Symmetry in Constraint Optimization*

Toby Walsh
NICTA and UNSW
Sydney, Australia
tw@cse.unsw.edu.au

Abstract

Symmetry is an important feature in constraint programming. Whilst there has been considerable progress in dealing with symmetry in constraint satisfaction problems, there has been less attention to symmetry in constraint optimization problems. In this paper, we propose some general methods for dealing with symmetry in constraint optimization problems.

1 Introduction

Many search problems contain symmetries. Symmetry occurs naturally in many problems (e.g. if we have identical machines to schedule, or identical jobs to process). Symmetry can also be introduced when we model a problem. Unfortunately, symmetries increases the size of the search space. We must take into account symmetry or we will waste much time visiting symmetric solutions, as well as those parts of the search tree which are symmetric to already visited states. Sophisticated methods have been developed in constraint satisfaction problems to eliminate symmetry either statically by the addition of symmetry breaking constraints [Puget, 1993; Crawford *et al.*, 1996; Aloul *et al.*, 2002; Flener *et al.*, 2002; Aloul *et al.*, 2003; Law and Lee, 2004; Puget, 2005; 2006; Walsh, 2006] or dynamically by modifying the search method to avoid symmetric states [Gent and Smith, 2000; Fahle *et al.*, 2001; Roney-Dougal *et al.*, 2004; Hentenryck *et al.*, 2003]. In this paper, we outline methods to extend such symmetry breaking methods to work in the context of constraint optimization.

2 Formal Background

A constraint satisfaction problem consists of a set of n variables, each with a domain of m possible values, and a set of constraints specifying allowed combinations of values for given subsets of variables. A solution of a constraint satisfaction problem is an assignment of a value to each variable satisfying the constraints. We will use upper case letters like X_i and B_j for variables and lower case letters like d_k for domain values. Without loss of generality, we can assume all variables take values from the same universal domain of values (and unary constraints are used to restrict these domains as appropriate).

A constraint optimization problem is a constraint satisfaction problem with an additional objective function f. A solution of a constraint optimization problem is a complete assignment that satisfies the constraints (a *feasible* assignment) that minimizes or maximizes the objective function as appropriate. In the rest of this paper, we shall consider just minimization as any maximization problem can be turned into a minimization problem by negating the objective function f. To solve such a constraint optimization problem, we consider branch and bound style methods that solve a sequence of constraint satisfaction problems with tightening bounds on the objective function.

3 Symmetry in Satisfaction

Symmetry occurs in many constraint satisfaction problems. A *variable symmetry* of a constraint satisfaction problem is a permutation of the variables that preserves solutions. More formally, a variable symmetry is a bijection σ on the indices of variables such that if $X_1 = d_1, \ldots, X_n = d_n$ is a solution then $X_{\sigma(1)} = d_1, \ldots, X_{\sigma(n)} = d_n$ is also. For example, suppose we wish to assign times (values) to exams (variables) in an exam scheduling problem and we have two exams taken by the same students. As we can interchange these two exams, the problem has a variable symmetry. A *value symmetry* of a constraint satisfaction problem, on the other hand, is a permutation of the values that preserves solutions. More formally, a value symmetry is a bijection θ on the values such that if $X_1 = d_1, \ldots, X_n = d_n$ is a solution then $X_1 = \theta(d_1), \ldots, X_n = \theta(d_n)$ is also. For example, suppose we wish to assign colours (values) to nodes (variables) in a graph colouring problem. Value symmetry permits us to interchange any two colours uniformly throughout a colouring. We can further distinguish between *solution symmetries* which preserve solutions, and *constraint symmetries* which are those solution symmetries that also preserve the set of constraints [Cohen *et al.*, 2006]. As similar results hold for both, we will consider here just solution symmetries.

