Tractable symmetry breaking using restricted search trees

Colva M. Roney-Dougal, Ian P. Gent, Tom Kelsey, Steve Linton [1]

Abstract. We present a new conceptual abstraction in symmetry breaking – the GE-tree. The construction and traversal of a GE-tree breaks all symmetries in any constraint satisfaction or similar problem. We give a polynomial-time algorithm for this construction in the case of CSPs with arbitrary value symmetries. We have implemented this technique, and supply experimental evidence of its practical effectiveness.

1 Introduction and motivation

Many search problems are symmetric, and this can cause significant problems for search. Not only do we find the same solution many times over, but worse, we search the same non-solutions time and time again. Without specialised techniques to break this symmetry and to eliminate or at least reduce duplicates, many problems are not amenable to solution as constraint satisfaction problems (CSPs).

Such a significant problem has led to a large body of research. Approaches taken include adding symmetry breaking constraints before search [3, 5], using constraints generated dynamically during search [1, 10], checking for duplicate nodes before visiting them [2, 4, 6], and constructing a specialised tree without symmetry [14]. An important development is the use of techniques from computational group theory to speed the symmetry breaking process [2, 8, 9, 12]. In this article we show how to build specialised trees without symmetry using computational group theory.

In [14], Van Hentenryk, Flener, Pearson and Ågren gave an efficient method for breaking three specific forms of value symmetry in CSPs. This motivated us to develop a new conceptual abstraction, the GE-tree, which can be defined for arbitrary symmetries of a search problem. The algorithms presented by [14] can be seen as efficient ways to construct GE-trees for those specific value symmetries. Using computational group theory techniques, we give a polynomial time algorithm for constructing GE-trees for *any* value symmetry of a CSP. We have implemented this and show its success in practice. Extensive investigations remain to be done, but just from the evidence presented in this article, we suggest that GE-trees are an important new concept.

We have found GE-trees to be an invaluable tool in two main ways. Firstly, they are useful as a conceptual tool: any search tree contains a GE-tree, and it is helpful to analyse the efficacy of symmetry breaking techniques by seeing how nearly they reduce the search space to that of a minimal GE-tree. We have also found that it is useful when discussing these types of problem to separate an analysis of the symmetries of a problem from a discussion of the constraints involved.

Secondly, GE-trees are a useful practical tool. For instance, it has proved possible to construct GE-trees for a great many types of symmetry. In this article we discuss only value symmetry, for reasons of space, but we have also been able to construct GE-trees for certain kinds of variable symmetry, and for some fully general (i.e. mixed variable and value) symmetry. A great deal of work remains to be done in this area.

In the next section we formally define GE-trees. The following two sections contain a description of a fast and memory-efficient technique for constructing a GE-tree for *arbitrary* value symmetry. After this we discuss the interactions between GE-trees and constraints, and compare the construction of GE-trees with other methods of symmetry breaking, namely SBDS [10] and SBDD [4]. We then present some experimental data, before concluding.

2 GE-trees

Definition 1 *A CSP is a set of constraints $\mathcal{C}$ acting on a finite set of variables $\Delta : A_1, A_2, \ldots, A_n$, each of which has finite domain of possible values $D_i := \mathrm{Dom}(A_i) \subseteq \Lambda$. A solution to a CSP is an instantiation of all of the variables in Δ such that all of the constraints in $\mathcal{C}$ are satisfied. Statements of the form $(var = val)$ are literals.*

Definition 2 *Given a CSP L, with a set of constraints $\mathcal{C}$, and a set χ of literals, a symmetry of L is a bijection $f : \chi \to \chi$ such that:*

1. *A (partial or full) assignment A of L satisfies all constraints in $\mathcal{C}$ if and only if $f(A)$ does.*
2. *An assignment A of L is a nogood if and only if $f(A)$ is a nogood.*

We denote nodes in a search tree by calligraphic letters, and elements of the symmetry group by lower case letters. The letter G is reserved for the symmetry group. Elements of the symmetry group act on the right, thus $(A = \alpha)g$ denotes the image of the literal $(A = \alpha)$ under the symmetry g.

