Formulation symmetries in circle packing

Alberto Costa, Leo Liberti[1,2]

LIX, École Polytechnique, 91128 Palaiseau, France

Pierre Hansen[3]

GERAD, HEC, 3000 chemin de la Côte-S.te-Catherine, H3T 2A7 Montreal, Canada

Abstract

The performance of Branch-and-Bound algorithms is severely impaired by the presence of symmetric optima in a given problem. We describe a method for the automatic detection of formulation symmetries in MINLP instances. A software implementation of this method is used to conjecture the group structure of the problem symmetries of packing equal circles in a square. We provide a proof of the conjecture and compare the performance of spatial Branch-and-Bound on the original problem with the performance on a reformulation that cuts away symmetric optima.

Keywords: MINLP, spatial Branch-and-Bound, Global Optimization, group, reformulation.

[1] Financial support by grants: ANR 07-JCJC-0151 "ARS", Digiteo 2009-14D "RMNCCO", Digiteo 2009-55D "ARM" is gratefully acknowledged.

[2] E-mail:`{costa,liberti}@lix.polytechnique.fr` (corresponding author: `liberti`).

[3] E-mail:`pierre.hansen@gerad.ca`. Secondary affiliation at *LIX, École Polytechnique.*

1 Introduction

It is well known that problems involving a high degree of symmetry are particularly difficult to solve with Branch-and-Bound (BB) techniques. Intuitively, since optimal solutions are to be found at leaf nodes of the BB tree, the presence of more optima causes longer branches, and hence a higher number of nodes to explore. One possibility for breaking symmetries, proposed in [6,8], is to reformulate the problem by adjoining *symmetry-breaking constraints* (SBC) to the original formulation, yielding a reformulation of the *narrowing* type [7]. The main theoretical contribution of this paper is the determination of the group structure of the problem of packing N equal circles in a square of side $2L$ (see [10] and `www.packomania.com`). We also perform a computational study to assess the efficacy of the proposed automatic symmetry-breaking technique.

1.1 Basic definitions and notation

For $n \in \mathbb{N}$ we let S_n be the symmetric group of order n (i.e. the group of all permutations of n symbols) and C_n be the cyclic group of order n (i.e. the group of rotations of a regular n-polygon). For a subset $N \subseteq \{1, \ldots, n\}$ we let $\mathrm{Sym}(N)$ be the symmetric group on the symbols of N. When G acts on a set X by means of the application gx of g to x for $g \in G, x \in X$, we let $Gx = \{gx \mid g \in G\}$ be the *orbit* of x in G and for a subset $Y \subseteq X$ we let $\mathrm{stab}(Y, G)$, the *setwise stabilizer* of Y in G, be the largest subgroup $H \leq G$ such that $HY = Y$ (i.e. $hy \in Y$ for all $h \in H, y \in Y$). For a permutation $\pi \in S_n$ let $\Gamma(\pi)$ be the set of disjoint cycles σ such that $\pi = \prod_{\sigma \in \Gamma(\pi)} \sigma$. We define $\pi[N] = \prod_{\sigma \in \Gamma(\pi) \cap \mathrm{stab}(N, S_n)} \sigma$. For a directed graph $D = (V, A)$ and $v \in V$ we denote by $\delta^+(v) = \{u \in V \mid (v, u) \in A\}$ the *forward star* of v in D and by $\delta^-(v) = \{u \in V \mid (u, v) \in A\}$ its *backward star*. An *automorphism* of D is a permutation $\pi \in \mathrm{Sym}(V)$ such that $\forall (u, v) \in A \ (\pi(u), \pi(v)) \in A$. For any subset $U \subseteq V$, the graph D' (called *minor*) obtained by *contracting* U has $V \smallsetminus U \cup \{v_U\}$ as vertex set, and an arc (u, v) is in D' either if $u, v \notin U$ and $(u, v) \in A$ or if $u \notin U$, $v = v_U$ and $\exists v' \in U \ (u, v') \in A$ (or vice versa). If D has no cycles then it is a *Directed Acyclic Graph* (DAG); if there exists a unique vertex $v \in V$ with $|\delta^-(v)| = 0$ then we denote v by $r(D)$ and call it the *root* of D (the absence of cycles implies that DAGs have at least one root). A DAG $D = (V, A)$ such that $\forall v \in V \smallsetminus \{r(D)\} \ |\delta^-(v)| = 1$ is a *tree*; for all $v \in V \smallsetminus \{r(D)\}$ we define v^-, the *parent* of v, to be the unique vertex in $\delta^-(v)$; if $v = r(D)$ we assume $v^- = r(D)$. In a tree $D = (V, A)$, for all $v \in V$ the *level* $\ell(v)$ of v is the smallest number of times the parent operator need be applied to v in order to yield $v^{-\cdots-} = r(D)$; given $v \in V$ we denote by