The set of symmetries of a constraint satisfaction or optimization problem form a group under composition. In this

*NICTA is funded by the Australian Government's Department of Communications, Information Technology and the Arts and the Australian Research Council DCITA and ARC through Backing Australia's Ability and the ICT Centre of Excellence program.

work, we place no restrictions on the type of group. In particular, we are not restricted to products of the symmetry group, S_n. However, we do assume that the symmetries are known in advance. For instance, if we are coloring a graph and use a straight forward model with variables for nodes and values for colors, we know that the values are fully interchangeable. A number of methods have been developed to find symmetries in a constraint satisfaction problem automatically. It would be interesting to adapt these methods to finding symmetries in constraint optimization problems automatically.

Symmetries are problematic as they increase the size of the search space. For instance, if we have m interchangeable values, symmetry increases the size of the search space by a factor of $m!$. One simple and common mechanism to deal with symmetry in constraint satisfaction problems is to add constraints which eliminate symmetric solutions [Puget, 1993]. Suppose we have a set Σ of variable symmetries. We can eliminate all symmetric solutions due to these symmetries by posting "lex leader" constraints which ensure that the solution is ordered lexicographically before any of its symmetries [Crawford *et al.*, 1996]. More precisely, we post:

$$[X_1, \ldots, X_n] \leq_{\text{lex}} [X_{\sigma(1)}, \ldots, X_{\sigma(n)}]$$

For each $\sigma \in \Sigma$ where X_1 to X_n is a given fixed ordering on the variables. Similar lex leader constraints can be posted to eliminate value symmetries, as well as symmetries which act simultaneously on variables and values [Puget, 2006; Walsh, 2006].

4 Variable Symmetry in Optimization

We now lift both the definition of symmetry and the idea of eliminating symmetries by posting lex leader constraints to constraint optimization problems. We define a *variable symmetry* of a constraint optimization problem as a bijection σ on the indices of the variables that preserves feasibility. Note that such a symmetry may *not* preserve the value of the objective function. For example, consider the constraint optimization on two integer variables with constraints $0 \leq X_1 \leq 4$, $0 \leq X_2 \leq 4$, $X_1 + X_2 \leq 6$ and the objective function $-2.X_1 - X_2$. Then X_1 and X_2 are symmetric. Given any assignment that is feasible (e.g. $X_1 = 1$ and $X_2 = 3$), we can swap X_1 and X_2 and construct another feasible assignment (in this case, $X_1 = 3$ and $X_2 = 1$). The objective function does not have this symmetry as the value of the objective changes (from -5 to -7) under this mapping. One appealing feature of this definition of symmetry of a constraint optimization problem is that we can use any of the existing methods for finding symmetries that have been developed for constraint satisfaction problems.

We cannot break variable symmetry in constraint optimization using exactly the same static symmetry breaking methods as in constraint satisfaction. Returning to our running example, suppose we post the lex leader constraint: $[X_1, X_2] \leq_{\text{lex}} [X_2, X_1]$. This gives a new constraint optimization problem with solution -9, whilst the original constraint optimization problem had a solution of -10. The added symmetry breaking constraint conflicts with minimizing the objective. Fortunately, we only need a small modification to the lex leader method to ensure that the symmetry breaking constraints do not conflict. To eliminate symmetric solutions from a constraint optimization problem, we can post a symmetry breaking constraint for each variable symmetry which ensures:

$$[f(X_1, \ldots, X_n), X_1, \ldots, X_n] \leq_{\text{lex}} [f(X_{\sigma(1)}, \ldots, X_{\sigma(n)}), X_{\sigma(1)}, \ldots, X_{\sigma(n)}]$$

Such generalized lex leader constraints limit search to those assignments within each symmetry class to those with the smallest objective value, and between those with an equally small objective, to those that are lexicographically least. Note that if f is constant, this degenerates to the lex leader constraint used to break symmetry in constraint satisfaction problems (as would be expected). We can therefore use generalized lex leader constraints within a branch and bound algorithm to find the optimal solution. To propagate such generalized lex leader constraints, we propose the following simple encoding which introduces Boolean variables, B_i to record where the sequence is lex ordered:

$$\begin{aligned} f(X_1, \ldots, X_n) &\leq f(X_{\sigma(1)}, \ldots, X_{\sigma(n)}) \\ f(X_1, \ldots, X_n) \geq f(X_{\sigma(1)}, \ldots, X_{\sigma(n)}) &\equiv \neg B_1 \\ B_i \vee (X_i \leq X_{\sigma(i)}) & \\ B_i \vee (X_i < X_{\sigma(i)}) &\equiv B_{i+1} \end{aligned}$$

