Tractable Symmetry Breaking for CSPs with Interchangeable Values

P. Van Hentenryck
Department of Computer Science
Brown University, Box 1910
Providence, RI 02912

P. Flener, J. Pearson, and M. Ågren
Department of Information Technology
Uppsala University, Box 337
S-751 05 Uppsala, Sweden

Abstract

Symmetry breaking in CSPs has attracted considerable attention in recent years. Various general schemes have been proposed to eliminate symmetries during search. In general, these schemes may take exponential space or time to eliminate all symmetries. This paper studies classes of CSPs for which symmetry breaking is tractable. It identifies several CSP classes which feature various forms of value interchangeability and shows that symmetry breaking can be performed in constant time and space during search using dedicated search procedures. Experimental results also show the benefits of symmetry breaking on these CSPs, which encompass many practical applications.

1 Introduction

Symmetry breaking for constraint satisfaction problems has attracted considerable attention in recent years (See the last CP proceedings, the CP'01 and CP'02 workshops on symmetries, and [Crawford *et al.*, 1996; Freuder, 1991; Backofen and Will, 1999; Puget, 1993] for some less recent papers). Indeed, many applications naturally exhibit symmetries and symmetry breaking may drastically improve performance (e.g., [Barnier and Brisset, 2002; Meseguer and Torras, 2001; Puget, 2002]). An important contribution in this area has been the development of various general schemes for symmetry breaking in CSPs (e.g., SBDS [Gent and Smith, 2000] and SBDD [Fahle *et al.*, 2001; Focacci and Milano, 2001]). Unfortunately, these schemes, in general, may require exponential resources to break all symmetries. Indeed, some schemes require exponential space to store all the nogoods generated through symmetries, while others may take exponential time to discover whether a partial assignment is symmetric to one of the existing nogoods. As a consequence, practical applications often place limits on how many nogoods can be stored and/or which symmetries to break.

This paper approaches symmetry breaking from a different, orthogonal, standpoint. *Its goal is to identify classes of CSPs that are practically relevant and for which symmetry breaking is tractable, i.e., symmetry breaking is polynomial in time and space*. It identifies several such classes, which encompass many practical applications. *These CSPs feature various forms of value interchangeability and the paper shows that symmetry breaking can be performed in constant time and space during search using dedicated search procedures*. The paper also reports preliminary experimental results indicating that symmetry breaking on these CSPs brings significant computational benefits. Finally, the paper introduces the new notions of existential and abstract nogoods, which were used to derive the results for the various CSP classes. We believe that these notions are helpful to derive many other classes of tractable symmetries. In particular, they also allowed us to derive classes of tractable *variable* symmetries, which we cannot include here for space reasons. As such, this paper should be viewed only as a first step in this fascinating area.

It is also useful to contrast our approach with the research avenue pioneered by [Freuder, 1991] on value interchangeability. Freuder also introduced various forms of value interchangeability. However, his goal was to *discover* symmetries inside CSPs and to reformulate them by preprocessing to remove symmetries. Unfortunately, discovering symmetries in CSPs is not tractable for many interesting classes of CSPs. *Our research, in contrast, assumes that modellers are aware of the symmetries in their applications*. It focuses on how to exploit this knowledge *during search* to eliminate symmetries efficiently. Consider, for instance, the scene-shooting problem featured in [Van Hentenryck, 2002]. This problem aims at producing a movie (or a series) at minimal cost by deciding when to shoot scenes. Each scene involves a number of actors and at most 5 scenes a day can be filmed. All actors of a scene must be present on the day the scene is shot. The actors have fees representing the amount to be paid per day they spend in the studio. The goal of the application is to minimize the production costs. An optimal solution to this problem can be modelled as an assignment of scenes to days which minimizes the production costs. It should be apparent that the exact days assigned to the scenes have no importance in this application and are fully interchangeable. What is important is how the scenes are clustered together. Our approach does not aim at discovering this fact; rather it focuses on how to exploit it to eliminate the symmetries it induces.

The rest of the paper is organized as follows. After some preliminaries, Sections 3, 4, and 5 study three classes of CSPs for which symmetry breaking is tractable. For space reasons, only the first class is presented in full detail with proofs. Section 6 briefly reports some experimental results. Section 7 concludes the paper.

2 Preliminaries

This section defines the main concepts used in this paper. The definitions, although they capture the informal meaning of CSPs, are non-standard and simplify the definitions and proofs considerably. The basic idea is that the set of constraints is abstracted by a Boolean function which holds if all the constraints are satisfied (since we are not interested in the constraint structure). Solutions are also represented as functions (assignments) from variables to their sets of values.

Definition 2.1. A *CSP* is a triplet $\langle V, D, C\rangle$, where V denotes the set of variables, D denotes the set of possible values for these variables, and $C : (V \to D) \to \mathit{Bool}$ is a constraint that specifies which assignments of values to the variables are solutions. A *solution* to a CSP $\mathcal{P} = \langle V, D, C\rangle$ is a function $\sigma : V \to D$ such that $C(\sigma) = \mathit{true}$. The set of solutions to a CSP $\mathcal{P}$ is denoted by $\mathit{Sol}(\mathcal{P})$.

