Groupoids and Conditional Symmetry

No Author Given

No Institute Given

Abstract. We introduce groupoids – generalisations of groups in which not all pairs of elements may be multiplied, or, equivalently, categories in which all morphisms are invertible – as the appropriate algebraic structures for dealing with conditional symmetries in Constraint Satisfaction Problems (CSPs). We formally define the Full Conditional Symmetry Groupoid associated with any CSP, giving bounds for the number of elements that this groupoid can contain. We describe conditions under which a Conditional Symmetry sub-Groupoid forms a group, and, for this case, present an algorithm for breaking all conditional symmetries that arise at a search node. Our algorithm is polynomial-time when there is a corresponding algorithm for the type of group involved. We prove that our algorithm is both sound and complete – neither gaining nor losing solutions. We report on an implementation of the algorithm.

1 Introduction

The notion of symmetry in CSPs is well understood, and there are several methods for identifying and breaking symmetries [1]. Group theory has been a central topic in mathematics since the early nineteenth century. It is the mathematical abstraction of symmetry, and is a vital tool across many areas of mathematics, physics, chemistry and computer science. Computational group theoretic techniques have been successfully used to break symmetries in CSPs [2–6].

Conditional symmetries in CSPs are, loosely speaking, parts of the problem that become interchangeable when some condition is satisfied. Typically the condition will be that a subset of the variables have been set to a particular choice of values. Definitions of conditional symmetry, together with initial approaches for identifying and breaking them, are given in [7–10]. A key problem is that the set of conditional symmetries of a CSP does not, in general, form a group. This means that we can not safely use group theoretic results and algorithms. A sound but incomplete algorithm (i.e. one which never misses an existing solution, but which may report more than one symmetrically equivalent solution) for breaking conditional symmetries was given in [11]. This algorithm involved two groups at each node in search. The first group represented all non-conditional symmetries (if any), the second represented the symmetries that had arisen as a result of the assignments made in search so far (if any). By performing Symmetry Breaking by Dominance Detection (SBDD) [12, 13] using the first group, then – if not dominated – on the second group, many conditional symmetries were broken in a restricted class of problems. This motivates the research question: is there a

suitable mathematical abstraction of conditional symmetry that can be used to identify, classify and break such symmeteries in an arbitrary CSP?

In this paper we describe groupoids as the class of mathematical objects that are the appropriate abstraction for conditional symmetries. Using groupoids we can fully describe, enumerate and analyse conditional symmetries for any CSP. Groupoids are generalisations of groups, in which not all elements can be composed. A basic introduction to groupoids is given in [14], with more detailed discussions and results appearing in [15, 16]. We can define, for any CSP, the Full Conditional Symmetry Groupoid, containing elements that capture both the symmetry and the condition under which it arises. This approach allows us to classify conditional symmetries in terms of the sub-groupoid(s) in which they are contained. Moreover, we can identify conditional symmetries that have properties that allow us to develop effective symmetry breaking techniques.

We next provide a formal exposition of groupoids and conditional symmetries. This is followed by some instructive examples (Section 3); a description of a sound and complete conditional symmetry breaking algorithm (Section 4) which, for a restricted class of CSPs, has polynomial time complexity at each node in search; our computational experience with an implementation of the algorithm (Section 5); and some concluding remarks.

2 Groupoids and conditional symmetries

There are two main "nontechnical" definitions of a conditional symmetry. The first is a symmetry that is only applicable to extensions of a particular partial assignment. The second, more general, definition is a symmetry that is only applicable when a constraint is satisfied. We will (eventually) define conditional symmetries to be maps from literals to literals that are applicable to sets of literals – variable-value pairs – provided that such sets satisfy preconditions. To do this, we need to prepare a theoretical framework.

2.1 Notation from CSPs

We will use the following notation throughout: the set of variables will denoted with a V, individual variables $v_1, \ldots, v_n$ and sometimes w. For simplicity, we will assume that there is a single domain D for all variables (since constraints can be added to give different domains for chosen variables). A *literal* of a CSP is a pair (w, d) where $w \in V$ and $d \in D$.

Definition 1. *A* partial assignment f *is a set of literals that contains at most one literal for each variable. A partial assignment* g *is an* extension *of* f *if* $f \subseteq g$

We say that a partial assignment f is *complete* if $|f| = |V|$. That is, all variables are assigned values by f. A *solution* to a CSP is a complete partial assignment that violates no constraint.

2.2 Introduction to Groupoids

Groupoids are generalisations of groups. The reason that we have to leave groups behind when talking about conditional symmetries is that every element of a symmetry group can be applied to *all* partial assignments, and hence one cannot capture the concept of "condition".

Definition 2. *A* groupoid *is a set G, and a basis set B, together with a partial operation, composition, and two maps s and t from G to B that together satisfy the following:*

1. *composition, gh, of two elements, g and h in G is defined only when $t(g) = s(h)$ — this is what is meant by saying that the composition is partial;*
2. *if the products gh and hk are defined then $g(hk) = (gh)k$ is defined;*
3. *for every g in G there are left and right identity elements λ_g and ρ_g s.t. $\lambda_g g = g = g\rho_g$;*
4. *Each element has an inverse g^{-1} s.t. $gg^{-1} = \lambda_g$ and $g^{-1}g = \rho_g$.*

It is useful to think of maps s and t as representing source and target respectively. Formally, a *groupoid* is a category where every morphism has an inverse. This is in contrast to a group, which is a category with one object where every morphism has an inverse.