Note that, since $f(X_1, \ldots, X_n) \leq f(X_{\sigma(1)}, \ldots, X_{\sigma(n)})$, we could simply have posted $f(X_1, \ldots, X_n) = f(X_{\sigma(1)}, \ldots, X_{\sigma(n)}) \equiv \neg B_1$. However, such an equality is typically more difficult to propagate. For instance, if f is linear, it is NP-hard to enforce GAC on $f(X_1, \ldots, X_n) = f(X_{\sigma(1)}, \ldots, X_{\sigma(n)})$, but polynomial to enforce GAC on $f(X_1, \ldots, X_n) \geq f(X_{\sigma(1)}, \ldots, X_{\sigma(n)})$. In addition, we can get more pruning by using a more general hypothesis. We now consider this generalized lex leader method for three common classes of objective functions.

4.1 Symmetry-invariant Objective

Consider a constraint optimization problem with variable symmetry in which the objective function is invariant to symmetry. That is, $f(X_1, \ldots, X_n) = f(X_{\sigma(1)}, \ldots, X_{\sigma(n)})$ for every symmetry σ. Then, the generalized lex leader constraints simplify to the previous lex leader constraints:

$$[X_1, \ldots, X_n] \leq_{\text{lex}} [X_{\sigma(1)}, \ldots, X_{\sigma(n)}]$$

4.2 Linear Objective

Consider a constraint optimization problem with variable symmetry in which the objective function is linear. That is, $f(x_1, \ldots, X_n) = \sum_{i=1}^{n} a_i.X_i$. Then, we post:

$$\begin{aligned} \sum_{i=1}^{n} (a_i - a_{\sigma(i)}).X_i \leq 0 & \\ \sum_{i=1}^{n} (a_i - a_{\sigma(i)}).X_i \geq 0 &\equiv \neg B_1 \\ B_i \vee (X_i \leq X_{\sigma(i)}) & \\ B_i \vee (X_i < X_{\sigma(i)}) &\equiv B_{i+1} \end{aligned}$$

Returning to our running example, we get:

$$\begin{aligned} X_2 \leq X_1 & \\ X_2 \geq X_1 & \equiv \neg B_1 \\ B_1 \vee (X_1 \leq X_2) & \\ B_1 \vee (X_1 < X_2) & \equiv B_2 \\ B_1 \vee (X_2 \leq X_1) & \\ B_2 \vee (X_2 < X_1) & \equiv B_3 \end{aligned}$$

These symmetry breaking constraints reduce the 22 feasible solutions down to the optimal solution ($X_1 = 4, X_2 = 2$ with objective value -10) and 9 other feasible solutions, each representative of an equivalence class of solutions with a different objective value.

4.3 Value-based Objective

Consider a constraint optimization problem with variable symmetry in which the objective function is just a function of the set or multi-set of values used by the feasible assignment. One such objective is $\sum_{i=1}^{n} g(X_i)$ where $g(X)$ represents the cost of using the value X. A second example is the objective which counts the number of values used, $|\{X_i \mid 1 \leq i \leq n\}|$. Such objective functions are invariant to variable symmetries: $f(X_1, \ldots, X_n) = f(X_{\sigma(1)}, \ldots, X_{\sigma(n)})$. In such cases, the generalized lex leader method returns the simple lex leader constraint:

$$[X_1, \ldots, X_n] \leq_{\text{lex}} [X_{\sigma(1)}, \ldots, X_{\sigma(n)}]$$

5 Value Symmetry in Optimization

In a similar fashion, we define a *value symmetry* of a constraint optimization problem as a bijection θ on values that preserves feasibility. Again, such a symmetry may *not* preserve the value of the objective function. For example, consider again the constraint optimization problem on two integer variables with constraints $0 \leq X_1 \leq 4$, $0 \leq X_2 \leq 4$, $X_1 + X_2 \leq 6$ and the objective function $-2.X_1 - X_2$. The values 1 and 2 are symmetric. Given any assignment that is feasible (e.g. $X_1 = 1$ and $X_2 = 3$), we can swap 1 and 2 and construct another feasible assignment (in this case, $X_1 = 2$ and $X_2 = 3$). The objective function does not have this symmetry as the value of the objective changes (from -5 to -7) under this mapping.