Algorithms to solve CSPs manipulate partial assignments. It is often important to reason about which variables are already assigned (the domain of the partial assignment) and the set of values they are assigned to (the image of the assignment). Note that domains represent sets of variables in this paper (not values as is traditional), since we use functions to model (partial) assignments.

Definition 2.2. Let $\mathcal{P} = \langle V, D, C\rangle$ be a CSP. A *partial assignment* for $\mathcal{P}$ is a function $\alpha : W \subseteq V \to D$. The *domain* of α, denoted by $\mathit{dom}(\alpha)$, is W. The *image* of α, denoted by $\mathit{image}(\alpha)$, is the set $\{\alpha(v) \mid v \in \mathit{dom}(\alpha)\}$. For each value $d \in \mathit{image}(\alpha)$, we use $\alpha^{-1}(d)$ to denote the set $\{v \mid v \in \mathit{dom}(\alpha) \wedge \alpha(v) = d\}$.

In this paper, we often denote a partial assignment α by a conjunction of equations

$$v_{i_1} = \alpha(v_{i_1}) \wedge \cdots \wedge v_{i_k} = \alpha(v_{i_k})$$

where $\mathit{dom}(\alpha) = \{v_{i_1}, \ldots, v_{i_k}\}$. For instance, the partial assignment $v_1 = 1 \wedge v_2 = 2 \wedge v_3 = 3$ represents the function whose domain is $\{v_1, v_2, v_3\}$ and which assigns the value i to v_i. We also denote the empty assignment by ϵ. The next four definitions define nogoods formally and when a (partial) assignment violates a nogood.

Definition 2.3. Let $\mathcal{P}$ be a CSP and α be a partial assignment for $\mathcal{P}$. A partial assignment θ is an *extension* of α for $\mathcal{P}$ if $\mathit{dom}(\alpha) \subseteq \mathit{dom}(\theta)$ and $\forall v \in \mathit{dom}(\alpha) : \theta(v) = \alpha(v)$.

Definition 2.4. Let $\mathcal{P} = \langle V, D, C\rangle$ be a CSP and α be a partial assignment for $\mathcal{P}$. A *completion* of α for $\mathcal{P}$ is an extension $\alpha' : V \to D$ of α for $\mathcal{P}$. The set of all completions of α for $\mathcal{P}$ is denoted by $\mathit{Comp}(\alpha, \mathcal{P})$.

Definition 2.5. Let $\mathcal{P}$ be a CSP. A *nogood* α for $\mathcal{P}$ is a partial assignment for $\mathcal{P}$ such that $\mathit{Comp}(\alpha, \mathcal{P}) \cap \mathit{Sol}(\mathcal{P}) = \emptyset$.

Definition 2.6. Let $\mathcal{P}$ be a CSP, α be a nogood for $\mathcal{P}$, and θ be a partial assignment for $\mathcal{P}$. θ *violates* nogood α for $\mathcal{P}$ if θ is an extension of α for $\mathcal{P}$.

Proposition 2.7. *Let $\mathcal{P}$ be a CSP. If θ violates nogood α for $\mathcal{P}$, then $\mathit{Comp}(\theta, \mathcal{P}) \cap \mathit{Sol}(\mathcal{P}) = \emptyset$.*

This last proposition shows that nogoods can be lifted from the children to their parent.

Proposition 2.8. *Let $\mathcal{P} = \langle V, D, C\rangle$ be a CSP and $D = \{d_1, \ldots, d_m\}$. Let α be a partial assignment for $\mathcal{P}$ with $\mathit{dom}(\alpha) = \{v_{i_1}, \ldots, v_{i_k}\}$ and let every $\alpha_i = \alpha \wedge v_{i_{k+1}} = d_i$ $(1 \leq i \leq m)$ be a nogood for $\mathcal{P}$. Then α is a nogood for $\mathcal{P}$.*

3 Value-Interchangeable CSPs

There are many applications in resource allocation and scheduling where the exact values taken by the variables are not important. What is significant is which variables take the *same* values or, in other terms, how the variables are clustered. This notion of symmetry is what [Freuder, 1991] calls full interchangeability for all variables and all values. The prototypical example is graph-coloring. Another, more complex, example is the scene allocation application mentioned in the introduction. This section shows that symmetry breaking for these problems is tractable and takes constant time and space. We first define value-interchangeable CSPs.

Definition 3.1. Let $\mathcal{P} = \langle V, D, C\rangle$ be a CSP. $\mathcal{P}$ is *value-interchangeable* if, for each solution $\sigma \in \mathit{Sol}(\mathcal{P})$ and each bijection $b : D \to D$, the function $b \circ \sigma \in \mathit{Sol}(\mathcal{P})$.

In the following, we use ICSP as an abbreviation for value-interchangeable CSP. The following theorem is a fundamental characterization of nogoods for ICSPs.

Theorem 3.2. *Let $\mathcal{P} = \langle V, D, C\rangle$ be an ICSP, let α be a nogood for $\mathcal{P}$, and let $b : D \to D$ be a bijection. Then $b \circ \alpha$ is a nogood.*

The closure of a nogood α for an ICSP captures all the "symmetric" nogoods that can be obtained from α.

Definition 3.3. Let $\mathcal{P} = \langle V, D, C\rangle$ be an ICSP and α be a nogood for $\mathcal{P}$. The *closure* of α for $\mathcal{P}$, denoted by $\mathit{Closure}(\alpha, \mathcal{P})$, is the set $\{b \circ \alpha \mid b : D \to D \text{ is a bijection}\}$.

