Constraint Programming Letters 1 (2007) Submitted 1/2007; Published

Breaking Symmetry in Injective Problems

Jean-François Puget JFPUGET@ILOG.FR
ILOG
9 av de Verdun
94253 Gentilly CEDEX, FRANCE

Editor:

Abstract

Adding symmetry breaking constraints is one of the oldest ways of breaking variable symmetries for CSPs. For instance, it is well known that all the symmetries for the pigeon hole problem can be removed by ordering the variables. We have generalized this result to all CSPs where the variables are subject to an all different constraint. In such case it is possible to remove all variable symmetries with a partial ordering of the variables. We shows how this partial ordering can be automatically computed using computational group theory (CGT). We further show that partial orders can be safely used together with the GE-tree method. Various experiments show the efficiency of our method.

Keywords: Symmetry breaking, Alldifferent, Computational Group Theory

1. Introduction

A symmetry for a Constraint Satisfaction Problem (CSP) is a mapping of the CSP onto itself that preserves its structure as well as its solutions. If a CSP has some symmetries, it may be the case that all symmetrical variants of every dead end encountered during the search must be explored before a solution can be found. Even if the problem is easy to solve, all symmetrical variants of a solution are also solutions, and listing all of them may just be impossible in practice. Among symmetries, two categories have been studied in detail : variable symmetries, and value symmetries. A variable symmetry is a permutation of variables that leave a given CSP invariant. A value symmetry is a permutation of values that leave the CSP invariant. Both kind of symmetries can be combined.

Let us introduce an example that will be used throughout the paper. This problem is the sports league scheduling (problem 026 in the CSPLIB [4]). The problem is to schedule a tournament of n teams over $n-1$ weeks, with each week divided into $n/2$ periods. A game between two teams must occur every period of every week. A tournament must satisfy the following three constraints : every team plays once a week; every team plays at most twice in the same period over the tournament; every team plays every other team. A natural model for this problem is to introduce a matrix of variables x_{ij} representing the game played during period i of week j. The values of the variables are the possible games.

Here is a solution for $n = 6$ (note that games i vs. j and j vs. i are the same) :

0 vs. 1	0 vs. 2	2 vs. 4	3 vs. 5	1 vs. 4
2 vs. 5	1 vs. 3	1 vs. 5	0 vs. 4	2 vs. 3
3 vs. 4	4 vs. 5	0 vs. 3	1 vs. 2	0 vs. 5

This problem has many symmetries. First of all, weeks can be exchanged. This means that the columns of the matrix can be freely exchanged. This is a variable symmetry. For instance, the following is a solution of the problem obtained by swapping the first two columns :

0 vs. 2	0 vs. 1	2 vs. 4	3 vs. 5	1 vs. 4
1 vs. 3	2 vs. 5	1 vs. 5	0 vs. 4	2 vs. 3
4 vs. 5	3 vs. 4	0 vs. 3	1 vs. 2	0 vs. 5

Second, the periods can be exchanged. This means that the rows of the matrix can be freely permuted. This is a variable symmetry. For instance, swapping the first two rows yields the following solution :

1 vs. 3	2 vs. 5	1 vs. 5	0 vs. 4	2 vs. 3
0 vs. 2	0 vs. 1	2 vs. 4	3 vs. 5	1 vs. 4
4 vs. 5	3 vs. 4	0 vs. 3	1 vs. 2	0 vs. 5

Third, the teams themselves can be exchanged. Any permutation of teams defines a permutation of the games. This is a value symmetry. For instance, swapping teams 1 and 2 yields the following solution :

2 vs. 3	1 vs. 5	2 vs. 5	0 vs. 4	1 vs. 3
0 vs. 1	0 vs. 2	1 vs. 4	3 vs. 5	2 vs. 4
4 vs. 5	3 vs. 4	0 vs. 3	1 vs. 2	0 vs. 5

The three types of symmetries can be independently applied. Therefore, there are $(n-1)!(n/2)!n!$ symmetries in this problem, which is 4877107200 for $n = 8$. Listing all solutions may be impossible for $n = 8$, not to speak about larger values for n.

Adding symmetry breaking constraints is one of the oldest ways of breaking variable symmetries for constraint satisfaction problems (CSPs). For instance, it is shown in [2] that all variable symmetries could be broken by adding one lexicographical ordering constraint per symmetry. Unfortunately, this method is not tractable in general, as there may be an exponential number of symmetries. It has been shown that in general there is no way to break all symmetries of a problem with a polynomial number of lexicographic constraints[22]. However, several polynomial cases have been proposed recently. In [3], a linear number of constraints are used to break symmetries for matrix problems. As expected, since there are a polynomial number of constraints, not all symmetries are broken. We present in this paper a class of problems for which all variable symmetries can be broken with a polynomial number of constraints.

We consider the class of *injective problems* where the variables are subject to an all different constraint, among other constraints. In these problems, the mapping from variables to values is injective by definition of the alldifferent constraint. After some definitions given in section 2, we show in section 3 that for such CSPs, all variable symmetries can be broken with at most $n-1$ binary constraints, where n is the number of variables. We then illustrate our method on several examples in section 4.

In [21] a general purpose method for breaking all value symmetries is given : the GE-tree method. We show in section 5 that this method can be safely combined with symmetry

breaking constraints, under some conditions on the order in which the search tree is traversed.

In section 6, we perform various experiments using complex problems. We summarize our findings and discuss some possible generalizations in section 7.

2. Symmetry, Graphs and CSPs

The symmetries we consider are permutations, i.e. one to one mappings (bijections) from a finite set onto itself. Without loss of generality, we can consider permutations of I^n, where I^n is the set of integers ranging from 1 to n. For instance, we can label the variables of a graph with integers, such that any variable symmetry is completely described by a permutation of the labels of its variables. This is formalized as follows.

Let S^n be the set of all permutations of the set I^n. The image of i by the permutation σ is denoted i^σ.

A permutation $\sigma \in S^n$ is fully described by the vector $[1^\sigma, 2^\sigma, \ldots, n^\sigma]$. The product of two permutations σ and θ is defined by $i^{(\sigma\theta)} = (i^\sigma)^\theta$.

