New symmetries in mixed-integer linear optimization

Symmetry heuristics and complement-based symmetries

Philipp M. Christophel[⋆], Menal Güzelsoy, and Imre Pólik

SAS Institute, Inc.
Cary, NC, USA

Abstract. We present two novel applications of symmetries for mixed-integer linear programming. First we propose two variants of a new heuristic to improve the objective value of a feasible solution using symmetries. These heuristics can use either the actual permutations or the orbits of the variables to find better feasible solutions.
Then we introduce a new class of symmetries for binary MILP problems. Besides the usual permutation of variables, these symmetries can also take the complement of the binary variables. This is useful in situations when two opposite decisions are actually symmetric to each other. We discuss the theory of these symmetries and present a computational method to compute them.
Examples are presented to illustrate the usefulness of these techniques.

1 Introduction

1.1 Symmetries in MILP

Let us consider an integer programming problem[1] of the form:

$$\begin{aligned} \min\ & c^T x \\ & Ax \le b \\ & x \in \{0,1\}^n , \end{aligned} \tag{1}$$

where $x, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m\times n}$. In general, following the concepts in [1, 2], a *symmetry* of problem (1) is a permutation of its rows and columns such that the permuted problem is identical to the original one. In particular, let π^{r} and π^{c} be permutations of the rows and columns, respectively, then they form a symmetry of problem (1) if

$$A_{i,j} = A_{\pi^{\mathrm{r}}(i),\pi^{\mathrm{c}}(j)} \tag{2a}$$

$$b_j = b_{\pi^{\mathrm{r}}(j)} \tag{2b}$$

$$c_i = c_{\pi^{\mathrm{c}}(i)} \tag{2c}$$

[⋆] `philipp.christophel@sas.com`, corresponding author

[1] The concepts in this paper naturally extend to the mixed-integer linear case, but for simplicity of notation we will use the pure binary form.

Now, if x is a feasible solution of (1), then $\pi^{\mathrm{c}}(x)$, which is defined naturally as

$$\pi^{\mathrm{c}}(x)_i = x_{\pi^{\mathrm{c}}(i)} \tag{3}$$

is also feasible with the same objective value, thus the two solutions are equivalent. It is important to detect the symmetry of the problem so that equivalent solutions are not appearing on the search tree.

Another important concept related to symmetries is the notion of *orbits*. Two columns (or rows) are said to belong to the same orbit if there is a permutation that maps one to the other. Naturally, each variable belongs to exactly one orbit, thus the orbits form a partition of the columns and rows.

These techniques are well known (see, e.g., [3] for a recent overview) and have been implemented in all major commercial software packages, including SAS/OR 13.1.

Permutations will be presented with the usual cycle notation. Whenever we talk about symmetries of a problem we will usually drop the row permutations. The all-1 vector will be denoted by e.

The paper is structured as follows. In Section 2 we present two improvement heuristics that use symmetry information. In Section 3 we introduce the notion of complement symmetries and in Section 4 we present a computational method to obtain these symmetries.

2 Symmetry heuristics

In this section we are interested in symmetries of problem (1) that satisfy conditions (2a) and (2b), but not (2c). We will call these *constraint symmetries*, since applying them to a solution maintains feasibility, but not the objective value. It is important that the group of constraint symmetries can be larger than the original symmetry group of the problem, and there are examples for this even in the public benchmark set MIPLIB2010 [4] (see instances `bab5`, `msc98`, `neos_1109824`, `ns1766074`, `ran16x16` and `unitcal_7`, for example).

We can use these permutations to improve the objective value of a feasible solution. There are multiple ways to use this information, here we present two different schemes and discuss which one is better in certain situations.

2.1 Direct improvement heuristics

Having a list of generators of the group of constraint symmetries we can just apply them repeatedly to a feasible solution and see if we get a better objective value. An advantage of this heuristic is that using the symmetries guarantees that feasibility is preserved. In general, without the symmetry information this would be complicated to achieve. On the other hand, trying to permute the solution to minimize its objective value is equivalent to optimizing a linear objective function over a group defined by generators, a problem that is known to be NP hard [5].