We now show that the closure of a nogood can be characterized compactly and that membership to the closure of a nogood can be tested in linear time for ICSPs. We first introduce the concept of existential nogood, which simplifies the proofs and intuitions.

Definition 3.4. Let $\mathcal{P} = \langle V, D, C\rangle$ be an ICSP and α be a nogood for $\mathcal{P}$. Let $\mathit{image}(\alpha) = \{d_1, \ldots, d_k\}$. The *existential nogood* of α for $\mathcal{P}$, denoted by $\mathit{Enogood}(\alpha, \mathcal{P})$, is the set of all functions $\gamma : \mathit{dom}(\alpha) \to D$ satisfying

$\exists e_1, \ldots, e_k \in D :$

$\forall i \in 1..k : \forall v_j \in \alpha^{-1}(d_i) : \gamma(v_j) = e_i \wedge \mathit{alldiff}(e_1, \ldots, e_k).$

The following lemma indicates that an existential nogood precisely captures the closure of a nogood.

Lemma 3.5. *Let $\mathcal{P}$ be an ISCP and α be a nogood for $\mathcal{P}$. Then $\mathit{Enogood}(\alpha, \mathcal{P}) = \mathit{Closure}(\alpha, \mathcal{P})$.*

Proof. Let $\mathit{image}(\alpha) = \{d_1, \ldots, d_k\}$. By definition of the image, we have that

$\forall i \in 1..k : \forall v_j \in \alpha^{-1}(d_i) : \alpha(v_j) = d_i \wedge \mathit{alldiff}(d_1, \ldots, d_k).$

We first show that $Closure(\alpha, \mathcal{P}) \subseteq Enogood(\alpha, \mathcal{P})$. Let $\gamma \in Closure(\alpha, \mathcal{P})$. This means that there exists a bijection b such that $\gamma = b \circ \alpha$. Thus, γ satisfies

$$\forall i \in 1..k : \forall v_j \in \alpha^{-1}(d_i) : \gamma(v_j) = b(d_i) \wedge \\ alldiff\,(b(d_1), \ldots, b(d_k))$$

by definition of a bijection, and hence $\gamma \in Enogood(\alpha, \mathcal{P})$. We now show that $Enogood(\alpha, \mathcal{P}) \subseteq Closure(\alpha, \mathcal{P})$. Let $\delta \in Enogood(\alpha, \mathcal{P})$. There exist some values $a_1, \ldots, a_k \in D$ such that

$$\forall i \in 1..k : \forall v_j \in \alpha^{-1}(d_i) : \delta(v_j) = a_i \wedge alldiff\,(a_1, \ldots, a_k).$$

Since $a_1, \ldots, a_k$ are all different, there exists a bijection b satisfying $\forall i \in 1..k : b(d_i) = a_i$. Hence δ can be rewritten as $b \circ \alpha$ and $\delta \in Closure(\alpha, \mathcal{P})$. □

It is not obvious that membership to an existential nogood can be tested efficiently, since it involves an existential quantification. However, due to the nature of the underlying constraints, it is easy to eliminate the existential variables by selecting a representative for each set $\alpha^{-1}(d_i)$. This is precisely the motivation underlying the concept of abstract nogoods defined below. Consider the existential formula

$$\exists e_1, e_2, e_3 \in D : \theta(v_1) = e_1 \wedge \theta(v_2) = e_2 \wedge \theta(v_3) = e_3 \wedge \\ \theta(v_4) = e_1 \wedge \theta(v_5) = e_2 \wedge alldiff\,(e_1, e_2, e_3).$$

The variables e_1, e_2, and e_3 can be eliminated to produce

$$\theta(v_1) = \theta(v_4) \wedge \theta(v_2) = \theta(v_5) \wedge alldiff\,(\theta(v_1), \theta(v_2), \theta(v_3)).$$

Definition 3.6. Let $\mathcal{P} = \langle V, D, C \rangle$ be an ICSP and α be a nogood for $\mathcal{P}$. Let $image(\alpha) = \{d_1, \ldots, d_k\}$ and let

$$v_{r_i} \in \alpha^{-1}(d_i) \quad (1 \le i \le k).$$

The *abstract nogood* of α wrt $\mathcal{P}$, denoted by $Anogood(\alpha, \mathcal{P})$, is the set of functions $\gamma : dom(\alpha) \to D$ satisfying

$$\forall i \in 1..k : \forall v_j \in \alpha^{-1}(d_i) : \gamma(v_j) = \gamma(v_{r_i}) \wedge \\ alldiff\,(\gamma(v_{r_1}), \ldots, \gamma(v_{r_k})).$$

Lemma 3.7. *Let $\mathcal{P}$ be an ICSP and α be a nogood for $\mathcal{P}$. Then $Enogood(\alpha, \mathcal{P}) = Anogood(\alpha, \mathcal{P})$.*

For simplicity, we will often denote the abstract nogood condition in terms of global constraints

$$\forall i \in 1..k : allequal(v_j \in \alpha^{-1}(d_i))\ \gamma(v_j) \wedge \\ alldiff\,(\gamma(v_{r_1}), \ldots, \gamma(v_{r_k}))$$

where $allequal(i \in S)\ a_i$ holds if all the a_i are the same value. It can be seen that there exists a linear time algorithm to test whether a partial assignment θ violates a nogood in $Closure(\alpha, \mathcal{P})$. It suffices to test whether θ satisfies the formula above whenever $dom(\alpha) \subseteq dom(\theta)$.