Given $i \in I^n$ and a permutation group $G \subseteq S^n$, the *orbit* of i in G, denoted i^G, is the set of elements to which i can be mapped to by an element of G :

$$i^G = \{i^\sigma | \sigma \in G\}$$

Given $i \in I^n$ and a permutation group $G \subseteq S^n$, the *stabilizer* of i in G, denoted i_G, is the set of permutations of G that leave i unchanged :

$$i_G = \{\sigma \in G | i^\sigma = i\}$$

A *constraint satisfaction problem* $\mathcal{P}$ (CSP) with n variables is a triple $\mathcal{P} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$ where $\mathcal{V}$ is a finite set of variables $(v_i)_{i \in I^n}$, $\mathcal{D}$ a finite set of finite sets $(\mathcal{D}_i)_{i \in I^n}$, and every constraint in $\mathcal{C}$ is a subset of the cross product $\bigotimes_{i \in I^n} \mathcal{D}_i$. Without loss of generality, we can assume that $\mathcal{D}_i \subseteq I^k$ for some k.

A *literal* is a statement of the form $x_i = j$ where $j \in \mathcal{D}_i$.

An *assignment* is a set of literals, one for each variable of the CSP. A *partial* assignment is a subset of an assignment.

A *solution* to $(\mathcal{V}, \mathcal{D}, \mathcal{C})$ is an assignment that is consistent with every member of $\mathcal{C}$.

A *symmetry* is a bijection from literals to literals that maps solutions to solutions. Our definition is similar to the solution symmetries in [1].

A *variable symmetry* is a symmetry g such that there is a permutation σ of the variables such that $(x_i = j)^g = (x_{i^\sigma} = j)$. In such case, we will denote g by σ :

$$(x_i = j)^\sigma = (x_{i^\sigma} = j) \tag{1}$$

A *value symmetry* is a symmetry g such that there exists a permutation θ of I^n such that $(x_i = j)^g = (x_i = j^\theta)$. In such case we will denote g by θ :

$$(x_i = j)^\theta = (x_i = j^\theta) \tag{2}$$

The set of symmetries of a CSP forms a mathematical group. Indeed, the composition of two symmetries is a symmetry, the identity function is a symmetry, and any symmetry

can be inverted. This observation led to the publication of various symmetry breaking techniques that use the properties of the symmetry group. Those use Computational Group Theory(CGT) [24] : GAP-SBDD [6], GAP-SBDS [5], and GE-tree [21]. In the next section, we will see how CGT can be used to compute a set of symmetry breaking constraints.

3. Breaking Variable Symmetry

Adding constraints is one of the oldest methods for reducing the number of variable symmetries of a CSP[14].

3.1 Lex Leader Constraints

In [2], it is shown that all the variable symmetries of any CSP can be broken by the following constraints.

$$\forall \sigma \in G, \ \mathcal{V} \preceq \mathcal{V}^\sigma \tag{3}$$

For a given σ, the constraint $(\mathcal{V} \preceq \mathcal{V}^\sigma)$ is semantically equivalent to the disjunction of the constraints :

$$v_1 < v_{1^\sigma}$$

$$v_1 = v_{1^\sigma} \wedge v_2 < v_{2^\sigma}$$

$$\vdots$$

$$v_1 = v_{1^\sigma} \wedge \ldots \wedge v_{i-1} = v_{(i-1)^\sigma} \wedge v_i < v_{i^\sigma}$$

$$\vdots$$

$$v_1 = v_{1^\sigma} \wedge \ldots \wedge v_{n-1} = v_{(n-1)^\sigma} \wedge v_n < v_{n^\sigma}$$

$$v_1 = v_{1^\sigma} \wedge \ldots \wedge v_{n-1} = v_{(n-1)^\sigma} \wedge v_n = v_{n^\sigma}$$

If the last constraint is omitted, the set of constraints is denoted $\mathcal{V} \prec \mathcal{V}^\sigma$.

3.2 A Polynomial Number of Constraints

The size of G, hence the number of constraints (3), can grow exponentially with the number of variables. Using the fact that the variables $\mathcal{V}$ are subject to an all different constraint, we can significantly reduce the number of symmetry breaking constraints.

Given a permutation σ, let $s(\sigma)$ be the smallest i such that $i^\sigma \neq i$, and let $r(\sigma)$ be equal to $(s(\sigma))^\sigma$. By definition $k^\sigma = k$ for all $k < s(\sigma)$, and $s(\sigma)^\sigma \neq s(\sigma)$. Let us now look at the constraint $\mathcal{V} \preceq \mathcal{V}_\sigma$. There is an all different constraint on the variables $\mathcal{V}$, which means that $v_i = v_{i^\sigma}$ if and only if $i^\sigma = i$. In particular, $v_k = v_{k^\sigma}$ for all $k < s(\sigma)$, and $v_{s(\sigma)} \neq v_{(s(\sigma))^\sigma}$. Therefore, only one disjunct for the constraint can be true, namely :

$$v_1 = v_{1^\sigma} \wedge \ldots \wedge v_{s(\sigma)-1} = v_{(s(\sigma)-1)^\sigma} \wedge v_{s(\sigma)} < v_{(s(\sigma))^\sigma}$$

Since $k^\sigma = k$ for $k < s(\sigma)$ and $s(\sigma)^\sigma = r(\sigma)$, this can be simplified into

$$v_{s(\sigma)} < v_{r(\sigma)}$$

We have just proved the following result.

Lemma 1. *Given a CSP where the variables $\mathcal{V}$ are subject to an all different constraint, and a variable symmetry group G for this CSP, then all variable symmetries can be broken by adding the following constraints :*

$$\forall \sigma \in G, v_{s(\sigma)} < v_{r(\sigma)} \tag{4}$$

Note that if two permutations σ and θ are such that $s(\sigma) = s(\theta)$ and $r(\sigma) = r(\theta)$, then the corresponding symmetry breaking constraints are identical. Therefore, it is sufficient to state only one symmetry breaking constraints for each pair i, j such that there exists a permutation σ with $i = s(\sigma)$ and $j = r(\sigma)$.

The set of these pairs can be computed using what is known as the Schreier Sims algorithm[24]. This algorithm constructs a stabilizers chain $G_0, G_1, \ldots, G_n$ as follows :

$$G_0 = G$$

$$\forall i \in I^n,\ G_i = i_{G_{i-1}}$$

By definition,

$$G_i = \{\sigma \in G : 1^\sigma = 1 \wedge \ldots \wedge i^\sigma = i\}$$

and so

$$G_n = \{id\}$$

$$G_n \subseteq G_{n-1} \subseteq \ldots G_1 \subseteq G_0$$

The Schreier Sims algorithm also computes set of coset representatives U_i. Those are orbits of i in G_{i-1} :

$$U_i = i^{G_{i-1}}$$

By definition, U_i is the set of values which i is mapped to by all symmetries in G that leave at least $1, \ldots, (i-1)$ unchanged.

From now on, we will assume that all the groups we use are described by a stabilizers chain and coset representatives.