We consider trees to consist of nodes which are labelled by variables (except for the leaves, which are unlabelled), and edges labelled by values. We think of this as meaning that the variable is set to that value as one traverses the path from the root of the tree toward the leaves. At a node $\mathcal{N}$, the partial assignment given by reading the labels on the path from the root to $\mathcal{N}$ (ignoring the label at $\mathcal{N}$ itself) is the *state* at $\mathcal{N}$. We often identify nodes with their state, when the meaning is clear. By the *values in* $\mathcal{N}$ we mean the values that occur in literals in the state at $\mathcal{N}$, and similarly for variables. We often speak of a permutation as mapping a node $\mathcal{N}$ to a node $\mathcal{M}$, although strictly speaking the permutation maps the literals in the state at $\mathcal{N}$ to the literals in the state at $\mathcal{M}$.

Before giving a formal definition of a GE-tree, we need some more group-theoretic definitions.

Definition 3 *Let G be a group of symmetries of a CSP. The* stabiliser *of a literal $(X = \alpha)$ is the set of all symmetries in G that map $(X = \alpha)$ to itself. This set is itself a group. The* orbit *of a literal $(X = \alpha)$, denoted $(X = \alpha)^G$, is the set of all literals that can be reached from $(X = \alpha)$ by a symmetry in G. That is*

$$(X = \alpha)^G := \{(X = \alpha)g : g \in G\}.$$

The orbit of a node is defined similarly.

Definition 4 *A* Group Equivalence tree *(GE-tree) for a constraint satisfaction problem with symmetry group G is any search tree satisfying the following:*

1. *No node is isomorphic under G to any other node.*
2. *Given a full assignment $\mathcal{A}$, there is at least one leaf of the tree which lies in $\mathcal{A}^G$.*

Therefore, the nodes of the tree represent equivalence classes of partial assignments under the group, and the action of the group on the tree fixes every node.

We define a GE-tree to be *minimal* if the deletion of any node (and its descendants) will delete at least one full assignment.

3 Construction of GE-trees for value symmetries

In this section we describe how to construct a GE-tree when the symmetries of the problem act on values. In the most general case, a *value symmetry* is any permutation g where if $(A_1 = \alpha)g = (A_2 = \beta)$ then $A_1 = A_2$. Note in particular that this allows the existence of symmetries g acting as follows:

$$(A_1 = \alpha)g = (A_1 = \beta), \quad (A_2 = \alpha)g = (A_2 = \gamma).$$

Our description in this section makes no mention of constraints: see Section 5 for a discussion of how constraints interact with the symmetry breaking.

The first step is to relabel the values such that the domains of the variables are pairwise disjoint. We may now consider the permutations to be acting on the values themselves, rather than on literals. The size of the set upon which G acts is $\sum_{i=1}^{n} |D_i|$.

With the previous example, we might make the values distinct by adding subscripts, to show which variable can be assigned to them. The action of g would then be written

$$\alpha_1 g = \beta_1, \quad \alpha_2 g = \gamma_2.$$

We construct a tree as follows. Suppose that a node $\mathcal{N}$ has state $\bigwedge_{1 \leq i \leq k}(A_i = \alpha_i)$. We compute the subgroup of G that stabilises each of $\{\alpha_i : 1 \leq i \leq k\}$ (the *pointwise stabiliser* of the α_i) and denote this subgroup by $G_{(\alpha_i:1 \leq i \leq k)}$. We select a variable A_{k+1} which is not in $\mathcal{N}$ (it does not matter whether we always choose the same variable at the same depth), and label $\mathcal{N}$ with A_{k+1}. We then compute the orbits $\{O_j : 1 \leq j \leq o_{k+1}\}$ of $G_{(\alpha_i:1 \leq i \leq k)}$ on D_{k+1}. For each such orbit O_j we select a representative β_j. For $1 \leq j \leq o_{k+1}$ we then create an edge from $\mathcal{N}$ labelled with β_j.

Theorem 5 *A tree $\mathbf{T}$ constructed as in the preceding paragraph is a GE-tree.*

Proof: Suppose first that two nodes are isomorphic. This means that there exists a $g \in G$ and nodes $\mathcal{N}$ and $\mathcal{M}$ with $\mathcal{M}g = \mathcal{N}$. We will show that $\mathcal{M} = \mathcal{N}$.

Let $\mathcal{P}$ be the node of greatest depth in the tree that is an ancestor of both $\mathcal{M}$ and $\mathcal{N}$ and let X be its label. If $\mathcal{M} \neq \mathcal{N}$ then the state at $\mathcal{M}$ contains more literals than the state at $\mathcal{P}$, and so for some α, β we must have $(X = \alpha) \in \mathcal{N}$ and $(X = \beta) \in \mathcal{M}$. However, since $\mathcal{M}g = \mathcal{N}$ we have $\alpha g = \beta$. The symmetry g must fix $\mathcal{P}$, since $\mathcal{M}$ and $\mathcal{N}$ agree on the variables in $\mathcal{P}$. So α is in the same orbit as β under the pointwise stabiliser of the values in $\mathcal{P}$, which is a contradiction. Thus $\mathcal{M} = \mathcal{P} = \mathcal{N}$.