A group is a groupoid where the basis set B contains a single element, so that all pairs of elements can be composed in any order. It can help to think of groupoids as being just like groups except that not all elements can be composed.

Often groupoids will be presented as collections of triples,

$$(s(g), g, t(g))$$

then composition of two elements g and h can be written as:

$$(s(g), g, t(g))(s(h), h, t(h)) = (s(g), gh, t(h))$$

provided that $t(g) = s(h)$. Note that this definition follows the usual algebraic convention of "acting from the right" so that gh means "do g then do h".

A *subgroupoid* of a group G is a subset of the elements of G that itself forms a groupoid under the same partial operation. The base set of a subgroupoid is the set of all sources and targets that occur for elements of the subgroupoid.

2.3 Groupoids and conditional symmetries

We now turn to formalising conditional symmetries as groupoid elements. The key problem in providing the definitions is that for composition to be defined, the target of one element has to be *equal* to the source of the next. This makes it impossible to represent sources and targets as the simple condition given by the constraint. Instead, we have a separate groupoid element for every partial assignment that satisfies a condition. We will develop the theory and show that this leads to well-defined groupoids and useful theoretical results.

We will denote the image of an object α under a map ϕ by α^ϕ. A *literal bijection* of a CSP is a bijection ϕ from the set of literals of the CSP to itself. The following definition begins the process of capturing the notion of a condition.

Definition 3. *A literal bijection is a* symmetry with respect to *a, where a is a subset of literals, if in its induced action on sets of literals, whenever f is a solution that contains a then f^ϕ is a solution and whenever f is a non-solution that contains a then f^ϕ is a non-solution.*

It is clear that a *symmetry* of a CSP is a symmetry with respect to all sets of literals, so that the minimal sets of literals a with respect to which a bijection is a symmetry are the conditions under which it is a symmetry. Thus, our work is orthogonal to whether symmetries are syntactic or semantic (see for example [17–19]).

Before defining conditional symmetry groupoids, we need precise descriptions of conditions and symmetries arising from conditions. A *condition* is a predicate on literals. Any such predicate can be described by listing the sets of literals upon which it holds. If a symmetry in a CSP is present only when some condition holds, we will describe this situation by adding one generator to the conditional symmetry groupoid for each set of literals which satisfy the condition.

Definition 4. *We define the* full conditional symmetry groupoid *G of a CSP P to be the set of all triples as follows:*

$$G = \{(g, \pi, f) : g, f \text{ sets of literals}, \pi \text{ a symmetry with respect to } g, g^\pi = f\}.$$

The product (in this order) of two groupoid elements $(g, \pi, f), (h, \sigma, k)$ is defined only if $h = f$ in which case the product is $(g, \pi\sigma, k)$ and $g^{\pi\sigma}$ will equal k.

We work throughout with subgroupoids of the full conditional symmetry groupoid of the CSP. Note that if $(g, \pi, f) \in G$ and h is a set of literals that contains g, then $(h, \pi, h^\pi) \in G$, as π is a symmetry with respect to g only if it is a symmetry with respect to all extensions of g. We say that (h, π, h^π) is an *extension* of (g, π, f), and that the conditional symmetry groupoid is *closed under extensions*.

We call the first entry of each triple its *precondition* and the last its *postcondition*. Note that one can deduce the postcondition of a groupoid element from its precondition and permutation, however we will often write down the postcondition for clarity.

Lemma 1. *With these definitions, the full conditional symmetry groupoid G of a CSP is a groupoid.*

Proof. The set of elements of G is clear, its base set is the power set of the set of literals. We must show that composition, where defined, is associative and that each element has left and right identities and inverses. Let $(g_1, \pi_1, f_1), (g_2, \pi_2, f_2)$ and $(g_3, \pi_3, f_3) \in G$, and suppose that $g_2 = f_1$ and $g_3 = f_2$. Then

$$((g_1, \pi_1, f_1)(g_2, \pi_2, f_2))(g_3, \pi_3, f_3) = (g_1, \pi_1\pi_2, g_1^{\pi_1\pi_2})(g_3, \pi_3, f_3).$$

Now, $g_2 = f_1 = g_1^{\pi_1}$, so $f_2 = g_2^{\pi_2} = g_1^{\pi_1\pi_2}$. Hence $g_3 = f_2 = g_1^{\pi_1\pi_2}$ and the product of the two groupoid elements in the previous displayed equation is defined, and is equal to $(g_1, \pi_1\pi_2\pi_3, g_1^{\pi_1\pi_2\pi_3})$. Conversely

$$\begin{aligned}(g_1, \pi_1, f_1)((g_2, \pi_2, f_2)(g_3, \pi_3, f_3)) &= (g_1, \pi_1, f_1)(g_2, \pi_2\pi_3, g_2^{\pi_2\pi_3}) \\ &= (g_1, \pi_1\pi_2\pi_3, g_1^{\pi_1\pi_2\pi_3}),\end{aligned}$$

as required.