By definition, for each element $j \in U_i$, there exists at least one permutation $\sigma \in G_{i-1}$ such that $i^\sigma = j$ and $j = r(\sigma)$. The converse is also true. If there exists a permutation σ such that $i = s(\sigma)$ and that $j = r(\sigma)$, then $j \in U_i$. Therefore, the constraints (4) can be rewritten into :

$$\forall i \in I^n, \forall j \in U_i,\ i \neq j \Rightarrow v_i < v_j$$

There are $\sum_{i=1}^{n}(|U_i|-1)$ such constraints. All the permutations of G_{i-1} leave the numbers $1,\ldots,i-1$ unchanged. Therefore U_i is a subset of $\{i,\ldots,n\}$. Then $|U_i|-1 \leq n-i$. Therefore, the number of constraints is bounded from above by $\sum_{i=1}^{n}(n-i) = n(n-1)/2$. We have just proved the following result.

Theorem 2. *Given a CSP with n variables $\mathcal{V}$ such that there exists an all different constraint on these variables, and given coset representatives sets U_i for the variable symmetry group of the CSP, then all the variable symmetries can be broken by at most $n(n-1)/2$ binary constraints. These constraints are given by :*

$$\forall i \in I^n, \forall j \in U_i,\ i \neq j \rightarrow v_i < v_j \tag{5}$$

3.3 A Linear Number of Constraints

The previous result can be improved by taking into account the transitivity of the $<$ constraints. Given $j \in I^n$, it may be the case that j belongs to several of the sets U_i. In such case, let us define $r(j)$ as the largest i different from j such that j belongs to U_i. If j belongs to no U_i other than U_j, then let $r(j) = j$.

Before stating our main result, let us prove the following.

Lemma 3. *With the above notations, if $j \in U_i$ and $i \neq j$ then $r(j) \in U_i$ and $r(j) < j$*

Proof. Let us assume that $j \in U_i$ and $i \neq j$. By definition of U_i there exists a permutation $\sigma \in G_{i-1}$ such that $i^\sigma = j$. Let $k = r(j)$. By definition of $r(j)$, $i \leq k$ and $j \in U_k$. Therefore, there exists a permutation $\theta \in G_{k-1}$ such that $k^\theta = j$. Let $\nu = \sigma\theta^{-1}$. Then, $i^\nu = i^{\sigma\theta^{-1}} = j^{\theta^{-1}} = k$. Moreover, $\nu \in G_{i-1}$ because $\sigma \in G_{i-1}$ and $\theta \in G_{k-1} \subseteq G_{i-1}$. Therefore, $k \in U_i$. The fact that $r(j) < j$ is an immediate consequence of the definition of $r(j)$.

We can now state our main result.

Theorem 4. *With the above notations, given a CSP with n variables $\mathcal{V}$, such that there exists an all different constraint on these variables, then all variable symmetries can be broken by at most $n-1$ binary constraints. These constraints are given by :*

$$\forall j \in I^n,\ r(j) \neq j \rightarrow v_{r(j)} < v_j \tag{6}$$

Proof. The number of constraints (6) is at most n by definition. Note that $r(1) = 1$ by definition of r, therefore, the number of constraints is at most $n-1$. Let us consider one of the constraints of (5). We are given i and j such that $j \in U_i$ and $i \neq j$. We want to prove that the constraint $c = (v_i < v_j)$ is implied by the constraints (6). Let us consider the sequence $(j, r(j), r(r(j)), r(r(r(j))), \ldots)$. Let us assume that the sequence never meets i. We have that $j \in U_i$ and $i \neq j$. By application of lemma 3, we get $r(j) \in U_i$ and $r(j) < j$. Since $r(j) \neq i$ by hypothesis, lemma 3 can be applied again. By repeated applications of lemma 3 we construct an infinite decreasing sequence of integers all included in U_i. This is not possible as U_i is finite. Therefore, there exists k such that $i = r^k(j)$. Moreover, we

have established $r^k(j) \neq r^{k-1}(j), \ldots, r(r(j)) \neq r(j), r(j) \neq j$. Therefore, the constraints $v_{r^k(j)} < v_{r^{k-1}(j)}, \ldots v_{r(r(j))} < v_{r(j)}, v_{r(j)} < v_j$ are constraints of (6). Together they imply $v_{r^k(j)} < v_j$ which is the constraint c. We have proved that the constraints (5) are implied by the constraints (6). Since the set of constraints (6) is a subset of the constraints (5), both sets of constraints are equivalent. Then, by theorem 2, the constraints (6) break all variable symmetries.

In fact, Theorem 4 can be slightly generalized by relaxing the surjection condition into a weaker form where any pair of variables appearing in the same orbit are different :

$$\forall i, j,\ j \in U_i \rightarrow x_i \neq x_j \tag{7}$$

Indeed, we only used the above condition in the proof of the theorem.

4. Some Examples

Let us see how our method works on various examples.

4.1 The Pigeon Hole Problem

Let us look first at a simple example, namely the pigeon hole problem. We are given n variables $(v_i)_{i \in I^n}$ with domains equal to I^{n-1}. There is an all different constraint on these variables. The group of variable symmetries for this problem is the set of all permutations S^n. The stabilizers chain is :

$$G_0 = S^n$$

$$\forall i \in I^n,\ G_i = \{\sigma \in S^n | \forall k \leq i, k^\sigma = k\}$$

The coset representatives are given by :

$$\forall i \in I^n,\ U_i = \{i, i+1, \ldots, n\}$$

From this we get :

$$r(1) = 1, r(2) = 1, \ldots, r(i) = i - 1, \ldots, r(n) = n - 1$$

The symmetry breaking constraints (6) are therefore :

$$v_1 < v_2, v_2 < v_3, \ldots, v_{i-1} < v_i, \ldots, v_{n-1} < v_n$$

Bound propagation on these constraints is sufficient to show that the CSP has no solution. Note that we only used $n-1$ constraints, whereas (3) contains an exponential number of constraints in this case.

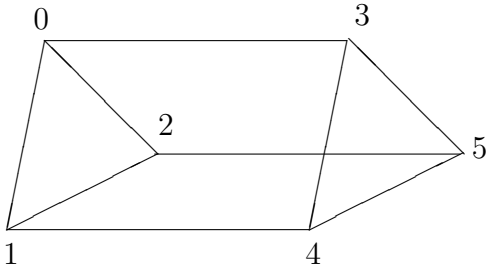
4.2 Graceful Graphs

Let us look at a more complex example than the pigeon hole problem. We say that a graph with m edges is *graceful* if there exists a labeling f of its vertices such that :

- $0 \leq f(i) \leq m$ for each vertex i,
- the set of values $f(i)$ are all different,
- the set values $|f(i), f(j)|$ for each edge (i, j) are all different.