Suppose next that $\mathcal{A}$ is a full assignment. We must show that there exists a $g \in G$ such that $\mathcal{A}g$ is a leaf in $\mathbf{T}$. We write $\mathcal{A}$ as

$$\bigwedge_{i=1}^{m}(A_i = \alpha_i).$$

Without loss of generality, A_1 is the label on the root of $\mathbf{T}$. Denote the orbit of α_1 under G by α_1^G. There is a unique $\beta_1 \in \Lambda$ such that β_1 is a label of an edge under the root and $\beta_1 \in \alpha_1^G$. Let $g \in G$ be such that $\alpha_1 g = \beta_1$. Then the first literal of $\mathcal{A}g$ is a node of $\mathbf{T}$.

Suppose that for some $g_k \in G$ the partial assignment $\wedge_{i=1}^{k}(A_i = \alpha_i g_k)$ is a node $\mathcal{N} \in \mathbf{T}$, with label A_{k+1}. For $1 \leq i \leq k$, denote $\alpha_i g_k$ by β_k. We show that there exists a $g_{k+1} \in G$ such that $\wedge_{i=1}^{k+1}(A_i = \alpha_i g_{k+1})$ is a node in $\mathbf{T}$. Denote the orbits of $G_{(\beta_i:1 \leq i \leq k)}$ on the domain of A_{k+1} by $O_1, \ldots, O_s$. There exists a unique j with $\alpha_{k+1} \in O_j$. Let β_{k+1} be the representative of O_j that is a label of a downedge from $\mathcal{N}$, let $h \in G_{(\beta_i:1 \leq i \leq k)}$ satisfy $\alpha_{k+1}h = \beta_{k+1}$, and let $g_{k+1} := g_k h$. Then the first $(k+1)$ literals of $\mathcal{A}g_{k+1}$ are the state of a node in $\mathbf{T}$. *QED*

Proposition 6 *A GE-tree $\mathbf{T}$ constructed as in Theorem 5 is minimal.*

Proof: Suppose that $\mathbf{T}$ is not minimal. Then there is a node $\mathcal{N} \in \mathbf{T}$ such that $\mathbf{T} \setminus \mathcal{N}$ is a GE-tree.

The node $\mathcal{N}$ cannot be the ancestor of any full assignment, as otherwise deleting $\mathcal{N}$ would result in the deletion of a full assignment $\mathcal{A}$. Since no symmetric equivalent of $\mathcal{A}$ lies in $\mathbf{T}$, the tree $\mathbf{T} \setminus \mathcal{N}$ would contain no representative of $\mathcal{A}$ under G, a contradiction.

Suppose without loss of generality that $\mathcal{N}$ itself has no descendants. This implies that every variable has been used in $\mathcal{N}$, and so $\mathcal{N}$ is a full assignment. This is a contradiction. *QED*

3.1 Complexity of this construction

We now consider the complexity of this construction of a GE-tree. This is crucial if we are to compare the use of GE-trees with other methods of symmetry breaking.

In this section we let $d_i := |D_i|$. We examine the time required to construct the edges leading down from level k, and to construct the nodes at level $(k+1)$. Recall the "soft-O" notation for complexity, where we ignore logarithmic terms. Thus $O^{\sim}(N) = O(N \log^c N)$ for some constant c. For all of the group-theoretic complexity results, see [13].

We consider the root to be level zero, and let the root be labelled A_1. In a standard search tree, the time required to construct the edges leading down from the root is $O(d_1)$.

In a GE-tree with symmetry group G, a base and strong generating set for G must be constructed before we construct the tree. We let $N := \sum_{i=1}^{n} d_i$, then there is a deterministic algorithm to do this in time $O^{\sim}(N^4 + tN^2)$, where t is the number of generators of G.

We may assume that $O(t) < O(N)$, since there is an $O(t \log N)$ algorithm to reduce the number of generators of a group to at most $O(N)$. For many practical purposes, $t < 5$.