The right identity of (g, π, f) is $(f, 1, f)$, the left identity is $(g, 1, g)$, and the inverse is (f, π^{-1}, g), where by 1 we mean the identity mapping on the set of all literals.

Lemma 2. *The full conditional symmetry groupoid of a CSP has a well-defined partial action on the set of all sets of literals, which maps (non-)solutions to (non-)solutions.*

Proof. The action of an element (g, π, f) on a set h of literals is as follows. If $g \not\subseteq h$ then the condition of (g, π, f) is not satisfied, and so the action is undefined. If $g \subseteq h$ then $h^{(g,\pi,f)} := h^\pi$.

To show that this is an action, we note that $h^{(g,\pi_1,f)(f,\pi_2,k)}$ is defined if and only if $g \subseteq h$ are sets of literals in which case we have

$$h^{(g,\pi_1,f)(f,\pi_2,k)} = h^{(g,\pi_1\pi_2,k)} = h^{\pi_1\pi_2} = (h^{(g,\pi_1,f)})^{(f,\pi_2,k)},$$

and that $h^{(g,1,f)} = h$ whenever $g \subseteq h$.

Whenever h is a full assignment and $h^{(g,\pi,f)}$ is defined, then, since π is a symmetry with respect to g and $g \subseteq h$, h^π is a (non-)solution exactly when h is a (non-)solution.

Next we show that our notions of conditional symmetry strictly generalise the standard notions of unconditional symmetry.

Lemma 3. *The full conditional symmetry groupoid of a CSP P contains the group of all symmetries of P.*

Proof. Elements of the symmetry group of the CSP are of the form $(\emptyset, \pi, \emptyset)$.

In general, however, the full conditional symmetry groupoid of a CSP is far too large for practial computation, and we must restrict the situation somewhat. Examples of sizes of conditional symmetry sub-groupoid of a particular CSP are given in Section 3. We now provide bounds for the number of elements in any full conditional symmetry groupoid:

Lemma 4. *The full conditional symmetry groupoid of a CSP P with $|V| = n$ and $|D| = d$ has order at least 2^{dn} and at most $2^{dn}(dn)!$.*

Proof. A precondition can be any subset of literals and the corresponding permutation can be any permutation of the full set of literals. The identity permutation is a symmetry with respect to any set of literals.

Of course, in practise we cannot identify *all* conditional symmetries of a CSP before solving it, so we identify as many non-identity conditional symmetries as possible, and then form the resulting groupoid.

Definition 5. *We say that a collection of conditional symmetry groupoid elements* $\{(g_i, \pi_i, f_i) \ : \ 1 \leq i \leq k\}$ generate *a conditional symmetry groupoid* G *if* G *is the smallest subgroupoid of the full conditional symmetry groupoid that contains all of the elements and is closed under extensions.*

The reason why any conditional symmetry groupoid is defined to be closed under extensions is that it must always be possible to consider conditions that are stronger than those initially given: if (v_1, α) implies some symmetry, then so does $\{(v_1, \alpha), (v_2, \beta)\}$. Later, when we discuss computing with conditional symmetry groupoids, we will describe a technique which avoids the creation of these additional elements.

Lemma 5. *The conditional symmetry groupoid* G *is generated by elements from* $A := \{(g_i, \pi_i, f_i) \ : \ 1 \leq i \leq k\}$ *if and only if each element of* G *is a product of elements that are extensions of elements of* A *and their inverses.*

Proof. We first show that the set of all elements of the conditional symmetry groupoid that are products of extensions of elements of A forms a groupoid and is closed under extensions. It is clear that it is closed under the partial product and the associativity follows from the associativity of the generalised conditional symmetry groupoid. If $(x_0, \sigma, x_k) = (x_0, \sigma_1, x_1)(x_1, \sigma_2, x_2) \cdots (x_k, \sigma_{k+1}, x_{k+1})$ is a product of extensions of elements of A, then we note that $(x, 1, x)$ is an extension of a left identity of an element of A, and similarly for $(y, 1, y)$, and that $(x_i, \sigma_i^{-1}, x_{i-1})$ and $(x_{i+1}, \sigma_{i+1}^{-1}, x_i)$ are extensions of right and left inverses of elements of A. Thus this set forms a groupoid. If (x, σ, y) is an extension of (x_0, σ, x_k) then it is clear that it is a product of elements of the form $(\overline{x_i}, \sigma_{i+1}, \overline{x_{i+1}})$, where $\overline{x_i}$ is a partial assignment that extends x_i.

Conversely, if (x, σ, y) is a product of extensions of elements of A then it must be contained in any groupoid that contains A, and so is an element of the groupoid generated by A. Thus the result follows.

3 Examples

Throughout the rest of this article we will consider two families of CSPs, both of which have conditional symmetries, to provide concrete examples of our somewhat technical definitions.

The first example is that of graceful windmill graphs. A labelling f of the vertices of a graph with e edges is *graceful* if f assigns each vertex a unique label from $\{0, 1, \ldots, e\}$ such that when each edge xy is labelled with $|f(x) - f(y)|$, the edge labels are all different: that is the edge labels take all values in $\{1, \ldots, e\}$. The *windmill* graphs are listed in Gallian's survey [20] as $C_n^{(t)}$. They consist of t copies of a cycle with n nodes, joined at a common central vertex. For $n = 3$

these graphs are graceful when $t \equiv 0, 1 \bmod 4$; examples of a solution and a non-solution are given in Figure 1, for $t = 4$.