Lemma 3.8. *Let $\mathcal{P}$ be an ICSP, α be a nogood for $\mathcal{P}$, and θ be a partial assignment. There exists a linear-time algorithm that tests whether θ violates any nogood in $Closure(\alpha, P)$.*

Proof. Direct consequence of lemmas 3.5 and 3.7 and the fact that the abstract nogood is linear in $|V|$. □

Lemma 3.9. *Let $\mathcal{P} = \langle V, D, C \rangle$ be an ICSP, $D = \{d_1, \ldots, d_m\}$, and α be a partial assignment for $\mathcal{P}$ with $dom(\alpha) = \{v_{i_1}, \ldots, v_{i_k}\}$. Let every $\alpha_i = \alpha \wedge v_{i_{k+1}} = d_i$ $(1 \le i \le m)$ be a nogood for $\mathcal{P}$. Then,*

1. *α is a nogood for $\mathcal{P}$;*
2. *if θ violates a nogood in $\bigcup_{i=1}^{m} Closure(\alpha_i, P)$, then θ violates a nogood in $Closure(\alpha, P)$.*

Lemmas 3.8 and 3.9 indicate that symmetry breaking is tractable for ICSPs. Lemma 3.9 indicates that abstract nogoods are needed only for frontier nodes of the search tree (i.e., closed nodes whose parents are open). Once its children are explored, the abstract nogood of the parent subsumes the abstract nogoods of its children. Hence, maintaining the nogood takes space $O(|F||V|)$ if F is the set of frontier nodes. We formalize this result using variable decomposition trees.

Definition 3.10. A *variable decomposition tree* for a CSP $\mathcal{P} = \langle V, D, C \rangle$, where $D = \{d_1, \ldots, d_m\}$, is a search tree where nodes represent partial assignments for $\mathcal{P}$ and nodes are decomposed as follows: given a node representing a partial assignment α, where $dom(\alpha) = \{v_{i_1}, \ldots, v_{i_k}\}$, its children represent the partial assignments $\alpha_i = \alpha \wedge v_{i_{k+1}} = d_i$ $(1 \le i \le m)$ for some variable $v_{i_{k+1}} \in V \setminus dom(\alpha)$.

Note that variable decomposition trees capture both static and dynamic variable orderings.

Theorem 3.11. *Let $\mathcal{P}$ be an ICSP and let F be the set of frontier nodes in a variable decomposition search tree for $\mathcal{P}$. Symmetry breaking for $\mathcal{P}$ requires $O(|F||V|)$ space for storing the nogoods. In addition, testing if a partial assignment θ violates a nogood takes $O(|F||V|^2)$ time in the worst case.*

The result above can be strengthened considerably in fact. We will show that search procedures exploring a variable decomposition tree can remove all symmetries in constant time and space. Before presenting the theoretical results, we illustrate the intuition using depth-first search. The basic intuition comes from the structure of the abstract nogoods. Consider the partial assignment

$$v_1 = 1 \wedge v_2 = 2 \wedge v_3 = 3 \wedge v_4 = 1 \wedge v_5 = 2$$

and assume that depth-first search tries to assign variable v_6 and that the set of possible values is 1...10. The failure of $v_6 = 1$ produces an abstract nogood θ which holds if

$allequal(\theta(v_1), \theta(v_4), \theta(v_6)) \wedge$
$allequal(\theta(v_2), \theta(v_5)) \wedge$
$alldiff(\theta(v_1), \theta(v_2), \theta(v_3))$

Since $v_1, \ldots, v_5$ remain instantiated when the next value is tried for v_6, the abstract nogood for this part of this next branch holds if $\theta(v_6) = 1$ imposing that v_6 be assigned a value different from 1. The assignments $v_6 = 2$ and $v_6 = 3$ produce similar abstract nogoods. Now observe what happens for an assignment of a value $i \in 4..10$, say 6. The abstract nogood θ is now defined as

$allequal(\theta(v_1), \theta(v_4)) \wedge$
$allequal(\theta(v_2), \theta(v_5)) \wedge$
$alldiff(\theta(v_1), \theta(v_2), \theta(v_3), \theta(v_6))$

which can be partially evaluated to $alldiff(1, 2, 3, \theta(v_6))$. The disjunction of all these nogoods can be partially evaluated to

```
bool ILABEL(𝒫) { return ILABELA(𝒫,ε); }
bool ILABELA(⟨V,D,C⟩,θ) {
    if dom(θ) = V then return C(θ);
    select v in V \ dom(θ);
    A := image(θ);
    if image(θ) ≠ D then
        select f in D \ image(θ); A := A ∪ {f};
    forall(d ∈ A)
        θ' := θ ∧ v = d;
        if ¬Failure(⟨V,D,C⟩,θ') then
            if ILABELA(⟨V,D,C⟩,θ') then
                return true;
    return false;
}
```

Figure 1: A Labeling Procedure for ICSPs.