$D[v] = (V[v], A[v])$ the subgraph of D consisting of vertices $u \in V$ for which there is a directed path in D from v to u.

2 Automatic symmetry detection

In this section we discuss a method for computing Mathematical Program (MP) symmetries automatically; conceptually, it is the same as in [8] and similar to [14] but the formal presentation is different. We consider a Mixed-Integer Nonlinear Program (MINLP) P:

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

where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^m$, $x \in \mathbb{R}^n$, and $\mathcal{X} \subseteq \mathbb{R}^n$ is a set which might include variable ranges $x^L \le x \le x^U$ as well as integrality constraints on a subset of variables $\{x_i \mid i \in Z\}$ for some $Z \subseteq \{1, \dots, n\}$. Let $\mathcal{G}(P)$ be the set of global optima of P and $\mathcal{F}(P)$ be its feasible region. We define the action of S_n on $\mathbb{R}^n$ as follows: $\forall \pi \in S_n, x \in \mathbb{R}^n$ let $\pi(x_1, \dots, x_n) = (x_{\pi^{-1}(1)}, \dots, x_{\pi^{-1}(n)})$ so that, for example, $(1,2,3)(x_1, x_2, x_3) = (x_3, x_1, x_2)$. The group $G^*_P = \mathrm{stab}(\mathcal{G}(P), S_n)$ is called the *solution group* of P. The solution group is the largest subgroup of S_n which maps every global optimum into another global optimum. Since G^*_P depends on $\mathcal{G}(P)$ it cannot, in general, be found before the solution process. We therefore try to find subgroups of G^*_P. In particular, we consider the subgroup of G^*_P consisting of all variable permutations which "fix the formulation" of P. For $\pi \in S_n$ and $\sigma \in S_m$ we define $\sigma P \pi$ to be the following MINLP:

$$\min\{f(\pi x) \mid \sigma g(\pi x) \le 0 \wedge \pi x \in \mathcal{X}\}, \tag{2}$$

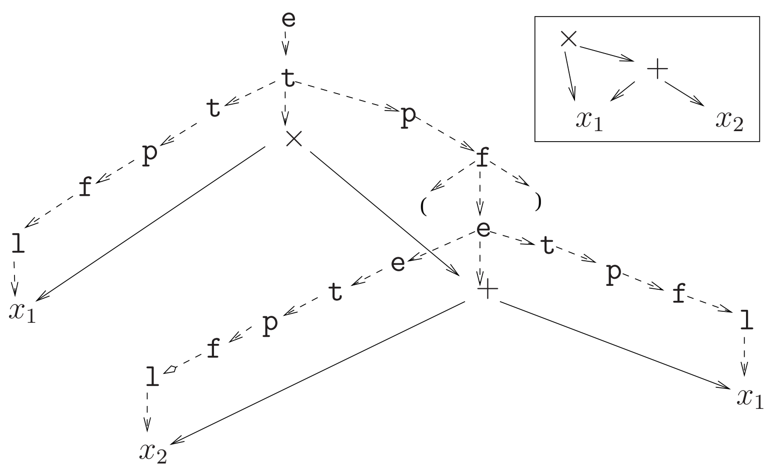
where σ acts on $g = (g_1, \dots, g_m)$ by $\sigma g = (g_{\sigma^{-1}(1)}, \dots, g_{\sigma^{-1}(m)})$. Consider the group $\bar{G}_P = \{\pi \in S_n \mid \exists \sigma \in S_m \ (\sigma P \pi) = P\}$. Whenever P is a Mixed-Integer Linear Program (MILP), $\bar{G}_P$ is called the *LP relaxation group* [11]. For general MINLPs, determining whether $\forall x \in \mathrm{dom}(f)\ f(\pi x) = f(x)$ and $\forall x \in \mathrm{dom}(g)\ \sigma g(\pi x) = g(x)$ is an undecidable problem.