The symmetries of the CSP include: (i) swapping the labels of the nodes other than the centre node of each blade of the windmill (flipping the blades); (ii) permuting the blades; (iii) replacing every value x with $e - x$ (hi-lo). It is shown in [11] that the central node cannot have label > 1 and $< e - 1$.

There are many conditional symmetries. Suppose that the cycles are triangles and the central node is labelled 0. In any triangle with (noncentral) labels a and b with $a < b$, we can replace a with $b - a$. If the central node is labelled 1 then in a triangle with node labels a and b we can replace label a with $b - a + 1$. Similar conditional symmetries apply to windmills with central node label $e - 1$ or e.

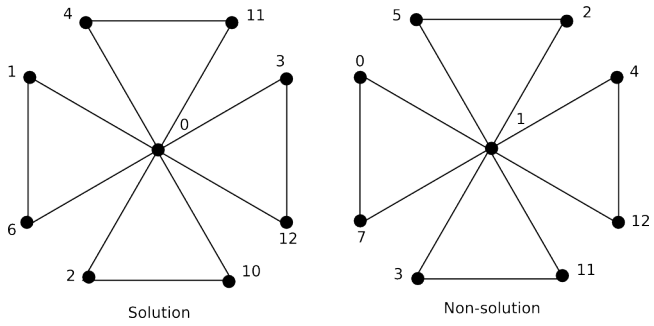

Fig. 1. The windmill graph $C_3^{(4)}$

There is a second type of conditional symmetry. In a labeling L with central node labelled 0 and any blade labelled 0, 1 and e, we can replace labels $0, 1, 2, 3, \ldots, e-2, e-1, e$ with $1, 0, 3, 4, \ldots, e-1, 2, e$. That is, label e is fixed, labels 0 and 1 are swapped, and the remaining labels are cycled around one place. We obtain a labelling L' with central node labelled 1 and a blade still labelled 0, 1 and e. To see that this is correct, observe that a satisfying complete labelling extending L cannot label any node with $e - 1$. Thus, except strictly on the blade labelled $0, 1, e$ all labels increase by 1 and so all label differences are unchanged. The same argument applies to unsatisfying full labellings which do not use the label $e - 1$. Any full labelling which uses $e - 1$ is unsatisfying and is transformed into an unsatisfying labelling using 2, and with two label difference equal to 1.

The inverse to this conditional symmetry requires the central node to be labelled 1 and one blade to be labelled $0, 1, e$. It replaces labels $0, 1, 2, 3, \ldots, e - 2, e - 1, e$ with $1, 0, e - 1, 2, \ldots, e - 2, e$. Similar conditional symmetries exist for graphs with central node $e - 1$ and for graphs with central node e and a suitably labelled triangle.

Note that *any* satisyfing labelled in which the central node is labelled 1 must have a blade labelled $0, 1, e$ (since edge label e must occur). It is therefore tempting, as was done in [11] to believe that the condition that such a blade is present is redundant. This is not the case, however, since although all solutions with

central node labelled 1 would be mapped to solutions, there are non-solutions with central node labelled 1 which would be mapped to solutions. This is illustrated in Figure 1: the non-solution on the right is mapped to the solution on the left, if we only consider preconditions involving the central node.

Some simple computations give a flavour of the size of these groupoids. For the two-blade windmill $C_3^{(2)}$, the conditional symmetry groupoid has 64 272 elements *not including* extensions. That is, it has 64 272 elements, none of which is an extension of another element of the groupoid. These are all the elements that need to be considered in symmetry breaking, since any two partial assignments equivalent under the groupoid are equivalent under one of these elements. In fact, the full groupoid, including all extensions, has order 748 819 (actually even this count only includes conditional symmetries whose preconditions are compatible with the all-different constraint on the vertex labels). To get a more detailed picture we can grade the conditional symmetries according to the size of their preconditions, which we call their *rank*. Among the non-extensions, we have 16 unconditional symmetries of rank 0; 896 conditional symmetries of rank 2; 3712 of rank 3; 41216 of rank 4 and 18432 of rank 5. Including extensions, we get 16 of rank 0, 560 of rank 1; 8624 of rank 2; 67760 of rank 3; 286422 of rank 4 and 385027 of rank 5.

Our second example consists of a family of graph-colouring problems. There is a central fully-connected graph with n nodes, each forming a node of an outer K_n. There are $n+2$ colours. One central node can be labelled with either the nth or $n+1$st colour; another central node has domain colours 1 and $n+2$; all other nodes can be labelled with colours from 1 through n. An example for $n = 3$ with colours *red, green, blue, orange* and *purple* is given in Figure 2. The unconditional value symmetries are that $n-1$ colours on the remaining nodes of the outer subgraphs involving the two special nodes can be fully permuted. Conditional symmetries arise when nodes of the central subgraph are labelled. If colour $n+1$ and/or colour $n+2$ is used, ($8 = orange$ and/or $9 = purple$ in Figure 2), then we can fully permute all colours in the rest of the relevant outer subgraph(s). Whenever a non-special central node (node 7 in Figure 2) is instantiated, we can fully permute the labels of the relevant outer nodes. We use this class of problems to illustrate our conditional symmetry breaking algorithm described in the following section. We also use the $n = 3$ example to illustrate conditional symmetry groupoids and actions.