$$\theta(v_6) = 1 \vee \theta(v_6) = 2 \vee \theta(v_6) = 3 \vee \mathit{alldiff}(1, 2, 3, \theta(v_6))$$

which cannot be satisfied by any assignment to v_6. It follows that v_6 must only be assigned to the previously assigned values $1, 2, 3$ or to exactly one new value in 4..10. In other words, in a variable decomposition tree, only some of the children need to be explored: those which assign variable $v_{i_{k+1}}$ to a value in $dom(\alpha)$ and exactly one other child. *Note that this result is independent from the set of constraints*. It is the essence of the labeling procedure for graph-coloring in [Kubale and Jackowski, 1985] and in the scene allocation problem in [Van Hentenryck, 2002]. This procedure, which breaks all symmetries for ICSPs, is formalized in Figure 1. It uses the function $Failure(\mathcal{P}, \theta)$ ($\mathcal{P} = \langle V, D, C\rangle$) which satisfies the property

$$Failure(\mathcal{P}, \theta) \Rightarrow [\forall \beta \in Comp(\theta, \mathcal{P}) : \neg C(\beta)].$$

To our knowledge, this paper provides the first formal proof that these algorithms break all the symmetries in their respective problems. To prove the correctness of ILABEL, and other related search procedures, it is useful to introduce the concept of compact variable decomposition tree.

Definition 3.12. A *compact variable decomposition tree* for an ICSP $\mathcal{P} = \langle V, D, C\rangle$, where $D = \{d_1, \ldots, d_m\}$, is a search tree where nodes represent partial assignments for $\mathcal{P}$ and nodes are decomposed as follows: given a node representing a partial assignment α, where $dom(\alpha) = \{v_{i_1}, \ldots, v_{i_k}\}$, for some variable $v_{i_{k+1}} \in V \setminus dom(\alpha)$, there are children representing the partial assignments $\alpha_i = \alpha \wedge v_{i_{k+1}} = d_o$ for all $d_o \in image(\alpha)$ and exactly one child representing a partial assignment $\alpha_i = \alpha \wedge v_{i_{k+1}} = d_n$ for some $d_n \in D \setminus image(\alpha)$, if $D \setminus image(\alpha)$ is not empty.

The following lemma, which states that compact variable decomposition trees are complete, follows directly from the examination of the nogoods above.

Lemma 3.13. *Let $\mathcal{P} = \langle V, D, C\rangle$ be an ICSP and S be all the complete assignments in a variable decomposition tree for $\mathcal{P}$. Then, the closure $\{b \circ \alpha \mid \alpha \in S \wedge b : D \rightarrow D$ is a bijection$\}$ is equal to $Sol(\mathcal{P})$.*

Our next lemma states that a compact variable decomposition tree never matches any nogood generated during search.

Lemma 3.14. *Let $\mathcal{P}$ be an ICSP and T be a compact variable decomposition tree for $\mathcal{P}$. The partial assignment α of a node in T never matches any nogood generated during the exploration of T (except the one it possibly generates).*

Proof. By Lemma 3.9, it suffices to show that a partial assignment θ never matches a nogood generated by its siblings or the siblings of one of its ancestors in the tree. The proof is by induction on the depth of the tree. At the depth of θ, the result follows from the inspections of the nogood as discussed earlier. Consider a depth $dp' < dp$ and a nogood α generated by one of the left or right branches at that depth. We can restrict attention to the projection of θ to the variables instantiated at that depth, i.e., we can restrict attention to $\theta' : dom(\alpha) \rightarrow D$ satisfying $\forall v \in dom(\alpha) : \theta'(v) = \theta(v)$.

We show that $\theta' \notin Closure(\alpha, \mathcal{P})$. Let v be the variable assigned at depth d'. Observe that there exists a partial assignment α' such that $\alpha = \alpha' \wedge v = e_1$ and $\theta' = \alpha' \wedge v = e_2$, for some $e_1, e_2 \in D$ and $e_1 \neq e_2$. By definition of a compact variable decomposition tree, the values e_1 and e_2 must belong to $image(\alpha') \cup \{d\}$, where $d \notin image(\alpha')$.

Consider the case where $e_1, e_2 \in image(\alpha)$. That means that there exist v_i and v_j in $dom(\alpha)$ such that $\alpha(v) = \alpha(v_i) \neq \alpha(v_j)$ and $\theta'(v) = \theta'(v_j) \neq \theta'(v_i)$. If $\theta' \in Closure(\alpha, \mathcal{P})$, it can be rewritten as $\theta' = b \circ \alpha$ for some bijection b. Hence $\theta'(v) = \theta'(v_i)$, which is impossible.

Assume now that $e_1 \in image(\alpha')$ and $e_2 = d$. It follows that $\alpha(v) = \alpha(v_i)$ for some $v_i \in dom(\alpha')$ and that $\theta'^{-1}(d) = \{v\}$. If $\theta' \in Closure(\alpha, \mathcal{P})$, then $\theta' = b \circ \alpha$ for some bijection. Hence $\theta'(v) = \theta'(v_i)$ which is impossible.