We therefore introduce the following restriction: f, g_i $(i \le m)$ must be strings of the formal language $\mathscr{L}$ on the alphabet $\mathscr{A}$ given by the operators in $\{+, -, \times, \div, \uparrow, \log, \exp, (,)\}$ (where $a \uparrow b = a^b$), the variable symbols in $\{x_1, \dots, x_n\}$ and the constant symbols in $\mathbb{R}$, where $\mathscr{L}$ is recognized by the following formal grammar [13]:

$$\left.\begin{array}{lllllll}
\mathtt{e} : \mathtt{t} & | & \mathtt{e} + \mathtt{t} & | & \mathtt{t} - \mathtt{t} & & \\
\mathtt{t} : \mathtt{p} & | & \mathtt{t} \times \mathtt{p} & | & \mathtt{p} \div \mathtt{p} & & \\
\mathtt{p} : \mathtt{f} & | & \mathtt{f} \uparrow \mathtt{f} & & & & \\
\mathtt{f} : \mathtt{l} & | & -(\mathtt{e}) \ | \ \log(\mathtt{e}) & | & \exp(\mathtt{e}) & | & (\mathtt{e}) \\
\mathtt{l} : x_i & & (1 \le i \le n) & | & c \quad (c \in \mathbb{R}). & &
\end{array}\right\} \tag{3}$$

The lexical recognition performed by the grammar to decide whether a string a of $\mathscr{A}^*$ belongs to $\mathscr{L}$ naturally yields the (well-known) *derivation tree* $\mathcal{D}_a = (\mathcal{V}_a, \mathcal{A}_a)$ of a [13]. The *expression tree* $T'_a = (V'_a, A'_a)$ associated to a is such that V'_a consists of all the symbols of $\mathscr{A}$ in a, and, for $u \in V'_a$ there is an arc (u, v) whenever $v = \mathrm{argmin}\{\ell_{\mathcal{D}_a}(z) - \ell_{\mathcal{D}_a}(u) \mid \ell_{\mathcal{D}_a}(z) > \ell_{\mathcal{D}_a}(u) \wedge z \in \mathcal{D}_a[u^-]\}$. For all $v \in V_a$ we let $\lambda(v)$ (v's *label*) be the alphabet symbol that v represents. Let $\mathscr{V} = \{x_1, \ldots, x_n\} \subseteq \mathscr{A}$ be the set of variable symbols of $\mathscr{A}$. For two nodes $u, v \in V'_a$ we define the equivalence relation $u \sim v$ given by $\lambda(u), \lambda(v) \in \mathscr{V} \wedge \lambda(u) = \lambda(v)$, and consider the DAG $T_a = (V, A)$ defined as $T'_a / \sim$ (i.e. T_a is the graph minor obtained by contracting each set in the partition $V'_a / \sim$).

For example, the string $a = x_1 \times (x_2 + x_1)$ is recognized by the grammar (3) to be an element of $\mathscr{L}$. It yields the derivation tree $\mathcal{D}_a$ (dashed arrows) and the expression tree T'_a (full arrows) below. The DAG T_a obtained by contracting the two vertices labelled by x_1 is shown in the upper right box.