Suppose that we have partial assignments $f = \{8 = orange, 3 = red, 4 = blue\}$ and $g = \{8 = orange, 3 = blue, 4 = red\}$. Let π be the permutation that swaps *red* and *blue* on nodes 3 and 4, and leaves everything else unchanged. Then (f, π, g) is an element of the the full conditional symmetry groupoid of the CSP, since the conditions of Definition 4 are satisfied. If h is any partial assignment that contains f, then $h^{(f,\pi,g)} = h^{\pi}$. For this example, the elements of the full conditional symmetry groupoid form a group, which allows us to define an analogue of SBDD. Moreover, the group acts only on values which allows us to check dominance in time polynomial in the number of literals.

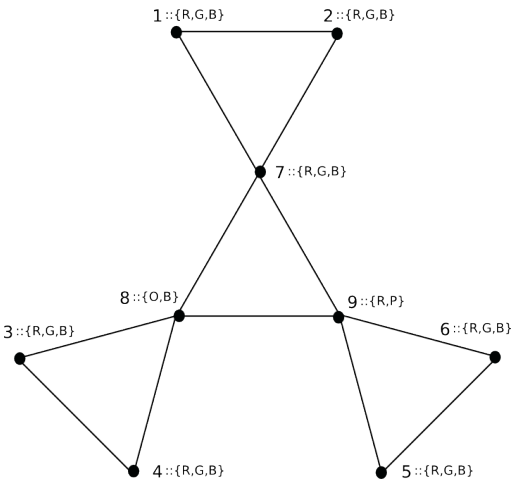

Fig. 2. Example $n = 3$

4 Symmetry breaking with groupoids

4.1 Symmetry Breaking by Dominance Detection

Symmetry breaking by dominance detection for groups of symmetries uses the following principle: given the current search node g then for each previously found nogood f, and each symmetry π test the inclusion $f^\pi \subseteq g$. This can be done either by some supplied procedure, as in SBDD, or by adding new constraints that rule out all images under the symmetry group of extensions of f.

Lemma 6. *If $f \subseteq g$ are partial assignments, and (h, π, k) is an element of the full conditional symmetry groupoid, then $f^{(h,\pi,k)} \subseteq g^{(h,\pi,k)}$.*

Proof. If h is not a subset of f, then neither map is defined. Otherwise, $f^{(h,\pi,k)} = f^\pi = (f \setminus h)^\pi \cup k$ and $g^{(h,\pi,k)} = g^\pi = (g \setminus h)^\pi \cup k$. Since $(f \setminus h)^\pi \subseteq (g \setminus h)^\pi$, we have $f^\pi \subseteq g^\pi$ as required.

When implementing SBDD efficiently for groups or groupoids, the fact that groupoid actions are well-behaved with respect to extensions of partial assignments means that only failures at the top of subtrees need be stored during depth first search.

To check the dominance in SBDD in the group case we search for an element π of the symmetry group such that g is an extension of f^π. The situation with groupoids is a bit more complicated, as the action is only partial and so the following two cases have to be checked:

- Does there exist a groupoid element (h, γ, k) such that $f \subseteq h$ and $g \subseteq f^\gamma$?
- Otherwise we must search for some extension, f', of f and a groupoid element (h, γ, k) such that $f' \subseteq h$ and $g \subseteq f'^\gamma$.

Note that in case (ii) it is only worth considering extensions of f that assign values to no more variables than g, and one should only consider extensions that enable some new conditional symmetry to be used.

To avoid this two-case analysis, we *reverse* the normal process of SBDD and instead look for conditional symmetries that map the current partial assignment to a partial assignment that is an extension of the previous nogood.

Definition 6. *A partial assignment g is* dominated *by a nogood f with respect to a conditional symmetry groupoid G if there exists an element $(h, \gamma, k) \in G$ with $h \subseteq g$ and $f \subseteq g^\gamma$.*

The reason why this simplification is possible is that groupoids *are* closed under inversion, and so if a map exists in one direction we can always consider its inverse.

Unfortunately, searching for dominance under an arbitrary conditional symmetry groupoid can be extremely complicated, as unlike the case of groups we do not have notions of a base and strong generating set which give a normal form for group elements in terms of the generators. Developing analagous notions for groupoids will be the subjct of future research.

4.2 Partial assignments and local groups

The following property of groupoids is well-known, but we re-prove it here as it is extremely useful.

Lemma 7. *Let G be a conditional symmetry groupoid of a CSP P and let a be a fixed partial assignment of P. The set of all elements of G of the form (a, π, a) form a group.*

Proof. Let $(a, \pi_1, a), (a, \pi_2, a) \in G$. The product $(a, \pi_1\pi_2, a)$ is defined and is of the correct form. Associativity holds because within this set of elements the product of two elements is always defined. The element $(a, 1, a) \in G$ and is the identity of this subgroup. The inverse $(a, \pi_1^{-1}, a) \in G$ is of the correct form.