Assume that $e_2 \in image(\alpha')$ and $e_1 = d$. Then, $\alpha^{-1}(d) = \{v\}$ and $\theta'(v) = \theta'(v_i)$ for some $v_i \in dom(\alpha')$. If $\theta' \in Closure(\alpha, \mathcal{P})$, then $\theta' = b \circ \alpha$ for some bijection b. Hence, $\theta'(v) = b(d) \wedge \theta'^{-1}(b(d)) = \{v\}$, which is impossible since $\theta'(v) = \theta'(v_i)$ for some $v_i \in dom(\alpha')$. □

The correctness of Procedure ILABEL follows from the two lemmas above. Other search strategies, e.g., LDS, also remove all symmetries in constant time and space.

Theorem 3.15. *Procedure* ILABEL *breaks all the symmetries in constant time and constant space for an ICSP $\mathcal{P}$, i.e., it never matches any member of the closure of any nogood generated during search.*

4 Piecewise-Interchangeable CSPs

In many applications, e.g., in resource allocation and scheduling, the values are taken from disjoint sets but they are interchangeable in each set. For instance, in the scene allocation problem, we can easily imagine a version of the problem where days are divided in morning and afternoon sessions. Actors probably have strong preferences (and thus different fees for the morning and afternoon sessions) but the day of the session may not matter. This paper captures this (more limited) form of interchangeability by piecewise interchangeable CSPs. We first define this class of CSPs formally and then derive similar results as for ICSPs. We state the main definitions and theorems only, since the derivation is similar to the one for ICSPs. As traditional, we use $D_1 + D_2$ to denote the disjoint union of D_1 and D_2. Our definitions and results only consider two distinct sets of values, for simplicity. It is easy to generalize them to an arbitrary number of sets.

Definition 4.1. Let D_1 and D_2 be two disjoint sets. A *piecewise bijection* $b : D_1 + D_2 \to D_1 + D_2$ is a bijection defined as $b(d) = b_i(d)$ if $d \in D_i$ where $b_i : D_i \to D_i$ is a bijection $(1 \leq i \leq 2)$.

Definition 4.2. Let $\mathcal{P} = \langle V, D_1 + D_2, C\rangle$ be a CSP. $\mathcal{P}$ is *piecewise value-interchangeable* if, for each solution $\sigma \in Sol(\mathcal{P})$ and each piecewise bijection $b : D_1 + D_2 \to D_1 + D_2$, the function $b \circ \sigma \in Sol(\mathcal{P})$.

In the following, we use PICSP as an abbreviation for piecewise value-interchangeable CSP.

Definition 4.3. Let $\mathcal{P} = \langle V, D_1 + D_2, C\rangle$ be a PICSP and α be a nogood for $\mathcal{P}$. The *closure* of α for $\mathcal{P}$, denoted by $Closure(\alpha, \mathcal{P})$, is the set $\{b \circ \alpha \mid b : D_1 + D_2 \to D_1 + D_2$ is a piecewise bijection$\}$.

We now specify the abstract nogoods for PICSPs. The key intuition is to separate the values from D_1 and D_2.

Definition 4.4. Let $\mathcal{P} = \langle V, D_1 + D_2, C\rangle$ be a PICSP and α be a nogood for $\mathcal{P}$. Let $image(\alpha) = \{d^1_1, \ldots, d^1_k, d^2_1, \ldots, d^2_l\}$, where $d^1_i \in D_1, d^2_i \in D_2$, and let

$$v_{r^1_i} \in \alpha^{-1}(d^1_i) \quad (1 \leq i \leq k);$$
$$v_{r^2_i} \in \alpha^{-1}(d^2_i) \quad (1 \leq i \leq l).$$

The *abstract nogood* of α wrt $\mathcal{P}$, denoted by $Anogood(\alpha, \mathcal{P})$, is the set of functions $\gamma : dom(\alpha) \to D_1 + D_2$ satisfying

$$\begin{array}{l}
\forall i \in 1..k : allequal(v_j \in \alpha^{-1}(d^1_i))\ \gamma(v_j) \wedge \\
\forall i \in 1..k : \forall v_j \in \alpha^{-1}(d^1_i) :\ v_j \in D_1 \wedge \\
alldiff(\gamma(v_{r^1_1}), \ldots, \gamma(v_{r^1_k})) \wedge \\
\forall i \in 1..l : allequal(v_j \in \alpha^{-1}(d^2_i))\ \gamma(v_j) \wedge \\
\forall i \in 1..l : \forall v_j \in \alpha^{-1}(d^2_i) :\ v_j \in D_2 \wedge \\
alldiff(\gamma(v_{r^2_1}), \ldots, \gamma(v_{r^2_l})).
\end{array}$$

Figure 2 depicts a labeling procedure `PILABEL` for PICSPs, which breaks all value symmetries in constant time and space. Procedure `PILABEL` generalizes `ILABEL` by considering the already assigned values in both sets D_1 and D_2, as well as one new value (if any) from both sets.