For $a, b \in \mathscr{L}$ we define $a \cong b$ if and only if $T_a = T_b$: this can be established in linear time in $|a|, |b|$ by simply recursing on the respective DAGs. It is easy to show that if $a \cong b$ then $\mathrm{dom}(a) = \mathrm{dom}(b) \wedge \forall x \in \mathrm{dom}(a)\ a(x) = b(x)$ (thus the functions represented by the strings a, b are equal), but the converse may not hold. For a MP P' defined as $\min\{f'(x) \mid g'(x) \leq 0 \wedge x \in \mathcal{X}'\}$, we write $P \cong P'$ if: (a) P, P' have the same number of variables and constraints; (b) $\mathcal{X} = \mathcal{X}'$; (c) $f \cong f'$ and $\forall i \leq m\ (g_i \cong g'_i)$. We are finally in a position to define the *formulation group* $G_P = \{\pi \in S_n \mid \exists \sigma \in S_m\ (\sigma P \pi \cong P)\}$ of P. It is easy to show that $G_P \leq \bar{G}_P \leq G^*_P$ [6]. For MILPs, $G_P = \bar{G}_P$ [8].

Computing G_P involves testing whether the two formulations P and $Q = \sigma P \pi$ (for some $\sigma \in S_m, \pi \in S_n$) have the property $P \cong Q$. To this purpose we construct a DAG representation of MPs. To ease notation, we let $g_0 =$

f. Let $\mathbf{D}(P) = (\mathbf{V}, \mathbf{A})$ be the DAG given by $(\bigcup_{0 \le i \le m} T_{g_i})/\sim$. We define a *color label* $\gamma : \mathbf{V} \to \mathbb{N}$ such that: (i) nodes $u, v \in \mathbf{V} \cap \mathscr{V}$ with same ranges and integrality constraints have the same color (permute like variables); (ii) the root of T_{g_0} has a different color from the roots of the T_{g_i}'s (do not permute objective with constraints); (iii) for all $1 \le i \le m$ the roots of T_{g_i} all have the same color (permute constraints order); (iv) for distinct $i < j \in \{0, \ldots, m\}$, two nodes $u \in V_{g_i}, v \in V_{g_j}$ not in $\mathscr{V}$ have different colors (do not permute operators from different mathematical expressions); (v) for all $i \in \{0, \ldots, m\}$, two nodes in $u, v \in V_{g_i}$ not in $\mathscr{V}$ with same operator label, same tree levels (these are well defined for the DAGs T_{g_i} because $\sim$ only applies to nodes in $\mathscr{V}$), and in the case of noncommutative operators, same position in the ordered lists $\delta^+(u^-), \delta^+(v^-)$, have the same color (within the same mathematical expression, permute operators which "play the same role"). We then define an equivalence relation $\sim_\gamma$ on the nodes of $\mathbf{V}$ such that $u \sim_\gamma v$ if $\gamma(u) = \gamma(v)$, and consider the partition $\mathbf{V}_1 \ldots, \mathbf{V}_q$ of $\mathbf{V}$ induced by $\sim_\gamma$. We let $\mathcal{H}_P$ be the largest group of automorphisms of $\mathbf{D}$ fixing each $\mathbf{V}_h$, i.e. such that $\forall h \le q$ $(\mathcal{H}_P \mathbf{V}_h = \mathbf{V}_h)$. Consider the mapping $\phi : \mathcal{H}_P \to \mathrm{Sym}(\mathscr{V})$ given by $\phi(\pi) = \pi[\mathscr{V}]$. Then ϕ is a group homomorphism, and $\mathrm{Im}(\phi) = G_p$ (Theorems 1,2 in [8]). This reduces the Graph Isomorphism (GI) problem [2] to the problem of computing G_P: we use the GI-based software `nauty` [12] to compute a set of generators for $\mathcal{H}_P$ and then use ϕ to obtain a set of generators $\mathscr{G}$ for G_P. It is not known whether GI is in **P** or **NP**-complete; `nauty` has exponential complexity. Graph classes for which GI is as difficult as for general graphs are termed GI-complete: it is known that GI is polynomial on trees but DAGs are GI-complete. Our DAGs are "almost trees" and we can therefore expect them to be "practically easy" — this is in fact the case, as the computational experiments in [8] show.