A straightforward translation into a CSP exists where there is a variable v_i for each vertex v_i, see [10]. The variable symmetries of the problem are induced by the automorphism of the graph. There is one value symmetry, which maps v to $m - v$. More information on symmetries in graceful graphs is available in [13].

Theorem 4 states that all the variables symmetries can be broken by $n - 1$ binary constraints at most, where n is the number of vertices. Let us consider the following graph $K_3 \times P_2$[1] :



The group of variable symmetries of the corresponding CSP is equivalent to the group of symmetries of the graph. Such group can be computed by packages such as Nauty[11] or AUTOM[17]. This group G is :

$$\{[0,1,2,3,4,5],[0,2,1,3,5,4],[1,0,2,4,3,5],$$

$$[1,2,0,4,5,3],[2,0,1,5,3,4],[2,1,0,5,4,3],$$

$$[3,4,5,0,1,2],[3,5,4,0,2,1],[4,3,5,1,0,2],$$

$$[4,5,3,1,2,0],[5,3,4,2,0,1],[5,4,3,2,1,0]\}$$

The stabilizer chain is

$$G_0 = G$$

$$G_1 = 0_{G_0} = \{[0,1,2,3,4,5],[0,2,1,3,5,4]\}$$

$$G_2 = 1_{G_1} = \{[0,1,2,3,4,5]\}$$

All remaining stabilizers G_3, G_4, G_5 are equal to G_2.

Coset representatives are :

$$U_1 = 0^{G_0} = \{0,1,2,3,4,5\}$$

$$U_2 = 1^{G_1} = \{1,2\}$$

$$U_3 = 2^{G_2} = \{2\}$$

1. Vertices numbers start from 0 instead of 1 in this example.

All remaining coset representatives U_4, U_5 are equal to U_3.
From coset representatives we get :

$$r(0) = 0,\ r(1) = 0,\ r(2) = 1,\ r(3) = 0,\ r(4) = 0,\ r(5) = 0$$

Therefore, the constraints (6) given by theorem 4 are :

$$v_0 < v_1,\ v_0 < v_3,\ v_0 < v_4,\ v_0 < v_5,\ v_1 < v_2$$

4.3 Sports League Scheduling

Let us see how this method works on the example given in the introduction. It can be modeled with a 3 by 5 matrix model as follows :

x_0	x_1	x_2	x_3	x_4
x_5	x_6	x_7	x_8	x_9
x_{10}	x_{11}	x_{12}	x_{13}	x_{14}

Variable symmetries are generated by column permutations and by row permutations. We will identify a variable and its index. We first consider the group of variable symmetries G_0. We compute the orbit of 0 in this group. This is the set of variables to which x_0 can be mapped to using any permutation of both row and columns. It is easy to see that any variable can be reached this way. Therefore, the orbit of 0 is the set of all variables :

$$U_0 = 0^{G_0} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$$

We then consider the stabilizer G_1 of 0 in G_0. This is the set of symmetries that leave x_0 unchanged. Any symmetry in G_1 maps the first row to itself, and the first column to itself. The other row can be permuted, and the other columns can be permuted. Then we compute the orbit of 1 in G_1. It is easy to see that this is the first row except for 0. Therefore, the orbit of 1 in G_1 is :

$$U_1 = 1^{G_1} = \{1, 2, 3, 4\}$$

We then compute the stabilizer of 1, etc. This yields

$$\begin{array}{lllllll} G_2 & = & 1_{G_1}, & U_2 & = & 2^{G_2} & = \{2,3,4\} \\ G_3 & = & 2_{G_2}, & U_3 & = & 3^{G_3} & = \{3,4\} \\ G_4 & = & 3_{G_3}, & U_4 & = & 4^{G_4} & = \{4\} \\ G_5 & = & 4_{G_3}, & U_5 & = & 5^{G_5} & = \{5,10\} \\ G_i & = & \{id\}, & U_i & = & \{i\} & \forall i \geq 6 \end{array}$$

Then, we get :

$$\begin{array}{rcl} \forall i = 1, \ldots, 4,\ r(i) & = & i - 1 \\ r(5) & = & 0 \\ \forall i = 6, \ldots, 9,\ r(i) & = & 0 \\ r(10) & = & 5 \\ \forall i = 11, \ldots, 14,\ r(i) & = & 0 \end{array}$$

Then, constraints (6) are :

$$\begin{aligned} \forall i = 6, \ldots, 9,\ x_0 &< x_i, \\ \forall i = 11, \ldots, 14,\ x_0 &< x_i, \\ \forall i = 0, \ldots, 3,\ x_i &< x_{i+1} \\ x_0 < x_5 &< x_{10} \end{aligned}$$

In other words, x_0 must be the smallest element in the matrix, the first row must be increasing, and the first column must be increasing. In this case, we find the same constraints as the ones given in [3]. There are only 30 solutions satisfying these constraints. If we do not state these constraints, there are 21600 solutions, many of them being symmetric variants.

5. Breaking Variable and Value Symmetry

In [21], a general method for breaking all value symmetries is described. This method uses the group of value symmetries of the CSP. We will show that this method can be combined with symmetry breaking constraints when there are both variable symmetries and value symmetries. More precisely, we give the proof of the following result in the appendix.

Theorem 5. *Given a CSP where the variable are subject to an all different constraint, its group of variable symmetries G_1, and its group of value symmetries G_2, then the combination of the GE-tree method for breaking value symmetries with the symmetry breaking constraints (6) computes a set of solutions $\mathcal{S}$ such that :*

$$\forall S \in sol(\mathcal{P}), \exists \sigma \in G_1, \exists \theta \in G_2, \exists S' \in \mathcal{S},\ S^{\sigma\theta} = S'$$

In our implementation, we did not fully implement the GE-tree method, because it requires more computational group algorithms than what we have implemented so far. We simply compute the orbits for the group G of value symmetries. Then, only the minimum element of each orbit is left in the domain of the variable v_1. This amounts to use equation (13) given in the appendix.

The previous results can be applied to a prevalent problem in computer science applications, namely sub graph isomorphism (SGI). See [20] for applications of SGI in chemistry for instance. The SGI problem is easily cast into a CSP. We are given two graphs, $\mathcal{G}_1$ and $\mathcal{G}_2$ and we want to know if $\mathcal{G}_1$ is isomorphic to a sub graph of $\mathcal{G}_2$. A CSP is constructed where there is one variable v_i per vertex i of $\mathcal{G}_1$, and where the possible values are the vertices of $\mathcal{G}_2$. The constraints of the CSP express the isomorphism relationship. First of all, the variable must be all different. Second, for each edge $\{i, j\} \in E_1$ a binary constraint states that the edge $\{v_i, v_j\}$ is in E_2. Several global constraints have been devised for enforcing this relationship, see [20][26][15] for instance.