Also, the set of generators we have is usually very small. On one hand this is an advantage, because the algorithms that find these symmetries are faster, but it can easily happen that none of the symmetries improves the currently best solution. To alleviate this situation we use a solution pool of the best few solutions along with a symmetry pool. Whenever a new solution is found we put it in the pool and every once in a while we delete the solutions whose objective value is not good enough. On the other hand, if none of the symmetries in the symmetry pool can improve any of the solutions in the solution pool, then we can combine symmetries to increase the size of the symmetry pool. We iterate this a few times or as long as we are finding better solutions.

This method is very fast, since the symmetries are usually permutations that move only a few elements, so checking whether they improve a solution is fast. Speed has its price, since this method cannot hope to cover all the solutions that can be generated. Another limitation of this method is that it cannot change the number of nonzeros in a solution. We will present an improvement in Section 4.3.

Example The following example illustrates the idea behind the direct improvement symmetry heuristic. Consider this problem:

$$\begin{aligned} \min \quad & c^T x \\ Ax &\geq 1 \\ x &\in \{0,1\}\,, \end{aligned}$$

where

$$c = (1\ 2\ 3\ 3\ 1\ 2)$$
$$A = \begin{pmatrix} 1\,0\,1\,0\,0\,0 \\ 0\,1\,0\,1\,0\,0 \\ 1\,0\,0\,0\,1\,0 \\ 0\,1\,0\,0\,0\,1 \\ 0\,0\,1\,0\,1\,0 \\ 0\,0\,0\,1\,0\,1 \end{pmatrix}.$$

A feasible solution is

$$x = (1\ 1\ 1\ 1\ 0\ 0) \tag{4}$$

with objective value 9. The constraint symmetries of this problem are generated by

$$\mathcal{G}^{\mathcal{C}}(A) = \{(1,3),(2,4),(4,6),(1,2)(3,4)(5,6)\}\,. \tag{5}$$

The following table shows how symmetries are used to improve the solution:

Nr	x	z	symmetry
1	(1 1 1 1 0 0)	9	(4,6)
2	(1 1 1 0 0 1)	8	(1,2)(3,4)(5,6)
3	(1 1 0 1 1 0)	7	(4,6)
4	(1 1 0 0 1 1)	6	optimal

There are a few key points to note here. First, because of the different objective coefficients, this problem has no symmetries, but it still has a nontrivial constraint symmetry group. This is actually typical for a lot of optimization problems, where the constraint set is symmetric and the objective coefficients are breaking the symmetry.

Further, it is important to note that not all permutations of 4 ones give a feasible solution, for example, (0 1 0 1 1 1) is not feasible. This is because the permutation $(1, 4)$ is not a constraint symmetry. The strength of our heuristic is that applying the symmetries we do not have to worry about feasibility. In fact, each solution we get this way will be exactly the same feasible as the original solution. Finally, this procedure is not guaranteed to yield an optimal solution (as it does in our example), it can get stuck in a locally optimal solution. This motivates our second approach.

2.2 Orbit MIPping

Another way to use symmetry information to improve the objective value of a feasible solution is to use only the orbit information. Variables that can be mapped to each other with symmetries are said to belong to the same orbit. The orbits give a partition of the variables of problem (1) and it is easy to see that applying a symmetry to a solution cannot change the number of nonzeros on a given orbit. Our orbit mipping heuristic uses this simple fact to improve the solutions. Given a feasible solution $\tilde{x}$ and an orbit partition $\mathcal{O}_1, \ldots, \mathcal{O}_k$ we can look at the following subproblem:

$$\begin{aligned} \min\ & c^T x \\ Ax &\le b \\ \sum_{i \in \mathcal{O}_j} x_i &= \sum_{i \in \mathcal{O}_j} \tilde{x}_i,\ \forall j = 1, \ldots, k \\ x &\in \{0,1\}^n . \end{aligned} \tag{6}$$

In other words, for each orbit we fix the sum of variables to be the same as in $\tilde{x}$ and solve the resulting MILP. This will actually search a broader set than just the solutions that are symmetric to $\tilde{x}$. We can feed in $\tilde{x}$ as a feasible solution to the optimization when solving problem (34).