Once G_P is known, we aim to find a reformulation Q of P which ensures that at least one symmetric optimum of P is in $\mathcal{G}(Q)$. Such reformulations are known as *narrowings* [7]. A set of constraints $h(x) \le 0$ are SBCs with respect to $\pi \in G_P$ if there is $y \in \mathcal{G}(P)$ such that $h(\pi y) \le 0$. Adjoining SBCs to P yields a narrowing Q of P [8]. Let Ω be the set of nontrivial orbits of the action of G_P on $\mathscr{V}$, and let $\omega \in \Omega$. Then $\forall j \in \omega$ $x_{\min \omega} \le x_j$ are SBCs with respect to G_P [8].

3 Circle packing in a square

Circle Packing in a Square (CPS). Given $N \in \mathbb{N}$ and $L \in \mathbb{Q}_+$, can N non-overlapping circles of unit radius be arranged in a square of side $2L$?

We formulate the CPS as the following nonconvex NLP:

$$\max\{\alpha \mid \forall i < j \leq N \; \|x_i - x_j\|^2 \geq 4\alpha \wedge x \in [1-L, L-1]^{2N}\} \tag{4}$$

For any given $N, L > 1$, if a global optimum (x^*, α^*) of (4) has $\alpha^* \geq 1$ then the CPS instance is a YES one. Although the CPS is not usually formulated as (4) [10], our formulation is equivalent to the more usual one of maximizing the radius of circles in a unit square. Solving the nonconvex NLP (4) to global optimality using standard sBB solvers yields search trees of disproportionate sizes even for very small instances. One of the reasons for the slow convergence of sBB is that (4) has many symmetries.

Using the method described in Sect. 2, we conjecture that $G_{\text{SPS}} = C_2 \times S_N$. This is intuitively reasonable, i.e. we expect to be able to permute the two dimension indices and the circle indices. We show in Thm. 3.1 that this is indeed the case. Let Q be the CPS formulation (4).

Theorem 3.1 *The formulation group of the CPS is isomorphic to $C_2 \times S_N$.*

Proof. For all $i < j \leq N$ call the constraints $\|x_i - x_j\|^2 \geq 4\alpha$ the *distance* constraints. Let $(x, \alpha) \in \mathcal{G}(Q)$; the following claims are easy to establish.

(i) The permutation $\tau = \prod_{i \leq N}(x_{i1}, x_{i2})$ is in G_Q (invariance of distance constraints); notice that $\langle \tau \rangle \cong S_2 = C_2$.

(ii) For any $i \leq N-1$, the permutation $\sigma_i = (x_{i1}, x_{i+1,1})(x_{i2}, x_{i+1,2})$ is in G_Q (invariance of distance constraints); notice that $\langle \sigma_i \mid i \leq N-1 \rangle \cong S_N$.

(iii) Any permutation moving α to one of the x variables is not in G_Q. This follows because the α has different range constraints than those of the x variables, hence the corresponding node colors given by γ in $\mathbf{D}$ are different.

(iv) If $\pi \in G_Q$ such that $\pi(x_{i1}) = x_{i2}$ for some $i \leq N$ then $\pi(x_{i1}) = x_{i2}$ for all $i \leq N$, as otherwise the term $x_{i1}x_{j1} + x_{i2}x_{j2}$ (appearing in the distance constraints) would be mapped to a term not appearing in the problem.

(v) For any $i \leq N-1$, if $\pi \in G_Q$ such that $\pi(x_{ik}) = x_{i+1,k}$ for some $k \leq 2$, then $\pi(x_{ik}) = x_{i+1,k}$ for all $k \leq 2$, as otherwise the term $x_{i1}x_{i+1,1} + x_{i2}x_{i+1,2}$ (appearing in some of the distance constraints) would be mapped to a term not appearing in the problem.