Theorem 4.5. *Procedure* `PILABEL` *breaks all the symmetries in constant time and constant space for PICSPs.*

```
bool PILABEL(𝒫) { return PILABELA(𝒫,ε); }
bool PILABELA(⟨V,D1+D2,C⟩,θ) {
    if dom(θ) = V then return C(θ);
    select v in V \ dom(θ);
    A1 := image(θ) ∩ D1; A2 := image(θ) ∩ D2;
    if image(θ) ∩ D1 ≠ D1 then
        select f in D1 \ image(θ); A1 := A1 ∪ {f};
    if image(θ) ∩ D2 ≠ D2 then
        select f in D2 \ image(θ); A2 := A2 ∪ {f};
    forall(d ∈ A1 ∪ A2)
        θ' := θ ∧ v = d;
        if ¬Failure(⟨V,D1+D2,C⟩,θ') then
            if PILABELA(⟨V,D1+D2,C⟩,θ') then
                return true;
    return false;
}
```

Figure 2: A Labeling Procedure for PICSPs.

5 Wreath-Value Interchangeable CSPs

We now consider another, more complex, class of CSPs, which assigns a pair of values (d_1, d_2) from a set $D_1 \times D_2$ to each variable v. Values in D_1 are fully interchangeable and, for a fixed value in D_1, the values in D_2 are fully interchangeable as well. These problems are called wreath value-interchangeable in this paper, because the symmetry group corresponds to a wreath product of symmetry groups [Camerson, 1999]. Such problems arise naturally in a variety of applications in resource allocation and scheduling. Consider, for instance, the problem of scheduling a meeting where different groups must meet some day of the week in some room subject to constraints. The days are fully interchangeable and, on a given day, the rooms are fully interchangeable. Symmetry breaking for a wreath value-interchangeable CSP is tractable.

Definition 5.1. Let D_1 and D_2 be two sets. A *wreath bijection* $b : D_1 \times D_2 \to D_1 \times D_2$ is a bijection defined as

$$b(\langle d_1, d_2\rangle) = \langle b_1(d_1), b_2^{d_1}(d_2)\rangle$$

where $b_1 : D_1 \to D_1$ is a bijection and $b_2^{d_1}$ $(d_1 \in D_1)$ is a bijection $D_2 \to D_2$.

Definition 5.2. Let $\mathcal{P} = \langle V, D_1 \times D_2, C\rangle$ be a CSP. $\mathcal{P}$ is *wreath value-interchangeable* if, for each solution $\sigma \in Sol(\mathcal{P})$ and each wreath bijection $b : D_1 \times D_2 \to D_1 \times D_2$, the function $b \circ \sigma \in Sol(\mathcal{P})$.

In the following, we use WICSP as an abbreviation for wreath value-interchangeable CSP. We also use the following notations. If $d = (d_1, d_2)$ is a pair, $d[1] = d_1$ and $d[2] = d_2$. If T is a set of tuples, $T[i]$ denotes the set $\{d[i] \mid d \in T\}$ and $filter(T, i, d_i)$ denotes the set $\{d \mid d \in T \wedge d[i] = d_i\}$. If $\alpha : D_1 \times D_2 \to D_1 \times D_2$ is an assignment, $\alpha^{-1}((d_1, D_2))$ denotes the set $\{\alpha^{-1}((d_1, d_2)) \mid d_2 \in D_2\}$.

Definition 5.3. Let $\mathcal{P} = \langle V, D_1 \times D_2, C\rangle$ be a WICSP and α be a nogood for $\mathcal{P}$. The *closure* of α for $\mathcal{P}$, denoted by $Closure(\alpha, \mathcal{P})$, is the set $\{b \circ \alpha \mid b : D_1 \times D_2 \to D_1 \times D_2$ is a wreath bijection$\}$.

We now specify abstract nogoods for WICSPs.

Definition 5.4. Let $\mathcal{P} = \langle V, D_1 \times D_2, C\rangle$ be a WICSP and α be a nogood for $\mathcal{P}$. Let $image(\alpha)[1] = \{d_1, \ldots, d_k\}$, let $filter(image(\alpha), 1, d_i) = \{d^i_1, \ldots, d^i_{l_i}\}$, and let

$$v_{r_i} \in \alpha^{-1}(d_i, D_2) \quad (1 \leq i \leq k),$$
$$v_{r^i_j} \in \alpha^{-1}((d_i, d_j)) \quad (1 \leq i \leq k, 1 \leq j \leq l_i).$$

The *abstract nogood* of α wrt $\mathcal{P}$, denoted by $Anogood(\alpha, \mathcal{P})$, is the set of functions $\gamma : dom(\alpha) \to D_1 \times D_2$ satisfying

$$\begin{array}{l}
\left[\begin{array}{l} \forall i \in 1..k : allequal(v_j \in \alpha^{-1}((d_i, D_2)))\gamma(v_j)[1] \wedge \\ alldiff(\gamma(v_{r_1})[1], \ldots, \gamma(v_{r_k})[1]) \end{array}\right] \wedge \\
\left[\begin{array}{l} \forall i \in 1..l_1 : allequal(v_j \in \alpha^{-1}(d_1, d^1_i)))\gamma(v_j)[2] \wedge \\ alldiff(\gamma(v_{r^1_1})[1], \ldots, \gamma(v_{r^1_{l_1}})[1]) \end{array}\right] \wedge \\
\ldots \\
\left[\begin{array}{l} \forall i \in 1..l_k : allequal(v_j \in \alpha^{-1}(d_1, d^k_i)))\gamma(v_j)[2] \wedge \\ alldiff(\gamma(v_{r^k_1})[1], \ldots, \gamma(v_{r^k_{l_k}})[1]) \end{array}\right].
\end{array}$$

```
bool WILABEL(𝒫) { return WILABELA(𝒫,ε); }
bool WILABELA(⟨V,D1 × D2,C⟩,θ) {
    if dom(θ) = V then return C(θ);
    select v in V \ dom(θ);
    A1 := image(θ)[1];
    if A1 ≠ D1 then
        select f in D1 \ A1; A1 := A1 ∪ {f};
    forall(d1 ∈ A1)
        A2 := filter(image(α),1,d1)[2];
        if A2 ≠ D2 then
            select f in D2 \ A2; A2 := A2 ∪ {f};
        forall(d2 ∈ A2)
            θ' := θ ∧ v = (d1,d2);
            if ¬Failure(⟨V,D1 × D2,C⟩,θ') then
                if WILABELA(⟨V,D1 × D2,C⟩,θ') then
                    return true;
    return false;
}
```