The rationale behind the efficiency of this method is that the extra equality constraints might imply a lot of fixings throughout the problem. This approach is useful if the constraint symmetry orbits of the solutions are large and there are only a few nonzeros on each orbit, or if the group of constraint symmetries is too large for the direct improvement heuristic to be efficient. A disadvantage of this method is that it is much slower than using the symmetries directly, but it can find better solutions. Orbit MIPping has a very important property that distinguish it from other improvement heuristics. If the set of constraint symmetries is larger than the symmetry group of the problem (something which is easy to check), then we know that there is another solution symmetric to the current solution, but with a different (better or worse) objective value.

3 Complement-based symmetries

In the second part of the paper we are presenting a new class of symmetries for MILP, we show how to compute them and bring examples from public test libraries.

3.1 Motivation

Consider the following MILP:

$$\begin{aligned} \min\ & x_1 - x_2 \\ & x_2 \leq x_1 \\ & x_1, x_2 \text{ binary} \end{aligned} \tag{7}$$

The feasible set is depicted in Figure 1. Notice that swapping the two variables

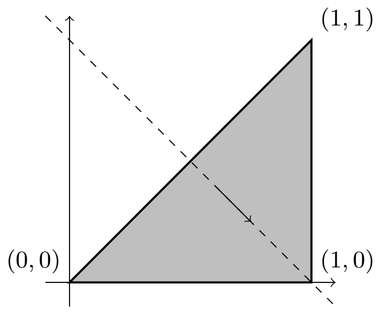

Fig. 1. The feasible set (shaded area) of the linear relaxation of problem (7). The dashed line denotes the axis of symmetry. The arrow indicates the direction of the improving objective value.

(which is the only nontrivial permutation-based symmetry for this problem) does not map the feasible set into itself, so this problem does not have a symmetry in the classical sense. However, the feasible set has an obvious axis of symmetry, denoted by the dashed line. Solutions $(0, 0)$ and $(1, 1)$ are equivalent, with the same objective value. Our goal is to algorithmically identify symmetries of this kind. Intuitively, these are symmetries that map the 0 value of a variable to the 1 value of another variable.

3.2 Definition and basic properties

Consider again problem (1). We will study symmetries of the following form:

$$x \mapsto Qx + q, \tag{8}$$

where Q is a matrix with exactly one ± 1 coefficient in each row and column, i.e., a signed permutation matrix. The pair (Q, q) is said to be a *signed symmetry* or *complement symmetry* of problem (1) if this mapping does not change the objective value and the feasibility of a solution vector x. More precisely, this happens if for all x we have

$$c^T x = c^T (Qx + q), \tag{9}$$

and we have a regular permutation matrix P such that again for all x we have

$$b - Ax = P\left(b - A(Qx + q)\right). \tag{10}$$

This means that the solution x and its image $Qx + q$ are equivalent for (1).

Since conditions (9) and (10) have to hold for all x, we can simplify them as follows:

$$\begin{aligned} Q^T c &= c \\ q^T c &= 0 \\ PAQ &= A \\ P(b - Aq) &= b. \end{aligned} \tag{11}$$

In addition to these requirements, our complement symmetry (Q, q) has to satisfy one additional condition: it has to map binary variables to binary variables.[2] Formally:

$$Qx + q \in \{0, 1\}^n \text{ if and only if } x \in \{0, 1\}^n. \tag{12}$$

It turns out that this can be achieved by selecting q appropriately:

Lemma 1. *If $q = (e - Qe)/2$, then $Qx + q$ is binary if and only if x is binary.*

Proof. Let x be an arbitrary binary vector and let i be an index, and let Q_i be the ith row of Q. Now Q_i has exactly one nonzero, let it be at the jth position, thus $Q_{ij} = \pm 1$. The ith component of $Qx + q$ is then

$$Q_i^T x + \frac{1 - Q_i^T e}{2} = Q_{ij} x_i + \frac{1 - Q_{ij}}{2} = \begin{cases} 1 - x_j, & \text{if } Q_{ij} = -1 \\ x_j, & \text{if } Q_{ij} = 1. \end{cases} \tag{13}$$

This shows that $(Qx + q)_i$ depends only on x_j, and that it is binary.