The time required to compute the number of orbits of G on D_1 is $O(td_1)$. Denote the number of such orbits by o_1. We select a representative of each orbit. This gives a total time at the root of $O\tilde{}(o_1 + td_1)$. Note that $o_1 < d_1$ and will generally be very small relative to d_1, else the problem does not display much symmetry.

We now compare the time requirement of extending from level k to level $k + 1$, where $k > 0$. For each node $\mathcal{N}$ we have a state, given by the path from the root to $\mathcal{N}$. Suppose for the sake of simplicity that there is a fixed variable ordering, so that all nodes at level i are labelled A_{i+1} for $0 \le i \le n - 1$. Each node at depth k is therefore already labelled with A_{k+1}.

In a standard search tree, the time required to construct level $k + 1$ is $O(\prod_{i=1}^{k+1} d_i)$, since at each of the $O(\prod_{i=1}^{k} d_i)$ nodes we must construct d_{k+1} edges.

We write $\mathcal{R}_H^X$ to mean a set of orbit representatives for the action of a permutation group H on a set X. In a GE-tree, the number of nodes at depth k is

$$\sum_{\alpha_1 \in \mathcal{R}_G^{D_1}} \sum_{\alpha_2 \in \mathcal{R}_{G_{\alpha_1}}^{D_2}} \cdots \sum_{\alpha_n \in \mathcal{R}_{G_{(\alpha_1, \dots, \alpha_{n-1})}}^{D_n}} |\mathcal{R}_{G_{(\alpha_1, \dots, \alpha_n)}}^{D_{n+1}}|.$$

For each node $\mathcal{N}$ we start by computing the pointwise stabiliser $G_{\mathcal{N}}$ of the values in $\mathcal{N}$, in time $O\tilde{}(N^4 + tN^2)$. We then compute the orbits of $G_{\mathcal{N}}$ on D_{k+1}, which requires time $O(t_{\mathcal{N}} d_{k+1})$, where $t_{\mathcal{N}}$ is the number of generators of $G_{\mathcal{N}}$. Since $d_{k+1} \le N$ and $O\tilde{}(t_{\mathcal{N}}) = O\tilde{}(N)$, this is $O\tilde{}(N^2)$ for each node $\mathcal{N}$. Denote the number of orbits of $G_{\mathcal{N}}$ on D_{k+1} by $o_{\mathcal{N},k+1}$: note that $o_{\mathcal{N},k+1} \le d_{k+1} \le N$. We finish by selecting a representative of each orbit. Thus the total cost at each node is is certainly no greater than $O\tilde{}(N^4)$.

As a consequence of this discussion we have the following important theorem.

Theorem 7 *Symmetry-breaking for a CSP with only value symmetries is tractable.*

4 Global value symmetries

One common form for a problem with value symmetries is for the symmetries to act globally. We say that the group consists of *global value symmetries* if, for all variables $X, Y \in \Delta$, values $\alpha, \beta \in \Lambda$, and symmetries $g \in G$, $(X = \alpha)g = (X = \beta)$ implies that $(Y = \alpha)g = (Y = \beta)$.

To determine that a group G acts in this way, it suffices to check that each of the generators of G acts in this way. Suppose from now on that this is so. As in Section 3, we consider the symmetries to be acting on the values directly, rather than on literals. However, we do *not* make distinct copies of each value for each variable, but instead allow the group to act on the union of the domains of each variable. Note that this means that the group is acting on fewer points than it was in the previous section: the size of the union of the domains rather than the sum of the domain sizes.

The tree is then constructed exactly as in the previous section: at each node $\mathcal{N}$ we compute the pointwise stabiliser $G_{\mathcal{N}}$ of the values in $\mathcal{N}$. We then compute the orbits of $G_{\mathcal{N}}$ on the domain of the label of $\mathcal{N}$, and construct one edge for each orbit, labelled with an orbit representative.

Proposition 8 *This construction produces minimal GE-trees.*

At this point, our tree can be seen to be a generalisation of the Compact Variable Decomposition Tree introduced in [14]. Suppose that G acts via global value symmetries, and is the symmetric group on these values. This means that given any pair P_1, P_2 of n-tuples of distinct values, there is an element of G that can simultaneously map each coordinate of P_1 to the corresponding coordinate of P_2. We denote the symmetric group on n points by $\mathrm{Sym}(n)$.