Definition 7. *Let P be a CSP with conditional symmetry groupoid G generated by $X := \{(a_i, \pi_{ij}, b_i) : 1 \leq i \leq m, j \in \mathcal{J}_i\}$. Let A be a partial assignment, and let*

$$C := \bigcup_{a_i \subseteq A} a_i.$$

Then the local group at A with respect to X, *denoted $\mathcal{L}_X(A)$ is the group consisting of the permutations σ of all elements of G of the form (C, σ, C).*

It follows from Lemma 7 that the group at A is always a group, if nonempty. Since there is always the identity unconditional symmetry, the group at A is always nonempty, although it may be the trivial group $(C, 1, C)$.

Lemma 8. *Checking for dominance under the $\mathcal{L}_X(A)$ is sound.*

Proof. Since all elements of $\mathcal{L}_X(A)$ are symmetries with respect to a subset of A, they will all map (non-)solutions extending A to (non-)solutions.

We now consider a special case where checking for dominance under the group at A is also complete.

Theorem 1. *Let G be a conditional symmetry groupoid of a CSP P, generated by a set $X = \{(a_i, \pi_{ij}, a_i) \ : 1 \leq i \leq m, j \in \mathcal{J}_i\}$, where each a_i is a partial assignment. Assume that for $1 \leq i, k \leq m$ we have $a_i^{\pi_{kj}} = a_i$ for all $j \in \mathcal{J}_k$. Then checking for dominance under $\mathcal{L}_X(A)$ is complete as well as sound.*

Proof. Let $H := \mathcal{L}_X(A)$.We show that if $(g, \sigma, h) \in G$ and $g \subseteq A$ then $\sigma \in H$. It is clear that if σ is a product of π_{ij} for $a_i \subseteq A$ then $\sigma \in H$. We wish to show that if σ is a product including at least one π_{kj} for some $a_k \not\subseteq H$ then (g, σ, h) does not act on A. It is clear that no extension of (a_k, π_{kj}, a_k) can act on g as $a_k \not\subseteq A$, and hence that one cannot postmultiply any extension of (a_k, π_{kj}, a_k) to make it act on A. We show that one cannot premultiply any extension of (a_k, π_{kj}, a_k) by an element of G to make it act on A. Since $a_k^{\pi_{il}} = a_k$ for all l, if (x, σ, y) can premultiply an extension of (a_k, π_{kj}, a_k) then $a_k \subseteq y$ and so $a_k \subseteq x$ and hence $x \not\subseteq A$ and we are done.

Using this theorem, we can identify some special cases where testing dominance is possible in polynomial time.

Theorem 2. *Let G be a conditional symmetry groupoid of a CSP P, generated by a set $\{(a_i, \pi_{ij}, a_i) \ : 1 \leq i \leq m, j \in \mathcal{J}_i\}$ where each a_i is a partial assignment. Assume that for $1 \leq i, k \leq m$, whenever $a_i \cup a_k$ is a partial assignment then $a_i^{\pi_{kj}} = a_i$ for all $j \in \mathcal{J}_k$. If the actions of π_{ij} on the set of extensions of a_i all belong to a class of symmetries for which there exists a polynomial time algorithm to determine group dominance, then testing conditional symmetry dominance under the groupoid G can be done in polynomial-time at each node.*

Proof. By Theorem 1 it is both sound and complete to test each partial assignment A for dominance by previous nogoods using only the local group at A with respect to X. The local group at A with respect to X can be constructed in polynomial-time, by examining which conditions of which generators are subsets of A and then generating a group with the corresponding permutations. Hence if there exists a polynomial-time algorithm for testing whether A is dominated by a nogood by an element of a group of symmetries then it can now be applied.

Corollary 1. *Let G be a conditional symmetry groupoid of a CSP P, generated by a set $\{(a_i, \pi_{ij}, a_i) \ : 1 \leq i \leq m, j \in \mathcal{J}_i\}$ where each a_i is a partial assignment. Assume that for $1 \leq i, k \leq m$, whenever $a_i \cup a_k$ is a partial assignment then $a_i^{\pi_{kj}} = a_i$ for all $j \in \mathcal{J}_k$. If the actions of π_{ij} on the set of extensions of a_i are value symmetries then testing conditional symmetry dominance is polynomial-time.*

Proof. This is immediate from the preceding theorem and [6].

We present a polynomial-time algorithm which takes as input a set of conditional symmetry groupoid generators for a CSP and determines whether the conditions of Corollary 1 apply and if so tests for groupoid dominance.
PREPROCESSING For each generator (a_i, π_{ij}, b_i) do

(i) If $a_i \neq b_i$ then return "inapplicable".
(ii) If π_{ij} is not a value symmetry on all literals in a_i, and on all literals involving variables not in a_i then return "inapplicable".
(ii) For each (a_k, π_{kl}, b_k) with $k \neq i$ such that $a_i \cup a_k$ is a partial assignment do
- If $a_i^{\pi_{kl}} \neq a_k$ then return "inapplicable".