Let $K = \langle \tau \rangle$ and $H_N = \langle \sigma_i \mid i \leq N-1 \rangle$. Claims (i)-(ii) imply that $K, H_N \leq G_Q$. It is easy to check that $KH_N = H_N K$; it follows that $KH_N \leq G_Q$ [4] and hence K, H_N are normal subgroups of KH_N. Since $K \cap H_N = \{e\}$, we have $KH_N \cong K \times H_N \cong C_2 \times S_N \leq G_Q$ [1]. Now suppose $\pi \in G_Q$ with $\pi \neq e$. By Claim (iii), π cannot move α so it must map x_{i1} to x_{j2} for some $i < j \leq N$; the action $i \to j$ on the circles indices can be decomposed into a product of transpositions $i \to i+1, \ldots, j-1 \to j$. Thus, by Claim (v) (resp. iv), π involves a certain product γ of τ and σ_i's; furthermore, since by definition γ maps x_{i1} to x_{j2}, any permutation in G_Q (including π) can be obtained as a product of these elements γ; hence π is an element of KH_N, which shows $G_Q \leq KH_N$, implying $G_Q \cong C_2 \times S_N$. □

By Thm. 3.1, $G_Q = \langle \tau, \sigma_i \mid i \le N-1 \rangle$. It is easy to show that there is just one orbit in the natural action of G_Q on the set $A = \{1,\ldots,N\} \times \{1,2\}$, and that the action of G_Q on A is not symmetric (otherwise G_Q would be isomorphic to S_{N2}, contradicting Thm. 3.1).

Proposition 3.2 *For any fixed $h \le 2$,*

(5) $\forall i \le N \smallsetminus \{1\} \quad x_{i-1,h} \le x_{ih}$

are SBCs with respect to any $\pi \in G_Q$.

Proof. Let $\bar{x} \in \mathcal{G}(Q)$; since the σ_i generate the symmetric group acting on the N circles, there exists a permutation $\pi \in G_Q$ such that $(\bar{x}_{\pi(i),h} \mid i \le N)$ are ordered as in (5). □

4 Computational results

A partial computational study on the SPS is currently in revision [9]. Here we focus on the CPS problem. We compare the effect of three distinct symmetry-breaking devices: (i) *fixing* the first circle to $(-L+1, L-1)$ (this can always be done by [10] and the technical report cited therein); (ii) adjoining the *weak* SBCs discussed at the end of Sect. 2; (iii) adjoining the *strong* SBCs given in Prop. 3.2. In particular, we test the configurations *original* (A), *weak* (B), *strong* (C), *fix* (D), *strong+fix* (E). We remark that *weak+fix* is the same as *fix*, as the weak SBCs imply the first circle. Our comparative solutions, yielded by running Couenne [3] on CPS instances in the configurations A-E described above, have been obtained on one 2.4GHz Intel Xeon CPU of a computer with 24 GB RAM running Linux. These results are shown in the table below, which contains the following statistics at termination (occurring after at most 2h of user CPU time): the objective function value f^* of the incumbent, the gap still open, the seconds of user CPU time taken, the number of BB nodes closed, the number of BB nodes still on the tree. The last column, labelled *R.t.*, lists the reformulation time: i.e. the seconds of user CPU time taken to find the formulation group, its longest orbit, and to adjoin the weak SBCs to the formulation.

References

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

[2] Babai, L., *Automorphism groups, isomorphism, reconstruction*, in: R. Graham, M. Grötschel and L. Lovász, editors, *Handbook of Combinatorics, vol. 2*, MIT Press, Cambridge, MA, 1996 pp. 1447–1540.