The pointwise stabiliser in the symmetric group of any set of points acts as the symmetric group on the remaining points. Thus at a node $\mathcal{N}$ we have one edge for each value which we have used so far (as these have orbits of size 1 under the pointwise stabiliser) and then a single edge labelled with a representative of the remaining values. This is precisely the search space described in [14, §3].

Similarly, one may show that the GE-trees which we construct contain a unique representative of each node that is explored in the other two cases considered in [14]. Consider, for instance, the direct product case (which they term "piecewise interchangeable CSPs"). Here $G = \mathrm{Sym}(a) \times \mathrm{Sym}(b)$ for some a and b, and the $a + b$ values are partitioned into two subsets Λ_1 and Λ_2 of size a and b that cannot be interchanged. The pointwise stabiliser in G of any set $\mathcal{S}$ of points has the following orbits on the full set of values. For each $\alpha \in \mathcal{S}$ there is an orbit $\{\alpha\}$ of length 1. There is an orbit consisting of all points of $\Lambda_1 \backslash \mathcal{S}$, and an orbit which contains all points of $\Lambda_2 \backslash \mathcal{S}$. Thus if a node $\mathcal{N}$ is labelled with X then it has the following downedges: one for each value in $\mathcal{N}$ that lies in the domain of X, one labelled with an element of Λ_1 that lies in the domain of X and is not in $\mathcal{N}$ and one labelled with an element of Λ_2 that lies in the domain of X and is not in $\mathcal{N}$.

Figure 1. A GE-tree for $G = \mathrm{Sym}(a) \times \mathrm{Sym}(b)$
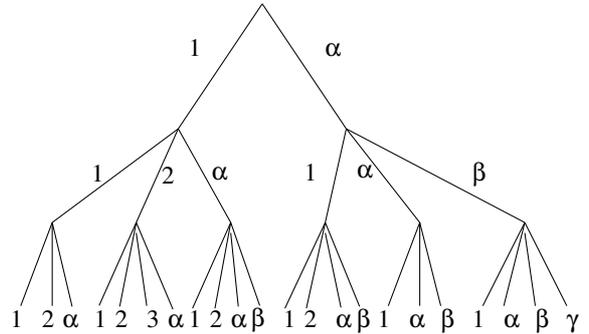
The first three levels of the resulting GE-tree are shown in Figure 1. We use natural numbers for elements of Λ_1 and greek letters for elements of Λ_2. Since the choice of variable at each node is completely arbitrary, we do not label the nodes. We have labelled the final level of edges at their ends purely for clarity.

The partial assignments given by the tree in Figure 1 are precisely the options considered in [14, §3]. A similar, but longer argument shows that the same is true for the final case that they consider, that of wreath product action.

4.1 Complexity of this construction

The complexity of this construction is similar to that for the preceding section. However, the group is now acting on the union of the domains for each variable, rather than a set whose size is the sum of the sizes of the domains for each variable. Therefore we now have

$N := |\cup_{i=1}^{n} D_i|$. The formula for the number of nodes at level k is the same as in Section 3.1, as is the formula for the time cost at each node (but with the new meaning for N).

5 GE-trees and search

Whilst we have described GE-trees as static objects, they will normally be constructed dynamically during search and hence the issue arises of how the GE-tree construction interacts with constraints.

We consider the situation where a CSP has arbitrary value symmetry and no variable symmetry. Recall that the edges below a node $\mathcal{N}$ labelled X are orbit representatives for the pointwise stabiliser of the partial assignment at $\mathcal{N}$ on the domain of X.

Lemma 9 *Let $\alpha \in \mathrm{Dom}(X)$ be an orbit representative under $G_\mathcal{N}$. If there exists $\beta \in \alpha^{G_\mathcal{N}}$ such that $\beta \notin \mathrm{Dom}(X)$ then there are no solutions extending $\mathcal{N} \wedge (X = \alpha)$.*

Proof: Since $\beta \notin \mathrm{Dom}(X)$, the value β must have been pruned from the domain of X at some point higher in the search tree. This implies that for some partial assignment $\mathcal{M} \subseteq \mathcal{N}$, the partial assignment $\mathcal{M} \wedge (X = \beta)$ causes failure. Thus any partial assignment in the orbit under G of $\mathcal{M} \wedge (X = \beta)$ causes failure. But there exists a $g \in G_\mathcal{N}$ such that $(X = \beta)g = (X = \alpha)$ and $\mathcal{M}g = \mathcal{M} \subseteq \mathcal{N}$, so $\mathcal{N} \wedge (X = \alpha)$ will cause failure. *QED*

Thus during dynamic construction of a GE-tree for value symmetries we do not construct these branches, leading to even greater time savings.

