) as a potential improvement direction since DSB strategies seem to be most efficient ones. Finally, we have introduced a family of highly symmetric Binary Quadratic Programs which proved to be relevant to the OI theory since they purposely embed the conditions under which the usage of Symmetry-Breaking Constraints is majoritarily advantageous.

Acknowledgments

The first author (GD) was supported by a CNPq Ph.D. thesis award. The second author (LL) gratefully acknowledges funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement n. 764759. We would like to thank two anonymous referees for all the comments and suggestions that helped us to improve our work.

References

1. Achterberg, T.: SCIP: Solving constraint integer programs. Mathematical Programming Computation **1**(1), 1–41 (2009)
2. Berstel, J., Boasson, L.: Context-free languages. In: J. van Leeuwen (ed.) Handbook of Theoretical Computer Science, vol. B, pp. 59–102. MIT Press, Cambridge, MA, USA (1990)
3. Bomze, I., Budinich, M., Pardalos, P., Pelillo, M.: The maximum clique problem. In: D.Z. Du, P. Pardalos (eds.) Handbook of Combinatorial Optimization: Supp. Vol. A, pp. 1–74. Springer US, Boston, MA (1998)
4. Costa, A., Hansen, P., Liberti, L.: Formulation symmetries in circle packing. In: R. Mahjoub (ed.) ISCO 2010 Proceedings, *ENDM*, vol. 36, pp. 1303–1310. Elsevier, Amsterdam, Netherlands (2010)
5. Dias, G., Liberti, L.: Orbital independence in symmetric mathematical programs. In: Z. Lu et al (ed.) COCOA 2015 Proceedings, *LNCS*, vol. 9486, pp. 467–480. Springer (2015)
6. Faenza, Y., Kaibel, V.: Extended formulations for packing and partitioning orbitopes. Mathematics of Operations Research **34**(3), 686–697 (2009)
7. Fischetti, M., Liberti, L.: Orbital shrinking. In: R. Mahjoub, V. Markakis, I. Milis, V. Paschos (eds.) ISCO 2012 Proceedings, *LNCS*, vol. 7422, pp. 48–58. Springer, Berlin, Heidelberg (2012)
8. Fischetti, M., Liberti, L., Salvagnin, D., Walsh, T.: Orbital shrinking: theory and applications. Discrete Applied Mathematics **222**, 109–123 (2017)
9. Fourer, R., Gay, D., Kernighan, B.: The AMPL Book, second edition edn. Cengage Learning, California, USA (2002)
10. Friedman, E.: Fundamental domains for integer programs with symmetries. In: A. Dress, Y. Xu, B. Zhu (eds.) COCOA 2007 Proceedings, *LNCS*, vol. 4616, pp. 146–153. Springer (2007)
11. Galli, S.: Parsing AMPL internal format for linear and non-linear expressions (2004). B.Sc. dissertation, DEI, Politecnico di Milano, Italy
12. Gallo, G., Hammer, P., Simeone, B.: Quadratic knapsack problems, pp. 132–149. Springer Berlin Heidelberg, Berlin, Heidelberg (1980)
13. IBM: ILOG CPLEX 12.6 - User's manual (2014)
14. Kaibel, V., Pfetsch, M.: Packing and partitioning orbitopes. Mathematical Programming **114**(1), 1–36 (2008)
15. Liberti, L.: Reformulations in mathematical programming: Definitions and systematics. RAIRO-RO **43**(1), 55–86 (2009)
16. Liberti, L.: Reformulations in mathematical programming: Automatic symmetry detection and exploitation. Mathematical Programming A **131**, 273–304 (2012)
17. Liberti, L.: Symmetry in mathematical programming. In: S. Leyffer, J. Lee (eds.) Mixed Integer Nonlinear Programming, *IMA Series*, vol. 154, pp. 263–286. Springer, New York (2012)

18. Liberti, L., Cafieri, S., Savourey, D.: Reformulation optimization software engine. In: K. Fukuda et al (ed.) Mathematical Software, *LNCS*, vol. 6327, pp. 303–314. Springer (2010)
19. Liberti, L., Ostrowski, J.: Stabilizer-based symmetry breaking constraints for mathematical programs. Journal of Global Optimization **60**, 183–194 (2014)
20. Margot, F.: Pruning by isomorphism in branch-and-cut. Mathematical Programming **94**, 71–90 (2002)
21. Margot, F.: Exploiting orbits in symmetric ILP. Mathematical Programming B **98**, 3–21 (2003)
22. Margot, F.: Symmetry in integer linear programming. In: M. Jünger et al (ed.) 50 Years of Integer Programming, pp. 647–681. Springer, Berlin (2010)
23. McKay, B.: Practical graph isomorphism. Congressus Numerantium **30**, 45–87 (1981)
24. McKay, B., Piperno, A.: Practical graph isomorphism, II. Journal of Symbolic Computation **60**, 94–112 (2014)
25. Ostrowski, J., Linderoth, J., Rossi, F., Smriglio, S.: Orbital branching. Mathematical Programming **126**(1), 147–178 (2011)
26. Pfetsch, M., Rehn, T.: A computational comparison of symmetry handling methods for mixed integer programs. Tech. Rep. 5209, Optimization Online (2015)
27. The GAP Group: GAP - Groups, Algorithms and Programming. Version 4.7.4 (2014)