Corollary 1. *Since*

$$q^T c = \frac{(e - Qe)^T c}{2} = \frac{e^T (c - Q^T c)}{2},$$

$Q^T c = c$ implies $q^T c = 0$, so with this choice of q we do not have to check the second condition in (11)*, since it will follow from the first one.*

[2] We have to deal with this additive term q only because the range of our variables is not symmetric about the origin. If we used ± 1 variables instead of binaries, then we would not need it. Thus it is not surprising that q is completely defined by Q.

3.3 Signed permutation matrices

Matrices with the structure of matrix Q are called signed permutation matrices in the literature (see [6]). They are especially important for us, since they form a group, called the hyperoctahedral group, which happens to be the symmetry group of the hypercube (and by duality, the symmetry group of the hyperoctahedron, whence the name). Any symmetry of the hypercube, thus any symmetry that maps binary solutions to other binary solutions can be described by a signed permutation matrix. In this sense, signed permutations are exactly the right set of symmetries to consider for binary problems.

In comparison, regular symmetries form a much smaller group as they always leave the origin fixed.

3.4 Interpretation

The symmetries defined by (Q, q) can map a binary variable into the complement of another (or the same) binary variable. This greatly enhances the group of symmetries that we can use for reducing the search space. In particular, if Q_{ij} is 1, then x_j is mapped to x_i as usual, but when Q_{ij} is -1, then x_j is mapped to the complement of x_i. Necessarily, x_i is mapped to the complement of x_j.

Consider for example the following problem:

$$\begin{aligned} \min\ & x_1 - x_2 + x_3 \\ & x_1 - x_2 + x_3 \leq 1 \\ & x_1, x_2, x_3 \text{ binary}, \end{aligned} \tag{14}$$

whose feasible set is depicted in Figure 2. Geometrically, the symmetries are

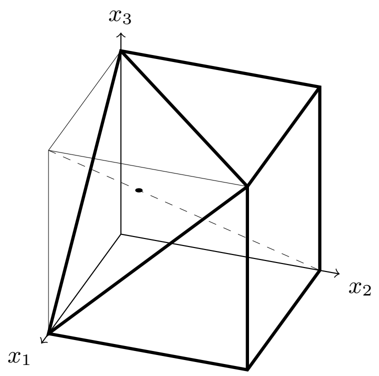

Fig. 2. A polyhedron inside the unit cube with the symmetry $(1\bar{2}3)$.

120° rotations about the dashed line. Extending the cycle notation for signed permutations we can write $(1\bar{2}3)$ to denote that x_1 is mapped to the complement of x_2, which in turn is mapped to x_3, which is mapped to x_1. This can be written equivalently as $(\bar{1}2\bar{3})$ using the complements of the variables.

4 Finding complement-based symmetries

Now that we have introduced complement symmetries we are going to present a way to obtain them from the problem description. The crux of the algorithm is a combination of a lifting procedure with the procedure of finding regular symmetries. Consider the following lifted problem (compare with (1)):

$$\begin{aligned} &\min\ (c^T x - c^T \bar{x} + c^T e)/2 \\ &Ax - A\bar{x} \leq 2b - Ae \\ &x + \bar{x} = 1 \\ &x, \bar{x} \in \{0,1\}^n \,. \end{aligned} \tag{15}$$

Here $\bar{x}$ is merely a symbol that denotes a variable. The constraints actually force it to be the complement of x, whence the notation, but it is important to understand that it is not the notation that makes it the complement but the constraints. The optimization problem (15) is closely related to (1):

Lemma 2. *If x is a feasible solution of* (1) *then $(x, \bar{x})$ is a feasible solution for* (15) *with the same objective value, and vica versa.*

Proof. For the forward implication let x be a feasible solution of (1) and let $\bar{x}$ be its complement. Then we can write:

$$\left(c^T x - c^T \bar{x} + c^T e\right)/2 = (c^T x + c^T x - c^T e + c^T e)/2 = c^T x \tag{16}$$

and

$$Ax - A\bar{x} = Ax - Ae + Ax \leq b - Ae + b = 2b - Ae \tag{17}$$

proving that $(x, \bar{x})$ is a feasible solution for (15) with the same objective value.