AT NODE A

1. Make a set L of the generators of G such that $a_i \subseteq A$.
2. Construct the local group $\mathcal{L}_X(A)$, which is $\langle \sigma : (a, \sigma, a) \in L \rangle$.
3. For each nogood B do
   (a) If B contains any literals with different variables from those in A, consider the next nogood.
   (b) Otherwise, by repeatedly computing point stabilisers check whether there exists a $\sigma \in \mathcal{L}_X(A)$ such that for all $(w, \alpha) \in A$ with $(w, \beta) \in B$ for some β we have $(w, \alpha)^\sigma = (w, \beta)$.
   (c) If such a σ exists, return "dominated".
4. Return "not dominated".

As in SBDD, we can safely backtrack when dominated, and continue to the next search node otherwise. Our approach is not restricted to value symmetries. We can obtain polynomial-time algorithms whenever a polynomial-time algorithm exists for symmetry breaking in the unconditional case. For example, a conditional symmetry breaking version of Puget's algorithm for all-different variable constraints [21] is possible.

The above algorithm exploits the case where the condition is fixed by the conditional symmetry, i.e. pre-conditions and postconditions are the same, and the condition is also fixed by the global symmetries. We have identified various other families of conditional symmetry of interest where efficient algorithms may be obtainable. These are (i) the conditions are fixed by the conditional symmetries but not by the global symmetries; (ii) the conditions may not be fixed as sets of literals, but variables in the condition are fixed by the conditional symmetry; (iii) condition is moved to a completely disjoint condition by the conditional symmetry, and (iv) onditional symmetries commute with global symmetries.

5 Implementation

We have implemented the algorithm described in Section 4, using the constraint solver Eclipse [22] to model CSPs and search for solutions, and using the GAP computational algebra package [23] to check for dominance. We ran the resulting code on problems from the graph-colouring class described in Section 3, since for this class of problems the full conditional symmetry subgroup forms a group. For

simplicity, we considered only conditional symmetries involving the two special central nodes, so the conditions are: one of the nodes is orange and the other is not purple; one is purple and the other is not orange; one is purple and the other is orange. If neither is orange or purple, then no additional symmetries arise. We also ignore variable symmetries, i.e. symmetries of the graph itself. Our results, set out in Table 1, show that (i) large numbers of solutions are returned when no symmetry breaking is done, (ii) exactly one member of each equivalence class of solutions under the global symmetry is returned when SBDD is used, and (iii) exactly one one member of each conditional symmetry class is returned whenever our algorithm is used. There is a high computational cost involved for both SBDD and conditional SBDD. The latter is more expensive because the groups are larger, and we have to reset both the symmetry group and our set of nogoods whenever a condition arises (which, for this class of problems is 3 out of every 4 partial assignments). However, the resulting pruning of the search tree leads to the correct number of symmetrically inequivalent solutions being returned.

Table 1. No symmetry breaking *vs* SBDD *vs* conditional SBDD.

	No SB				SBDD				cond. SBDD			
n	◇	□	△	◯	◇	□	△	◯	◇	□	△	◯
3	320	0	0.01	0.01	80	0.3	0.09	0.39	11	0.39	0.05	0.44
4	313,632	0	5.7	5.7	8,712	41	13	54	505	621	6	627

◇ Solutions □ Gap Time (sec) △ Eclipse time (sec) ◯ Total (sec)

6 Conclusions and Further Work

We have shown that careful use of the theory of groupoids allows us fully to capture the notion of conditional symmetry in CSPs. Our results show that groupoids play as fundamental a role in the study of conditional symmetries in CSPs as groups do in the study of unconditional symmetries.

Previous studies have identified the concept of conditional symmetry and proposed sound methods for breaking it [7–11]. Our work makes two significant advances over this literature. First, we have identified the algebraic structure of conditional symmetries. With this we can study the whole set of conditional symmetries of a problem, rather than just a small subset given to us by a programmer, and analyse sub- and super-groupoids. We have shown the enormous numbers of conditional symmetries and complicated structure that arises when we generate them all. For example, in Section 3 we alluded to mistakes in the literature which arose from only considering conditional symmetries that map solutions to solutions. The second major contribution is the provision of definitions and algorithms for conditional symmetry breaking. Previous methods

which rely on posting constraints for all conditional symmetries do not take into account the vast number of constraints involved. For example, in [7] a variant of SBDS is proposed which, in either of our examples in Section 3, would involve posting an infeasibly large number of constraints at every node, moreover without being able to remove them on backtracking. We have defined a notion of dominance, allowing us to give an analogue of SBDD for conditional symmetries.

We have also shown that it is possible to identify useful conditional symmetry sub-groupoids that are small enough to permit effective conditional symmetries breaking. We currently see no practical method for breaking full conditional symmetries in general. Therefore we propose that research should focus on special cases where it may be easier to break the symmetries than in the general case. We identified some such cases in Section 4.

Using groupoids, we can now identify, describe and enumerate conditional symmetries, and have an algebraic structure – the full conditional symmetry groupoid – that can be analysed for possible symmetry breaking algorithms. For example, analysis of the windmill graph problems (described in Section 3) leads us to believe that an effective heuristic for conditional symmetry breaking is to avoid dealing with the (large number of) high rank conditional symmetries that have little effect on the search tree. This, however, represents future work in which the theory of conditional symmetry groupoids will be of fundamental importance.