We next compare the search trees constructed by SBDD and SBDS.

Lemma 10 *SBDS, using all symmetries of the group, constructs a GE-tree.*

Proof: It is well-known that SBDS will find exactly one of each equivalence class of solutions, if the whole group is used.

We therefore need only show that under SBDS no two nodes are isomorphic. Suppose nodes $\mathcal{M}$ and $\mathcal{N}$ are isomorphic. This means that there is a $g \in G$ such that $\mathcal{M}g = \mathcal{N}$. This in turn implies that $\mathcal{M} = \mathcal{N}$, or else an SBDS constraint would have prevented us from constructing both nodes. *QED*

Lemma 11 *SBDD constructs a GE-tree, if the dominance check is applied at every node.*

Proof: It is well-known that SBDD will find exactly one of each equivalence class of solutions, provided that dominance checks are applied to the leaf nodes.

We therefore need only show that under SBDD no two nodes are isomorphic. Suppose nodes $\mathcal{M}$ and $\mathcal{N}$ are isomorphic, and that $\mathcal{M}$ is to the left of $\mathcal{N}$ in the tree. But then $\mathcal{M}$ dominates $\mathcal{N}$ unless $\mathcal{M} = \mathcal{N}$. *QED*

A more subtle problem is to determine when SBDD and SBDS construct the *same* tree for a problem with value symmetry as the GE-tree constructed in Section 3, when one includes the branch pruning of Lemma 9. We suppose that the values have been ordered, and that SBDD and SBDS always try the smallest value first, and that the smallest value in an orbit under a symmetry group is always chosen as an orbit representative.

Remark 12 *Suppose that a CSP has only value symmetries, that there are no constraints, and that no propagation other than forward checking is carried out. Under the above value ordering heuristic, SBDS will construct exactly the same tree as our GE-tree construction given in Section 3, provided that all symmetries are used and that checks are carried out at every node, and that both techniques select the same next variable on being given a partial assignment.*

To see this, let $\mathcal{N}$ be a partial assignment and X the next variable to consider. We wish to show that exactly the same images are considered for X under SBDS as in the GE-tree construction of Section 3. It is clear that if $\mathcal{N} = \emptyset$ then SBDS and our GE-tree construction will choose the same values for X, in the same order. SBDS will consider the possible values from smallest to largest through the domain of X. It will construct a branch if $\mathcal{N} \wedge (X = \alpha)$ is not symmetrically equivalent to any previous node. Therefore, α is the smallest member of its orbit under $G_\mathcal{N}$. All such smallest members are constructed. For suppose that β is the minimal member of an orbit of $G_\mathcal{N}$ on $\mathrm{Dom}(X)$, and that SBDS does not construct a branch for $(X = \beta)$. Then $\mathcal{N} \wedge (X = \beta)$ is symmetrically equivalent to a node $\mathcal{M}$. But this means that $\mathcal{N}$ must be symmetrically equivalent to some other node, since we only have value symmetry and β is the first element of its orbit that we have considered. But by induction, $\mathcal{N}$ cannot be mapped to any other node.

A second consideration is how the construction of a GE-tree for a subgroup of the symmetry group would interact with other symmetry-breaking techniques, principally SBDS and SBDD.

Theorem 13 *Let $\mathbf{T}$ be a GE-tree for the subgroup H of value symmetries of the symmetry group G of a CSP, constructed as in Section 3. If SBDS or SBDD is performed whilst searching $\mathbf{T}$ then exactly one of each equivalence class of solutions will be found.*

Proof: We provide a proof for SBDD; the SBDS case is similar. Let $\mathcal{A}$ be a solution. Then $\mathbf{T}$ will contain at least one terminal node which is a symmetric equivalent of $\mathcal{A}$. Denote these by $\mathcal{A}_1 := \mathcal{A}g_1$, $\mathcal{A}_2 = \mathcal{A}g_2, \ldots, \mathcal{A}_t = \mathcal{A}g_t$, where $g_i \in G$.

We must show that exactly one of these equivalent states will be found when searching $\mathbf{T}$ using SBDD. No more than one state is found, for if $\mathcal{A}_i$ is the first such then since $\mathcal{A}_i.g_i^{-1}g_j = \mathcal{A}_j$, the node $\mathcal{A}_i$ dominates all other $\mathcal{A}_j$.