	A		B		C		D		E		
Inst.	*f** *gap*	*CPU* *nodes* *tree*	*f** *gap*	*CPU* *nodes* *tree*	*f** *gap*	*CPU* *nodes* *tree*	*f** *gap*	*CPU* *nodes* *tree*	*f** *gap*	*CPU* *nodes* *tree*	*R.t.*
6_3	-1.149 0%	0.60 11 0	-1.149 0%	2.09 54 0	-1.149 0%	5.67 3 0	-1.143 0%	**0.22** **2** 0	-1.444 0%	4.02 1998 0	0.96
7_3	-1 0%	185.57 107973 0	-1.13 0%	1.74 8 0	-1.059 0%	10.53 61 0	-1.149 0%	**0.34** **2** 0	-1.149 0%	35.86 22832 0	0.97
8_3	-1.008 0%	7.63 1832 0	-1.035 0%	9.18 573 0	-1.048 0%	10.33 **3** 0	-1.048 0%	**1.07** 4 0	**-1.072** 0%	149.82 69141 0	0.96
9_3	-1 0%	4428.34 1374568 0	-1 0%	2021.78 571444 0	-0.608 0%	197.02 52854 0	-1 0%	**174.37** **51192** 0	-1 0%	595.87 236522 0	0.98
10_3	0 ∞	7200 1673901 669374	0 ∞	7200 1550374 599059	-0.604 0%	**430.70** **117738** **0**	0 ∞	7200 1809594 699698	**-0.71** 37.69%	7200 2136970 386244	0.99
9_4	-1.022 0%	374.18 112811 0	-2 0%	92.15 23306 0	-1.376 0%	13.91 3 0	-2 0%	2.76 65 0	-2 0%	**0.41** **1** 0	0.95
10_4	-1.563 0%	231.39 56084 0	-1.563 0%	150.61 32606 0	-1.268 0%	14.44 3 0	-1.575 0%	**1.26** **2** 0	**-1.591** 25.67%	7200 2086972 322879	0.97
11_4	-1.366 0%	3534.43 693143 0	-1.391 0%	368.51 67216 0	-1.258 0%	16.36 3 0	**-1.427** 0%	**1.49** **2** 0	-1.403 42.51%	7200 1704953 336519	0.97
18_4	0 ∞	7200 **289331** 134573	0 ∞	7200 305440 **139846**	-0.588 240.17%	7200 339794 105473	0 ∞	7200 409827 179132	**-0.743** **169.01%**	7200 385177 168919	0.99
20_5	0 ∞	7200 150161 71419	0 ∞	7200 144844 68535	-1.165 **0%**	**76.85** **5** **0**	0 ∞	7200 363353 171569	**-1.314** 52.17%	7200 261265 118476	0.99

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

[4] Clark, A., "Elements of Abstract Algebra," Dover, New York, 1984.

[5] Kucherenko, S., P. Belotti, L. Liberti and N. Maculan, *New formulations for the kissing number problem*, Discrete Applied Mathematics **155** (2007), pp. 1837–1841.

[6] Liberti, L., *Automatic generation of symmetry-breaking constraints*, in: B. Yang, D.-Z. Du and C. Wang, editors, *COCOA Proceedings*, LNCS **5165** (2008), pp. 328–338.

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

[8] Liberti, L., *Reformulations in mathematical programming: Symmetry*, Mathematical Programming (in revision).

[9] Liberti, L., *Symmetry in mathematical programming*, in: J. Lee and S. Leyffer, editors, *Mixed Integer Nonlinear Programming*, IMA, Minneapolis, in revision .

[10] Locatelli, M. and U. Raber, *Packing equal circles in a square: a deterministic global optimization approach*, Discrete Applied Mathematics **122** (2002), pp. 139–166.

[11] Margot, F., *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*, Springer, Berlin, 2010 pp. 647–681.

[12] McKay, B., *Practical graph isomorphism*, Congressus Numerantium **30** (1981), pp. 45–87.

[13] Mosses, P., *Denotational semantics*, in: J. van Leeuwen, editor, *Handbook of Theoretical Computer Science B: Formal Models and Semantics*, Elsevier, Amsterdam, 1990 pp. 575–631.

[14] Ramani, A. and I. Markov, *Automatically exploiting symmetries in constraint programming*, in: B. Faltings, A. Petcu, F. Fages and F. Rossi, editors, *Constraint Solving and Constraint Logic Programming*, LNAI **3419** (2005), pp. 98–112.