For the backward direction let $(x, \bar{x})$ be a feasible solution of (15). From the second constraint we get that $\bar{x}$ is in fact the complement of x, i.e., $\bar{x} = e - x$. Now

$$c^T x = c^T(x/2 - \bar{x}/2) + c^T(x/2 + \bar{x}/2) = (c^T x - c^T \bar{x} + c^T e)/2 \tag{18}$$

and

$$Ax = A(x/2 - \bar{x}/2) + A(x/2 + \bar{x}/2) \leq b - Ae/2 + Ae/2 = b. \tag{19}$$

The main result of this section is the following theorem:

Theorem 1. *There is a one-to-one correspondence between the regular, permutation symmetries of problem* (15) *and the complement-based symmetries of problem* (1).

Proof. The proof is constructive. First, let (Q, q) together with P be a signed symmetry of (1). Let $Q_+ \geq 0$ and $Q_- \leq 0$ be the positive and negative parts of Q such that $Q = Q_+ + Q_-$. Define the following matrices:

$$\widetilde{P} = \begin{pmatrix} P & 0 \\ 0 & (Q_+ - Q_-)^{-1} \end{pmatrix} \tag{20}$$

$$\widetilde{Q} = \begin{pmatrix} Q_+ & -Q_- \\ -Q_- & Q_+ \end{pmatrix}. \tag{21}$$

We claim that these two matrices together define a problem symmetry for (15). First of all notice that $Q_+ - Q_-$ is a regular permutation matrix, so its inverse exists and $\widetilde{P}$ is well-defined. Moreover, because of the zero blocks on the off-diagonal, $\widetilde{P}$ does not mix the two kinds of constraints in (15) together.

We have to check the objective, the right-hand side and the coefficient matrix of (15) under this symmetry:

The objective function is easy:

$$\widetilde{Q}^T \begin{pmatrix} c \\ -c \end{pmatrix} = \begin{pmatrix} (Q_+ + Q_-)^T c \\ -(Q_- + Q_+)^T c \end{pmatrix} = \begin{pmatrix} Q^T c \\ -Q^T c \end{pmatrix} = \begin{pmatrix} c \\ -c \end{pmatrix}, \tag{22}$$

where we used (9) in the last step.

The right-hand side is also very straightforward. It is obvious that we only need to check the effect of P, as the lower right block of $\widetilde{P}$ acts on an all-1 vector on the right-hand side:

$$\begin{aligned} P(2b - Ae) &= 2Pb - PAe = 2(b + PAq) - PAe = \\ &= 2b + 2PA(e - Qe)/2 - PAe = 2b + PAe - PAQe - PAe = \\ &= 2b - Ae, \end{aligned} \tag{23}$$

where we used the definition of q and the requirements (11). Finally, we can turn to the coefficient matrix:

$$\begin{aligned} \widetilde{P} \begin{pmatrix} A & -A \\ I & I \end{pmatrix} \widetilde{Q} &= \begin{pmatrix} P & 0 \\ 0 & (Q_+ - Q_-)^{-1} \end{pmatrix} \begin{pmatrix} A & -A \\ I & I \end{pmatrix} \begin{pmatrix} Q_+ & -Q_- \\ -Q_- & Q_+ \end{pmatrix} = \\ &= \begin{pmatrix} P & 0 \\ 0 & (Q_+ - Q_-)^{-1} \end{pmatrix} \begin{pmatrix} AQ_+ + AQ_- & -AQ_- - AQ_+ \\ Q_+ - Q_- & -Q_- + Q_+ \end{pmatrix} = \\ &= \begin{pmatrix} P & 0 \\ 0 & (Q_+ - Q_-)^{-1} \end{pmatrix} \begin{pmatrix} AQ & -AQ \\ Q_+ - Q_- & Q_+ - Q_- \end{pmatrix} = \\ &= \begin{pmatrix} PAQ & -PAQ \\ (Q_+ - Q_-)^{-1}(Q_+ - Q_-) & (Q_+ - Q_-)^{-1}(Q_+ - Q_-) \end{pmatrix} = \\ &= \begin{pmatrix} A & -A \\ I & I \end{pmatrix}. \end{aligned} \tag{24}$$

This shows that $\widetilde{P}$ and $\widetilde{Q}$ are indeed symmetries of (15).