The purpose of this paper is not to discuss how to efficiently enforce the isomorphism relationship. It is rather to look at the symmetries in this CSP. The conditions of theorem 5 applies here : all the variable symmetries can be broken by $|V_1| - 1$ binary constraints, and all the values symmetries can be broken by the GE-tree method. We only need to know the symmetries of the problem. They are induced by the automorphisms of the graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ as follows.

Let $\sigma \in aut(G1)$ be an automorphism of $\mathcal{G}_1$. Then, σ induces a variable symmetry. Indeed, let us assume that we have an isomorphism f between $\mathcal{G}_1$ and a sub graph of $\mathcal{G}_2$. Then, $(v_i = f(i))_{i \in I^n}$ is a solution of the CSP. Let us apply σ to this solution. We get the following assignment :

$$(v_{i^\sigma} = f(i))_{i \in I^n} \tag{8}$$

Let us consider an edge e of $\mathcal{G}_1$. By automorphism of σ, there exists an edge $e' = \{i, j\}$ of $\mathcal{G}_1$ such that $e = e'^\sigma$, i.e. $e = \{i^\sigma, j^\sigma\}$. By isomorphism of f, $f(e') = \{f(i), f(j)\}$ is an edge of $\mathcal{G}_2$. We have proved :

$$\forall i, j \in V_1,\ \{i^\sigma, j^\sigma\} \in E_1 \Rightarrow \{f(i), f(j)\} \in E_2$$

This means that the assignment (8) is a solution of the CSP. It proves that σ is a variable symmetry.

Similarly, let $\theta \in aut(\mathcal{G}_2)$ be an automorphism of $\mathcal{G}_2$. Then, θ induces a value symmetry. Indeed, let us apply θ to the solution $(v_i = f(i))_{i \in I^n}$ of the SGI CSP. We get the following assignment :

$$(v_i = f(i)^\theta)_{i \in I^n} \tag{9}$$

Let us consider an edge $e = \{i, j\}$ of $\mathcal{G}_1$. By isomorphism of f, $f(e) = \{f(i), f(j)\}$ is an edge of $\mathcal{G}_2$. By automorphism of θ, $f(e)^\theta = \{f(i)^\theta, f(j)^\theta\}$ is also an edge of $\mathcal{G}_2$. We have proved that :

$$\forall\ i, j \in V_1,\ \{i, j\} \in E_1 \Rightarrow \{f(i)^\theta, f(j)^\theta\} \in E_2$$

This means that the assignment (9) is a solution for the CSP. It proves that θ is a value symmetry.

Therefore, in order to break variable and values symmetries of the SGI problem, we need to compute the symmetries of the two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$. This can be done by automorphism packages such as NAUTY[11] or AUTOM[17].

6. Experimental Results

We use ILOG Solver[7] for solving our CSPs. Symmetries are automatically detected using the method of [17]. Then, we use a Schreier Simms algorithm to state the constraints (6) of Theorem 4. All the running times are expressed in seconds. They are measured on a 1.4 GHz Pentium M laptop running Windows XP and ILOG Solver 6.3. Unless otherwise stated, the search used to solve the examples is quite straightforward. It is a depth first search where one variable is selected at each node. The order in which variables is selected is the order given by variable indices. Values are tried in increasing order.

6.1 Graceful Graphs

Graceful graphs are introduced in section 4.2. In addition to variable symmetries, these problems have one non trivial value symmetry. If there are q edges, this symmetry maps the value a to the value q-1-a. Therefore, the orbits for this group are the sets $\{a, q-1-a\}$,

for $0 \leq a < q/2$. Our simplified GE-tree method simply removes the largest value in each orbit from the domain of v_0.

We have tested this approach on the graceful graphs of [13]. For each graph, we report the number of solutions of the CSP (sol), the size of the search tree (node) and the time (time) needed to compute all these solutions the running time. We also report these figures when the above symmetry breaking constraints are added (sym). In this case the running time includes the time needed to perform all the group computations. The running times are much better than the ones reported in [13].

graph	no sym			sym		
	sol	node	time	sol	node	time
$K_3 \times P_2$	96	1518	0.12	8	83	0.01
$K_4 \times P_2$	1440	216781	13.6	30	1863	0.27
$K_5 \times P_2$	480	34931511	4454	2	53266	6.5
$K_6 \times P_2$				0	1326585	305

Table 1. Computing all solutions for graceful graphs.

Let us look at the graph $K_5 \times P_2$. This graph has 10 vertices and 25 edges. We list the values for the variables $v_0, v_1, \ldots, v_9$ for the two solutions :

$$(0, 4, 18, 19, 25, 23, 14, 6, 3, 1)$$

$$(0, 6, 7, 21, 25, 24, 22, 19, 11, 2)$$

Let us apply the non trivial value symmetry to the second one. We get :

$$(25, 19, 18, 4, 0, 1, 3, 6, 14, 23)$$

Let us apply the following variable symmetry to it :

$$[4, 3, 2, 1, 0, 9, 8, 7, 6, 5]$$

This yields the first solution.

This example shows that we did not break all symmetries that are a product of a variable symmetry by a value symmetry. It is so despite the fact that all variable symmetries and all value symmetries are broken.

6.2 Most Perfect Magic Squares

Most perfect magic squares, studied in [12], are given as an example of a CSP with convoluted value symmetries in [21]. In [12], it is proven that most perfect magic squares are in a one to one relationship with *reversible squares*. A reversible square of size $n \times n$ (where $n \equiv 0 \mod 4$) has entries $1 \ldots n^2$ such that :

1. The sum or the two entries at diagonally opposite corners of any rectangle or subsquare equals the sum of the other pair of diagonally opposite corners.

2. In each row or column, the sum of the first and last entries equals the sum of the next and the next to last number, etc.

3. Diametrically opposed numbers sum to $n^2 + 1$.

Any solution is one of $2^{n+1}((n/2)!)^2$ symmetric equivalent [12]. For $n = 16$, this is about 2.13e+14.

Our model has one variable per cell with entries as values. Since all entries must be different, we can use the constraints of Theorem 4 to break all variable symmetries.