If the first symmetric variant of a partial assigment is always the one (if any) to be accepted as a node of a GE-tree then it is clear that a representative of each equivalence class of solutions is found, as SBDD will only reject states that contain symmetric equivalents of nogoods.

A problem might arise if the first of an equivalence class of solutions is not the chosen representative for the GE-tree. For then it might it might happen that a node $\mathcal{N}$ is posted as a nogood even though the subtree under it rooted at $\mathcal{M}$ has not been explored (because $\mathcal{M}g$ is the chosen representative). We need to show that $\mathcal{N}$ does not dominate (and hence rule out) the orbit representative $\mathcal{M}g$. This follows because g is a value symmetry, so both $\mathcal{M}$ and $\mathcal{M}g$ are below $\mathcal{N}$. Therefore $\mathcal{M}g$ will always be explored *before* $\mathcal{N}$ is posted as a nogood, and the resulting tree will always contain a representative of each equivalence class of solutions. *QED*

Remark 14 *If a GE-tree is constructed for the group of value symmetries of a CSP, then by Theorem 13 it is safe to use SBDD or SBDS to break all other symmetries. This combined approach has lower complexity than using SBDD or SBDS alone, as both are exponential whilst our GE-tree construction is low-degree polynomial.*

6 Experiments

In this section we present results from an implementation of value-symmetry CSP solving by GE-tree construction. We model and constrain the problem in the ECL^iPS^e [15] CSP system. We search by attributing to each variable its current partial assignment and its current domain. This information is passed to the GAP [7] computational algebra system. GAP returns a domain containing only symmetrically distinct values. Backtrack search proceeds (in ECL^iPS^e) on this (hopefully much smaller) domain.

Our initial experiments involved colouring non-symmetric graphs. Here the value symmetry is $\text{Sym}(n)$, where n is the number of colours. We compared the GE-tree method to SBDD using GAP–ECL^iPS^e [9]. We found that solution times were markedly reduced. For example, finding all 7-colourings of a 12-vertex graph took 50.10 cpu seconds (28.35 GAP, 21.75 ECL^iPS^e) using SBDD. Solving the same problem by GE-tree construction took 8.87 cpu seconds (6.12 GAP, 2.75 ECL^iPS^e). These results are expected: GAP–ECL^iPS^e SBDD performs a potentially exponential search for a dominating group element at each variable-value choice, whereas GE-tree construction has low-degree polynomial complexity (as described in Section 3.1). This motivated experimentation with more complex value symmetry.

A *most perfect magic square* [11] of size $n \times n$ (where $n \equiv 0 \mod 4$) has entries $1 \ldots n^2$ such that (i) each row, column and diagonal (including wrap around) has sum $(n^3 + n)/2$, (ii) each 2×2 block (including wrap around) has sum $2(n^2 + 1)$, and (iii) any pair distant by $n/2$ along a diagonal (including wrap around) sum to $n^2 + 1$. Any solution is one of $2^{n+1}((n/2)!)^2$ symmetric equivalents [11]. Many of these symmetries do not preserve constraints, and are therefore difficult to handle without computational group theory.

The natural model for this problem has cells as variables and entries as values. In addition to the constraints given above, we add the implied constraints that any two entries $n/2$ along any row (resp. column) with the left entry in an even column (resp. row) have the same sum. However, we cannot use the polynomial time algorithm because we have only variable symmetries. But the values of the n^2 variables are a permutation of $1 \ldots n^2$. So we search using a different model in which the variables are entries and the value is the cell the entry goes in. Now all symmetries are value symmetries and we construct a GE-tree to break them completely. However, the constraints are most naturally expressed on the original model, so we simply add channeling constraints between the dual sets of variables. Thus we have the simple expression of the constraints of the natural model, combined with the complete polynomial time symmetry breaking using the dual model.

Method	n	sols	GAP	Eclipse	Σcpu
SBDD	4	3	0.3	0.3	0.6
	8	10	5.4	125.4	130.8
	12	42	2748.2	12519.8	15268.0
GE-tree	4	3	0.2	0.1	0.3
	8	10	0.7	90.0	90.7
	12	42	29.1	10901.8	10930.9

Table 1. Most Perfect Magic Squares

Our results are summarised in Table 1, where the number of solutions are correct for SBDD and GE-tree, and the timings are cpu time in seconds using a 2.4GHz Pentium 4 processor. Since there are 294,912 symmetric equivalents of each solution for $n = 8$, breaking no symmetry is not a sensible option. Using SBDD on the value symmetries works well, but the GAP dominance checks are still potentially exponential. Solution by GE-tree construction is the most effective method, primarily due to the tractable group-theoretic questions being posed during construction. This suggests a practical and effective method for symmetry breaking in CSPs. If – either by formulation and/or the use of channeling constraints – all the symmetries are value symmetries, then GE-tree construction provides a polynomial time method for complete symmetry breaking. This is a win whenever the reduced cost of symmetry breaking outweighs the cost of a sub-optimal formulation and/or additional constraints.