Now let us prove the opposite direction. Take a regular permutation symmetry of (15). Obviously, we cannot map the different kinds of constraints into each other, so the row permutation matrix will look like this:

$$\begin{pmatrix} P_1 & 0 \\ 0 & P_2 \end{pmatrix}, \tag{25}$$

where P_1 and P_2 are permutation matrices acting on the two kinds of constraints. The column permutations however can be quite complex, so for now let us just partition it according to the block structure of the problem:

$$\begin{pmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{pmatrix}. \tag{26}$$

These matrices satisfy the following:

$$\begin{pmatrix} P_1 & 0 \\ 0 & P_2 \end{pmatrix} \begin{pmatrix} A & -A \\ I & I \end{pmatrix} \begin{pmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{pmatrix} = \begin{pmatrix} A & -A \\ I & I \end{pmatrix} \tag{27}$$

$$\begin{pmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{pmatrix} \begin{pmatrix} c \\ -c \end{pmatrix} = \begin{pmatrix} c \\ -c \end{pmatrix} \tag{28}$$

$$\begin{pmatrix} P_1 & 0 \\ 0 & P_2 \end{pmatrix} \begin{pmatrix} 2b - Ae \\ e \end{pmatrix} = \begin{pmatrix} 2b - Ae \\ e \end{pmatrix} \tag{29}$$

Now working out the lower two terms of (27) we get

$$P_2(Q_1 + Q_3) = I$$
$$P_2(Q_2 + Q_4) = I,$$

which implies $Q_1 + Q_3 = Q_2 + Q_4 = P_2^{-1}$. We are now going to prove that $Q_1 = Q_4$ and $Q_2 = Q_3$. For this let us consider the case when ${Q_1}_{ij} = 1$ for some indices i and j. Since the entire matrix is a permutation matrix, this implies that ${Q_2}_{ik} = 0$ and ${Q_3}_{kj} = 0$ for all values of k. Using $Q_1 + Q_3 = Q_2 + Q_4$ we get that

$${Q_4}_{ij} = (Q_2 + Q_4)_{ij} = (Q_1 + Q_3)_{ij} = 1. \tag{30}$$

Swapping the role of Q_1 and Q_4 we get that ${Q_4}_{ij} = 1$ implies ${Q_1}_{ij} = 1$, so a coefficient in Q_1 is 1 if and only if it is also 1 in Q_4. This proves that $Q_1 = Q_4$. The $Q_2 = Q_3$ result is proved similarly.

Now we can unambiguously define

$$Q := Q_1 - Q_2 = Q_4 - Q_3, \text{ and} \tag{31}$$

$$q := \frac{e - Qe}{2} \tag{32}$$

Now we claim that (Q, q) together with P_1 is a signed symmetry for (1), for which we only need to check the three conditions in (11).

The objective:

$$Q^T c = (Q_1 - Q_2)^T c = c$$

because of (28). The coefficient matrix (using $Q = (Q_1 - Q_2 + Q_4 - Q_3)/2$)

$$\begin{aligned} P_1 A Q &= P_1 A(Q_1 - Q_2 + Q_4 - Q_3)/2 = \\ &= P_1 A(Q_1 - Q_3)/2 - P_1 A(Q_2 - Q_4)/2 = A/2 - (-A)/2 = A, \end{aligned}$$

where we used the upper part of (27). Finally, the right-hand side:

$$\begin{aligned} P_1(b - Aq) &= P_1 b - P_1 A(e - Qe)/2 = P_1(2b - Ae)/2 + P_1 AQe/2 = \\ &= (2b - Ae)/2 + Ae/2 = b, \end{aligned}$$

where we used the upper part of (29).

Notice that the correspondence is indeed a bijection, as all the possible permutation symmetries of (15) must have the structure of $\widetilde{P}$ and $\widetilde{Q}$.

This completes the proof of Theorem 1.

Remark 1. The additional constraints in (15) ensure that the complement of the image is the image of the complement. They are necessary and cannot be dropped, otherwise there is nothing to link a variable to its complement. This is shown by the fact that they were used to prove that a permutation symmetry of (15) has the special block structure.