We report for various sizes the time used to compute the symmetry breaking constraints (CGT) as well as the time used for finding all non symmetrical solutions (search). We also report the results of [21], obtained with GAP-SBDD and with GE-tree on a computer about half the speed of ours. A direct comparison is difficult because they directly search for most perfect magic squares whereas we search for reversible squares. It is worth comparing the time spent in the symmetry computations though, because these deal with the same symmetry group. Our method spends much less time in symmetry computations because these need to be done only once before the search starts.

		us		GAP-SBDD		GE-tree	
n	sols	CGT	search	CGT	search	CGT	search
4	3	0.01	0.02	0.3	0.3	0.2	0.1
8	10	0.09	0.39	5.4	125.4	0.7	90.0
12	42	0.44	22.2	2745	12518	29.1	10901.8
16	35	4.6	275.6				

Table 3. Results for finding all solutions for most perfect magic squares.

6.3 SBDD with Sub Graph Isomorphism

Our implementation of the SBDD method [15] relies on SGI in order to prune search. At each node of the search tree, the SBDD method checks if a previously explored partial assignment is a symmetrical variant of a subset of the current partial assignment. This is called the dominance check in the SBDD method.

For a given node n, a dominance check is performed between n and each previously explored no-good ν. Our method constructs a colored graph for each state. Dominance check amounts to sub graph isomorphism. We need to check if there exists a graph isomorphism between the graph associated with the no-good ν and a sub graph of the graph associated with n.

The sub graph isomorphism can be solved using a separate CSP as explained in section 5. The symmetry breaking techniques discussed in this paper can be applied to these separate CSP as explained before.

We chose to evaluate our method on the balanced incomplete designs (BIBD) (problem 28 in the CSPLib [4]). A BIBD can be represented as a CSP with a v by b matrix model. Each variable in the matrix is a binary variable m_{ij} with domain $\{0, 1\}$. There are three sets of constraints :

1. $\Sigma_{j \in I^b}\ m_{ij} \ = \ r$, for all $i \in I^v$
2. $\Sigma_{i \in I^v}\ m_{ij} \ = \ k$, for all $j \in I^b$
3. $\Sigma_{j \in I^b}\ m_{ij} m_{i'j} \ = \ \lambda$, for all $i \in I^v, i' \in I^v, i < i'$

This problem is called the *master problem* in what follows. Any permutation of the rows or of the columns is symmetry of the master problem.

The graph associated to a given node n explored during the search is constructed as follows. There is a node per line i, and a node per column j. There is an edge between a line i and a column j if, and only if, $m_{ij} = 1$ in that state.

We ran experiments with various instances of the BIBD problem. For every instance, we report the time needed for three variants of SBDD :

- **A** : the running time for the SBDD method of [15]
- **B** : the running time for the method of **A** where constraints (11) are added to the SGI sub problems
- **C** : the running time of the method of **B** where the GE-tree method for breaking value symmetries is used for the SGI sub problems.

All 3 variants explore the same search tree. They also compute the same number of solutions. Those are also reported. The only difference between the 3 methods is how dominance checks are performed. More precisely, the difference is about how the symmetries are handled in the auxiliary CSP used for performing the dominance checks.

We also report the ratio between the time for **A**, and **B** and **C**. The geometrical means of the ratios are given. This shows that **B** is about 1.8 times faster than **A**, and that **C** is about 2.7 times faster than **A** on average. **C** can be up to 50 times faster than the basic SBDD method. Those results are quite impressive, given that we need to perform several dominance check (one for each no good) at each node of the search for the master problem. Each dominance check requires the computation of two graph automorphism groups before solving each SGI sub problem.

BIBD	SOLS	A	B	C	A/B	A/C
6 3 6	6	0.3	0.15	0.1	2.0	3.0
7 3 3	10	0.41	0.31	0.21	1.3	2.0
6 3 8	13	0.8	0.45	0.36	1.8	2.2
15 5 2	0	1.1	0.78	0.77	1.4	1.4
16 6 2	3	1.2	0.82	0.39	1.5	3.1
13 3 1	2	0.55	0.6	0.49	0.9	1.1
9 4 3	11	1.7	1.4	1	1.2	1.7
22 7 2	0	2.3	1.7	1.7	1.4	1.4
6 3 10	19	2.1	1	0.8	2.1	2.6
16 4 1	1	2.1	1.2	0.29	1.8	7.2
7 3 4	35	2.8	1.3	0.97	2.2	2.9
15 7 3	5	3.3	2	0.92	1.7	3.6
9 3 2	36	6.6	3.6	2.7	1.8	2.4
10 5 4	21	7.7	6.5	4.8	1.2	1.6
7 3 5	109	17	7.5	6.4	2.3	2.7
25 5 1	1	48	5.3	0.95	9.1	50.5
7 3 6	418	114	46	43	2.5	2.7
19 9 4	6	139	112	80	1.2	1.7
Mean					1.8	2.7

Table 2. Time for computing all solutions for BIBD.

7. Related Work and Conclusion

We have established a major result (Theorem 4) : all variable symmetries can be broken by a linear number of binary constraints if there is an all different constraints on all the variables of the CSP. These binary constraints define a (partial) ordering on the variables of the problem. This partial ordering can be computed using CGT algorithms once the group of the symmetries of the CSP is given. Furthermore we have proved that this partial ordering can be safely used in conjunction with the GE-tree method of [21].

Our method is fully automated. First, the group of symmetry is computed using the method of [17]. Then, a Schreier Simms algorithm [24] is used to compute the constraints (6) of Theorem 4. Experimental results on various complex problems show that these algorithms are quite efficient.

Our method is also simple enough that it can be applied manually. We provide several examples in section 4 that show how to apply it.

A number of recent work can be related to ours. First, the partial order described by Theorem 4 depends on the variable ordering which is chosen when computing the stabilizer chain. Different variable ordering can yield to different sets of constraints. This is studied extensively in [25]. Our method can be applied for breaking all value symmetries in surjection problems : the idea is to break the symmetries on variables representing the first occurrence of each value [18]. We have shown in [18] that we can safely combine variable symmetry breaking constraints with value symmetry breaking constraints. The proof is similar to the one given in appendix. More recently, we have proposed a new set of constraints that breaks all value and all variable symmetries [19]. A similar approach has been later proposed in [27]. However, both approach may require an exponential number of constraints in order to break all symmetries. Another way of generalizing our work is to look for classes of problems for which a polynomial number of constraints is sufficient for breaking all symmetries. A significant step in that direction is the SBS method of [23]. Our way of breaking symmetries in SGI problems has been first proposed in [16]. It has been rediscovered since in [28].

Another way of generalizing our work is to see how it could be applied to local symmetries, i.e. symmetries that appear during the search for solutions. Indeed, we have so far only considered symmetries that were present at the root node. One way of doing it is to compute the group of local symmetries, using the method of ([17]), at each node. Then we can state the constraints (6) to break these symmetries. However, this may be quite expensive. It may be possible not to start from scratch at each node.

Acknowledgments

I thank Marie Puget for her thorough proof reading.