7 Conclusion

In this article we have introduced a new conceptual abstraction, the GE-tree. This is a search tree containing a unique representative of each class of full assignments, which also has the property that no node is isomorphic to any other node. We have given a polynomial-time algorithm to construct this tree in the case of arbitrary value symmetries, and have shown that for certain kinds of symmetry it can be viewed as a generalisation of the tractable symmetry-breaking techniques discussed in [14]. We have also presented experimental data which allows us to conclude that this is an efficient practical approach to symmetry breaking.

Future areas of interest include further experimentation on value-symmetry CSPs, analysis of GE-tree construction for CSPs with mixed variable-value symmetry and for non-CSP problems, and further investigation into the relationship between GE-tree construction and existing symmetry breaking techniques.

REFERENCES

[1] R. Backofen and S. Will, 'Excluding symmetries in constraint-based search', in *Proc. CP-99*, pp. 73–87. Springer, (1999).

[2] C.A. Brown, L. Finkelstein, and P.W. Purdom, Jr., 'Backtrack searching in the presence of symmetry', in *Proc. AAECC-6*, ed., T. Mora, pp. 99–110. Springer-Verlag, (1988).

[3] J. Crawford, M. Ginsberg, E. Luks, and A. Roy, 'Symmetry-breaking predicates for search problems', in *Proc. KR'96*, pp. 149–159, (November 1996).

[4] Torsten Fahle, Stefan Schamberger, and Meinolf Sellmann, 'Symmetry breaking', in *Proc. CP 2001*, ed., T. Walsh, pp. 93–107, (2001).

[5] Pierre Flener, Alan M. Frisch, Brahim Hnich, Zeynap Kızıltan, Ian Miguel, Justin Pearson, and Toby Walsh, 'Breaking row and column symmetries in matrix models', in *Proc. CP 2002*, ed., Pascal Van Hentenryck, pp. 462–476. Springer-Verlag, (2002).

[6] Filippo Focacci and Michaela Milano, 'Global cut framework for removing symmetries', in *Proc. CP 2001*, ed., T. Walsh, pp. 77–92, (2001).

[7] The GAP Group, *GAP – Groups, Algorithms, and Programming, Version 4.3*, 2003. (http://www.gap-system.org).

[8] Ian P. Gent, Warwick Harvey, and Tom Kelsey, 'Groups and constraints: Symmetry breaking during search', in *Proc. CP 2002*, ed., Pascal Van Hentenryck, pp. 415–430. Springer-Verlag, (2002).

[9] I.P. Gent, W. Harvey, T. Kelsey, and S.A. Linton, 'Generic SBDD using computational group theory', in *Proc. CP 2003*, ed., Francesca Rossi, pp. 333–347. Springer-Verlag, (2003).

[10] I.P. Gent and B.M. Smith, 'Symmetry breaking in constraint programming', in *Proc. ECAI 2000*, ed., W. Horn, pp. 599–603. IOS Press, (2000).

[11] Dame Kathleen Ollerenshaw, 'On most perfect or complete 8x8 pandiagonal magic squares', *Proc. Royal Soc. London*, **407**, 259–281, (1986).

[12] Jean-François Puget, 'Symmetry breaking revisited', in *Proc. CP 2002*, ed., Pascal Van Hentenryck, pp. 446–461. Springer-Verlag, (2002).

[13] Akos Seress, *Permutation group algorithms*, number 152 in Cambridge tracts in mathematics, Cambridge University Press, 2002.

[14] P. van Hentenryck, P. Flener, J. Pearson, and M. Agren, 'Tractable symmetry breaking for CSPs with interchangeable values', in *Proc. IJCAI'03*, (2003).

[15] M. G. Wallace, S. Novello, and J. Schimpf, 'ECLiPSe : A platform for constraint logic programming', *ICL Systems Journal*, **12**(1), 159–200, (May 1997).