Figure 3: A Labeling Procedure for WICSPs.

Theorem 5.5. *Procedure* WILABEL *breaks all the symmetries in constant time and constant space for a WICSP.*

6 Experimental Results

This section gives some preliminary results showing the benefits of symmetry breaking on these classes of problems. Table 1 gives results on graph coloring, an ICSP, and on partitioned graph coloring, a PICSP. In partitioned graph coloring, the colors are divided in 4 groups and are fully interchangeable in each group. The constraints express the usual graph-coloring property: no two adjacent vertices are colored with the same color. Table 1 gives the result of coloring graphs with 50% edge density with at most $n/(2 * log\_2(n))$ colors, where n is the number of nodes. The first column gives the number of nodes, the second column gives the time to color the graphs using the default labeling procedure, and the third and fourth columns report the time to color the graphs and the partitioned graphs with symmetry breaking. Note that the default labeling does not exploit symmetries and hence its time is similar for both graph coloring and partitioned graph coloring (second column). The last column gives the ratio of PILABEL and the default labeling procedure (the gains are too large for coloring to be reported as a ratio). All results are an average of 5 runs in milliseconds under SICStus-Prolog. They clearly show that symmetry breaking on these CSPs brings significant benefits.

n	DL	$ILABEL$	$PILABEL$	%PILABEL
70	492	4	154	0.31
72	16418	96	4244	0.26
74	30790	34	18576	0.60
76	10622	24	2616	0.25
78	17786	26	15616	0.87
80	45268	86	9736	0.22
82	22946	46	15502	0.67
83	33206	42	9970	0.30

Table 1: Experimental Results on ICSPs and PICSPs.

7 Conclusion

This paper studied three classes of CSPs for which symmetry breaking is tractable. These CSP classes feature various forms of value interchangeability and allow symmetry breaking to be performed in constant time and space during search. Experimental results also show the benefits of symmetry breaking on these CSPs, which encompass many practical applications. There are many directions for future research. Of particular interest is the study of tractable classes of CSPs exhibiting variable symmetries where the set V has a complex structure (e.g., a Cartesian product). There are several classes of such CSPs for which symmetry breaking is tractable, although more complex. Finding effective search procedures for these CSPs is also a challenging problem.

8 Acknowldegments

Special thanks to the reviewers for their very interesting comments. All authors are partly supported by a STINT institutional grant IG2001-67 of STINT. P. Van Hentenryck is partly supported by an NSF ITR Award DMI-0121495. The Sweden-based authors are also supported by grant 221-99-369 of the Swedish Research Council.

References

[Backofen and Will, 1999] Rolf Backofen and Sebastian Will. Excluding symmetries in constraint-based search. In CP'99.

[Barnier and Brisset, 2002] N. Barnier and P. Brisset. Solving the Kirkman's schoolgirl problem in a few seconds. In CP'2002.

[Camerson, 1999] P. Cameron. Permutation Groups, Cambridge Univ. Press, 1999.

[Crawford *et al.*, 1996] J. Crawford et al. Symmetry breaking predicates for search problems. In KR'96.

[Fahle *et al.*, 2001] Schamberger Fahle, Torsten, Stefan, and Meinolf Sellmann. Symmetry breaking. In CP'2001.

[Focacci and Milano, 2001] Filippo Focacci and Michaela Milano. Global cut framework for removing symmetries. In CP'2001.

[Freuder, 1991] E. Freuder. Eliminating interchangeable values in constraint satisfaction problems. In *Proc. AAAI'91*.

[Gent and Smith, 2000] Ian P. Gent and B. Smith. Symmetry breaking during search in constraint programming. In ECAI'2000.

[Kubale and Jackowski, 1985] M. Kubale and D. Jackowski. A Generalized Implicit Enumeration Algorithm for Graph Coloring. *CACM*, 28(4):412–418, 1985.

[Meseguer and Torras, 2001] P. Meseguer and C. Torras. Exploiting symmetries within constraint satisfaction search. *Artificial Intelligence*, 129(1-2):133–163, 2001.

[Puget, 1993] J.-F. Puget. On the satisfiability of symmetrical constrained satisfaction problems. In ISMIS'93.

[Puget, 2002] J.-F. Puget. Symmetry Breaking Revisited. In CP'2002.

[Van Hentenryck, 2002] P. Van Hentenryck. Constraint and Integer Programming in OPL. *Informs Journal on Computing*, 14(4):345–372, 2002.