References

[1] David A. Cohen, Peter Jeavons, Christopher Jefferson, Karen E. Petrie, Barbara M. Smith : "Symmetry Definitions for Constraint Satisfaction Problems." Constraints 11(2-3) : 115-137 (2006)

[2] Crawford, J., Ginsberg, M., Luks E.M., Roy, A. "Symmetry Breaking Predicates for Search Problems." In proceedings of KR'96, 148-159.

[3] P. Flener, A. M. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, J. Pearson, T. Walsh. : "Breaking Row and Column Symmetries in Matrix Models. " Proceedings of CP'02, pages 462-476, 2002

[4] I. Gent, T. Walsh, and B. Selman CSPLIB, A problem library for constraints http://www.csplib.org

[5] Gent, I.P., and Harvey, W., and Kelsey, T. : "Groups and Constraints : Symmetry Breaking During Search". *In proceedings of CP 2002*, pp. 415-430.

[6] Gent, I.P., Harvey, W., Kelsey, T., Linton, S. : "Generic SBDD Using Computational Group Theory". *In proceedings of CP 2003*, pp. 333-437

[7] ILOG : ILOG Solver 6.3. User Manual. ILOG, S.A., Gentilly, France, Juillet 2006

[8] Y. C. Law and J. H. M. Lee. "Expressing Symmetry Breaking Constraints Using Multiple Viewpoints and Channeling Constraints". *In Proceedings of SymCon 03* (held in conjunction with CP-2003), pages 127-141, 2003.

[9] Y. C. Law and J. H. M. Lee. "Breaking Value Symmetries in Matrix Models using Channeling Constraints". *In Proceedings of the 20th Annual ACM Symposium on Applied Computing (SAC-2005)*, pages 375-380, 2005.

[10] Lustig, I.J., and Puget, J.F. (2001). "Program Does Not Equal Program : Constraint Programming and its Relationship to Mathematical Programming," Interfaces 31(6), 29-53.

[11] Mc Kay, B. : "Practical Graph Isomorphism" Congr. Numer. 30, 45-87, 1981

[12] Dame Ollerenshaw, K. "On most perfect or complete 8x8 pandiagonal magic squares". *In Proceedings Royal Society London*, 407, 259-281, 1986.

[13] Petrie, K., Smith, B.M. : "Symmetry breaking in graceful graphs." In proceedings of CP'03, LNCS 2833, 930-934, Springer Verlag, 2003.

[14] Puget, J.-F. : "On the Satisfiability of Symmetrical Constraint Satisfaction Problems." Proceedings of ISMIS'93 (1993), 350-361.

[15] Puget, J.-F. : "Symmetry Breaking Revisited". Constraints 10(1) : 23-46 (2005)

[16] Puget, J.-F. : "Symmetry Breaking in Alldifferent Problems". Proceedings of SymCon 04, Toronto, Canada, 2004.

[17] Puget, J.-F. : "Automatic Detection of Variable and Value Symmetries" Proceedings of CP 05, Sitges, 2005, 475-489.

[18] Puget, J.-F. : "Breaking All Value Symmetries in Surjection Problems" Proceedings of CP 05, Sitges, 2005, 490-504.

[19] Puget, J.-F. : "An Efficient Way of Breaking Value Symmetries" Proceedings of AAAI 06.

[20] Regin, J.-C. : "Developpement d'Outils Algorithmiques pour l'Intelligence Artificielle. Application la Chimie Organique." PhD thesis, Universite de Montpelier II, 1995 (in French)

[21] Roney-Dougal C.M., Gent, I.P., Kelsey T., Linton S. : "Tractable symmetry breaking using restricted search trees" To appear in proceedings of ECAI'04.

[22] Roy. A., Luks, E. : "The complexity of symmetry-breaking formulas", *Annals of Mathematics and Artificial Intelligence* , 41 (2004), 19-45 (with A. Roy).

[23] Pierre Flener, Justin Pearson, Meinolf Sellmann, Pascal Van Hentenryck : "Static and Dynamic Structural Symmetry Breaking." In proceedings of CP 2006 : 695-699

[24] Seress, A. : *Permutation Group Algorithms* Cambrige University Press, 2003.

[25] Barbara M. Smith. "Sets of Symmetry Breaking Constraints" In Proceedings of SymCon05, the 5th International Workshop on Symmetry in Constraints, 2005

[26] Sorlin, S., Solnon, C. : "A global constraint for graph isomorphism problems" In proceedings of CPAIOR'O4, LNCS, Springer.

[27] Toby Walsh : "General Symmetry Breaking Constraints." In proceedings of CP 2006 : 650-664.

[28] S. Zampelli, Y. Deville, P. Dupont, "Symmetry Breaking in Subgraph Pattern Matching" In proccedings of the Sixth International Workshop on Symmetry in Constraint Satisfaction Problems (SymCon'06), Nantes, France, September 25, 2006. A French version exists as : S. Zampelli, Y. Deville, P. Dupont, "Elimination des symtries pour l'appariement de graphes", JFPC'06, Nmes, France, June 7-9, pp. 357-367, 2006.

Appendix

We are given a CSP $\mathcal{P}$ with n variables v_i subject to an all different constraint among other constraints. Without loss of generality, we can assume that the domains of the variables are subsets of I^d for some d. It is shown in [3] how to transform a CSP$\mathcal{P}$ into a new CSP $\mathcal{P}'$ such that all value symmetries of $\mathcal{P}$ become variable symmetries of $\mathcal{P}'$. Although this was used for the case where any row permutation and any column permutation is a symmetry, this idea can be generalized to more general cases. The idea is to add $n \times d$ additional binary variables x_{ij} (variables with domains equal to $\{0, 1\}$). We also add the following channeling constraints :

$$\forall i \in I^n, j \in I^d,\ (x_{ij} = 1) \leftrightarrow (v_i = j) \tag{10}$$

These constraints state that $x_{ij} = 1$ if and only if $v_i = j$. Adding these new variables does not change the solutions of the CSP. Variable symmetries of $\mathcal{P}$ are equivalent to permutations of the rows of the x_{ij} matrix by (1). Value symmetries of $\mathcal{P}$ are equivalent to permutations of the columns of the same matrix by (2).

Let X be the vector obtained by concatenating the columns of the matrix x_{ij}. The variables x_{ij} are ranked in increasing values of i, then increasing values of j in the vector X.