References

1. Ian P. Gent, Jean-Francois Puget, and Karen Petrie. Symmetry in constraint programming. In Francesca Rossi, Peter van Beek, and Toby Walsh, editors, *Handbook of Constraint Programming*, pages 329–376. Elsevier, 2006.
2. Jean-Francois Puget. Symmetry breaking using stabilizers. In Rossi [26], pages 585–599.
3. Ian P. Gent, Warwick Harvey, and Tom Kelsey. Groups and constraints: Symmetry Breaking During Search. In Pascal Van Hentenryck, editor, *CP*, volume 2470 of *Lecture Notes in Computer Science*, pages 415–430. Springer, 2002.
4. Ian P. Gent, Warwick Harvey, Tom Kelsey, and Steve Linton. Generic SBDD using computational group theory. In Rossi [26], pages 333–347.
5. Tom Kelsey, Steve Linton, and Colva M. Roney-Dougal. New developments in symmetry breaking in search using computational group theory. In Bruno Buchberger and John A. Campbell, editors, *AISC*, volume 3249 of *Lecture Notes in Computer Science*, pages 199–210. Springer, 2004.
6. Colva M. Roney-Dougal, Ian P. Gent, Tom Kelsey, and Steve Linton. Tractable symmetry breaking using restricted search trees. In Ramon López de Mántaras and Lorenza Saitta, editors, *ECAI*, pages 211–215. IOS Press, 2004.
7. Ian P. Gent, Iain McDonald, and Barbara M. Smith. Conditional symmetry in the all-interval series problem. In Barbara Smith, Ian P. Gent, and Warwick Harvey, editors, *Proceedings of the Third International Worksop on Symmetry in Constraint Satisfaction Problems, a workshop of CP'2003*, 2003.
8. Ian P. Gent, Iain McDonald, Ian Miguel, and Barbara M. Smith. Approaches to conditional symmetry breaking. In *Proceedings of the satellite workshop of CP 2004, Symmetry in Constraints (SymCon'04)*, Toronto, September 2004.

9. Yuanlin Zhang and Eugene C. Freuder. Conditional interchangeability and substitutability. In *Proceedings of the satellite workshop of CP 2004, Symmetry in Constraints (SymCon'04)*, Toronto, September 2004.
10. Toby Walsh. General symmetry breaking constraints. In Frédéric Benhamou, editor, *CP*, volume 4204 of *Lecture Notes in Computer Science*, pages 650–664. Springer, 2006.
11. Ian P. Gent, Tom Kelsey, Steve Linton, Iain McDonald, Ian Miguel, and Barbara M. Smith. Conditional symmetry breaking. In van Beek [24], pages 256–270.
12. Torsten Fahle, Stefan Schamberger, and Meinolf Sellmann. Symmetry breaking. In Walsh [25], pages 93–107.
13. Filippo Focacci and Michela Milano. Global cut framework for removing symmetries. In Walsh [25], pages 77–92.
14. R. Brown. From groups to groupoids: A brief survey. *Bulletins of the London Mathematical Society*, 19:113–134, 1987.
15. A. Weinstein. Groupoids: Unifying internal and external symmetries. *Notices Amer. Math. Soc.*, 43:744–752, 1996.
16. N.D. Gilbert. Actions and expansions of ordered groupoids. *Journal of Pure and Applied Algebra*, 198:175–195, 2005.
17. Belaid Benhamou. Study of symmetry in constraint satisfaction problems. In Alan Borning, editor, *PPCP'94: Second International Workshop on Principles and Practice of Constraint Programming*, Orcas Island, Seattle, USA, 1994.
18. David A. Cohen, Peter Jeavons, Christopher Jefferson, Karen E. Petrie, and Barbara M. Smith. Symmetry definitions for constraint satisfaction problems. In van Beek [24], pages 17–31.
19. Berthe Y. Choueiry and Guevara Noubir. On the computation of local interchangeability in discrete constraint satisfaction problems. In *Proceedings of AAAI'98*, pages 326–333. American Association for Artificial Intelligence, 1998.
20. J.A. Gallian. Graph labeling. *The Electronic Journal of Combinatorics; Dynamic Surveys (DS6)*, 2003.
21. Jean-Francois Puget. Breaking symmetries in all different problems. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI*, pages 272–277. Professional Book Center, 2005.
22. M. G. Wallace, S. Novello, and J. Schimpf. ECLiPSe : A platform for constraint logic programming. *ICL Systems Journal*, 12(1):159–200, May 1997.
23. The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.4*, 2006. (`http://www.gap-system.org`).
24. Peter van Beek, editor. *Principles and Practice of Constraint Programming - CP 2005, 11th International Conference, CP 2005, Sitges, Spain, Proceedings*, volume 3709 of *Lecture Notes in Computer Science*. Springer, 2005.
25. Toby Walsh, editor. *Principles and Practice of Constraint Programming - CP 2001, 7th International Conference, CP 2001, Paphos, Cyprus, Proceedings*, volume 2239 of *Lecture Notes in Computer Science*. Springer, 2001.
26. Francesca Rossi, editor. *Principles and Practice of Constraint Programming - CP 2003, 9th International Conference, CP 2003, Kinsale, Ireland, Proceedings*, volume 2833 of *Lecture Notes in Computer Science*. Springer, 2003.