4.1 Implementation details

As the lifted problem (15) uses only data that is present in the original problem (1) we can modify the symmetry detection algorithms (see, e.g., [3]) to search for complement symmetries as well. This will incur only a moderate storage overhead, but obviously, the coefficient matrix would not have to be stored in its lifted form, the algorithm can be custom tailored to operate only on the original data.

4.2 Example

Consider again problem (14) depicted in Figure 2. The lifted version is:

$$\begin{aligned} \min\ (x_1 - x_2 + x_3 - \bar{x}_1 + \bar{x}_2 - \bar{x}_3 + 1)/2 & \\ x_1 - x_2 + x_3 - \bar{x}_1 + \bar{x}_2 - \bar{x}_3 \le 0 & \\ x_i + \bar{x}_i = 1,\ i = 1,2,3 & \\ x_1, x_2, x_3, \bar{x}_1, \bar{x}_2, \bar{x}_3 \text{ binary.} & \end{aligned} \tag{33}$$

Running symmetry detection on this problem we get that x_1, $\bar{x}_2$ and x_3 belong to the same orbit, exactly as we expected by looking at Figure 2. The symmetries for problem (33) are $(1\bar{2}3)$ and, equivalently, $(\bar{1}2\bar{3})$.

4.3 Using signed symmetries in heuristics

Signed symmetries can be used in the regular way in orbital branching and orbital fixing, see [2] for details. However, a probably more useful way to use them is to apply them to feasible solutions. Signed permutations can map a solution into a broader set of solutions by being able to change the number of nonzeros. Also, by modifying problem (34) slightly, we can use signed symmetries for orbit MIPping. Again, let $\tilde{x}$ be a feasible solution and let $\mathcal{O}_1, \ldots, \mathcal{O}_k$ be the signed orbit partition of problem (1).

$$\begin{aligned} \min\ & c^T x \\ & Ax \le b \\ \sum_{i \in \mathcal{O}_j} x_i + \sum_{\bar{i} \in \mathcal{O}_j} (1 - x_i) &= \sum_{i \in \mathcal{O}_j} \tilde{x}_i + \sum_{\bar{i} \in \mathcal{O}_j} (1 - \tilde{x}_i),\ \forall j = 1, \ldots, k \\ & x \in \{0,1\}^n , \end{aligned} \tag{34}$$

where $\bar{i} \in \mathcal{O}_j$ denotes the relation that the complement of variable $\tilde{x}_i$ belongs to the orbit $\mathcal{O}_j$.

5 Extensions and future work

The developments in this paper can be naturally extended to problems containing other types of variables. We chose a pure binary problem only to simplify the notation. Also, with a slight modification to the lifted problem (15) it is possible to map the lower bound of any boxed variable to the upper bound of a boxed variable. Looking at the conditions in (11) we have showed that signed symmetries are the largest class of symmetries that also map binary variables to binary variables. However, these conditions can be satisfied by other symmetries, which are thus not symmetries of the hypercube. These can be useful for special problem classes. These symmetries are the subject of future research.

References

1. Ostrowski, J.: Symmetry in Integer Programming. Ph.D. dissertation, Lehigh University (2008)
2. Ostrowski, J., Linderoth, J.T., Rossi, F., Smriglio, S.: Orbital branching. Mathematical Programming **126** (2011) 147–178
3. Katebi, H., Sakallah, K.A., Markov., I.L.: Graph symmetry detection and canonical labeling: Differences and synergies. CoRR (abs/1208.6271) (2012)
4. Koch, T., Achterberg, T., Andersen, E., Bastert, O., Berthold, T., Bixby, R.E., Danna, E., Gamrath, G., Gleixner, A.M., Heinz, S., Lodi, A., Mittelmann, H., Ralphs, T., Salvagnin, D., Steffy, D.E., Wolter, K.: MIPLIB 2010. Mathematical Programming Computation **3**(2) (2011) 103–163
5. Buchheim, C., Jünger, M.: Linear optimization over permutation groups. Discrete Optimization **2**(4) (2005) 308–319
6. Kerber, A.: Representations of permutation groups. I. Volume 240 of Lecture Notes in Mathematics. Springer-Verlag, Berlin, New York (1971)