Any variable symmetry σ of P is now a permutation of the rows of the matrix (x_{ij}). From [2], this variable symmetry is broken by the constraints :

$$X \preceq X^\sigma$$

that is,

$$(x_{11}, x_{12}, \ldots, x_{nk}) \preceq (x_{1^\sigma 1}, x_{i^\sigma 2}, \ldots, x_{n^\sigma k}) \tag{11}$$

Let us compare lexicographically the first k variables in both sides of the constraint. We have the two vectors :

$$(x_{11}, x_{12}, \ldots, x_{1k})$$

$$(x_{1^\sigma 1}, x_{1^\sigma 2}, \ldots, x_{1^\sigma k})$$

Let a be the value assigned to v_1, and b be the value assigned to v_{1^σ} in a given solution. Then, $x_{1a} = 1$ and $x_{1j} = 0$ for $j \neq a$. Similarly, $x_{1^\sigma b} = 1$ and $x_{1^\sigma j} = 0$ for $j \neq b$. Then, the first vector is lexicographically smaller than the second one if and only if $a \leq b$. This is equivalent to the condition $v_1 \leq v_{1^\sigma}$. By repeated applications of a similar argument, once for every row of the x_{ij} matrix, we prove the following result.

Lemma 6. *With the above notations, the constraint (11) is equivalent to :*

$$(v_1, v_2, \ldots, v_n) \preceq (v_{1^\sigma}, v_{2^\sigma}, \ldots, v_{n^\sigma})$$

Let us consider a value symmetry θ for $\mathcal{P}$. Then, θ is a permutation of the columns of the matrix. This symmetry is broken by the constraint :

$$X \preceq X^\theta$$

that is

$$(x_{11}, x_{12}, \ldots, x_{nk}) \preceq (x_{11^\theta}, x_{12^\theta}, \ldots, x_{nk^\theta}) \tag{12}$$

Let us compare lexicographically the first k variables in both sides of the constraint. We have the two vectors :

$$(x_{11}, x_{12}, \ldots, x_{1k})$$

$$(x_{11^\theta}, x_{12^\theta}, \ldots, x_{1k^\theta})$$

Let a be the value assigned to v_1. Then, $x_{1a} = 1$ and $x_{1j} = 0$ for $j \neq a$. Similarly, $x_{1j^\theta} = 1$ if and only if $j = a^{\theta^{-1}}$. Therefore, the following holds if and only if $a \leq a^{\theta^{-1}}$

$$(x_{11}, x_{12}, \ldots, x_{1k}) \preceq (x_{11^\theta}, x_{12^\theta}, \ldots, x_{1k^\theta})$$

This must be true for all possible θ. This is true if and only if a is the minimum of its orbit in G, where G is the group of value symmetries for the CSP :

$$a = min(a^G) \tag{13}$$

Let us now consider the second group of k variables on both sides :

$$(x_{21}, x_{22}, \ldots, x_{2k})$$

$$(x_{21^\theta}, x_{22^\theta}, \ldots, x_{2k^\theta})$$

Let b be the value assigned to v_2. Then, the following holds if and only if $b \leq b^{\theta^{-1}}$:

$$(x_{21}, x_{22}, \ldots, x_{2k}) \preceq (x_{21^\theta}, x_{22^\theta}, \ldots, x_{2k^\theta})$$

Then, let us consider the first $2k$ variables on each side altogether. We have that

$$(x_{11}, x_{12}, \ldots, x_{2k}) \preceq (x_{11^\theta}, x_{12^\theta}, \ldots, x_{2k^\theta})$$

in exactly one of the following two cases. The first case is :

$$(x_{11}, x_{12}, \ldots, x_{1k}) \prec (x_{11^\theta}, x_{12^\theta}, \ldots, x_{1k^\theta})$$

The second case is :

$$(x_{11}, x_{12}, \ldots, x_{1k}) = (x_{11^\theta}, x_{12^\theta}, \ldots, x_{1k^\theta})$$

and

$$(x_{21}, x_{22}, \ldots, x_{2k}) \preceq (x_{21^\theta}, x_{22^\theta}, \ldots, x_{2k^\theta})$$

The first case if equivalent to $a < a^{\theta^{-1}}$. The second case is equivalent to $a = a^{\theta^{-1}}$ and $b \leq b^{\theta^{-1}}$. The condition $a = a^{\theta^{-1}}$ is equivalent to $\theta \in a_G$. When considering all possible θ, we get the following conditions :

$$\forall \theta \in a_G,\ b \leq b^{\theta^{-1}}$$

This means that b must be the minimum of its orbit in a_G when $b \neq a$. Note that if $b = a$, b is also the minimum of its orbit in a_G, because its orbit is then $\{a\}$. We have proved that

$$(x_{11}, x_{12}, \ldots, x_{2k}) \preceq (x_{11^\theta}, x_{12^\theta}, \ldots, x_{2k^\theta})$$

holds if and only if the following holds : a is the minimum of its orbit in G, and b is the minimum of its orbit in a_G.

More generally, let a_i be the value assigned to the variable v_i in a solution to the CSP. By a repeating the above argument with the first i rows of the x_{ij} matrix, we get the following result :

Lemma 7. *With the above notations, a_i is the minimum of its orbit in the group of symmetries that leave $a_1, a_2, \ldots a_{i-1}$ unchanged.*

This is equivalent to the GE-tree method for breaking all value symmetries [21], when the variables and the values are tried in an increasing order during search.

From [2], it is safe to add all possible symmetry breaking constraints (3). In particular, it is safe to state all the constraints (11) and all the constraints (12) together. By lemma 6, the set of constraints (11) is equivalent to all the symmetry breaking constraints for $\mathcal{P}$. By lemma 7, the set of constraints (12) is equivalent to the GE-tree method for breaking value symmetries. We have just proved the following result.

Lemma 8. *Given a CSP, its group of variable symmetries G_1, and its group of value symmetries G_2, then the combination of the GE-tree method for breaking value symmetries with the symmetry breaking constraints (3) computes a set of solutions $\mathcal{S}$ such that :*

$$\forall S \in sol(\mathcal{P}), \exists \sigma \in G_1, \exists \theta \in G_2, \exists S' \in \mathcal{S},\ S^{\sigma\theta} = S'$$

Theorem 4 in section 3 says that the set of all those constraints (3) is equivalent to the constraints (6) when there is an all different constraints on all the variables $\mathcal{V}$. This yields the following result.

Theorem 5. *Given a CSP where the variable are subject to an all different constraint, its group of variable symmetries G_1, and its group of value symmetries G_2, then the combination of the GE-tree method for breaking value symmetries with the symmetry breaking constraints (6) computes a set of solutions $\mathcal{S}$ such that :*

$$\forall S \in sol(\mathcal{P}), \exists \sigma \in G_1, \exists \theta \in G_2, \exists S' \in \mathcal{S},\ S^{\sigma\theta} = S'$$