As with variable symmetries, we cannot use the same static symmetry breaking methods in constraint optimization as in constraint satisfaction. Returning to our running example, suppose we post the lex leader constraint: $[X_1, X_2] \leq_{\text{lex}} [\theta(X_1), \theta(X_2)]$ where $\theta(1) = 2$, $\theta(2) = 1$ and $\theta(i) = i$ otherwise. This gives a new constraint optimization problem with solution -9, whilst the original constraint optimization problem had a solution of -10. The added symmetry breaking constraint again conflicts with minimizing the objective. We can, however, modify the lex leader method to ensure that the symmetry breaking constraints do not conflict with the objective function. To eliminate symmetric solutions from a constraint optimization problem, we can post a symmetry breaking constraint for each value symmetry which ensures:

$$[f(X_1, \ldots, X_n), X_1, \ldots, X_n] \leq_{\text{lex}}$$
$$[f(\theta(X_1), \ldots, \theta(X_n)), \theta(X_1), \ldots, \theta(X_n)]$$

Again, this limits search to those assignments within each symmetry class with the smallest objective value, and between those with an equally small objective, to those that are lexicographically least. To propagate such generalized lex leader constraints, we propose the following simple encoding which introduces Boolean variables, B_i to record where the sequence is lex ordered:

$$\begin{aligned} f(X_1, \ldots, X_n) \leq f(\theta(X_1), \ldots, \theta(X_n)) & \\ f(X_1, \ldots, X_n) \geq f(\theta(X_1), \ldots, \theta(X_n)) & \equiv \neg B_1 \\ B_i \vee (X_i \leq \theta(X_i)) & \\ B_i \vee (X_i < \theta(X_i)) & \equiv B_{i+1} \end{aligned}$$

Returning to our running example, these symmetry breaking constraints reduce the 22 feasible solutions down to the optimal solution ($X_1 = 4, X_2 = 2$ with objective value -10) and 13 other feasible solutions. They eliminate, for instance, the feasible assignment $X_1 = 1$, $X_2 = 0$ as this has an objective value of -2 which is larger than the objective value of -4 corresponding to the symmetric assignment $X_1 = 2, X_2 = 0$. By contrast, the usual lex leader constraint, $[X_1, X_2] \leq_{\text{lex}} [\theta(X_1), \theta(X_2)]$ would eliminate $X_1 = 2$, $X_2 = 0$ in favour of the symmetric and lexicographically smaller assignment $X_1 = 1, X_2 = 0$.

6 Generator symmetries

One difficulty with the generalized lex leader method is that the set of symmetries, Σ can be exponentially large in general. For instance, if we have m interchangeable values, then Σ contains $m!$ symmetries. To deal with large number of symmetries in propositional satisfiability problems, Aloul *et al.* suggest we might break only a subset of the symmetries [Aloul *et al.*, 2002]. For example, we might only break those symmetries which are generators of the symmetry group. This idea lifts immediately to symmetries in constraint optimization problems. We can post generalized lex leader constraints corresponding to just a subset of the symmetries. In the case of interchangeable values, we have shown that posting just those symmetry breaking constraints corresponding to the generators which swap neighbouring values is enough to eliminate all symmetric solutions.

7 Variable and Value Symmetry

If a constraint optimization problem contains both variable and value symmetries, the symmetry breaking constraints for the variable and value symmetries can be combined. Note that each symmetry breaking constraint has to order the variables within an assignment in the same way. Symmetries can also act simultaneously on both the variables and values. Consider, for instance, a standard model of the n-queens problem with a variable, X_i representing the position of the queen along the ith row. The rotational symmetry of the chessboard maps a solution with $X_i = j$ onto a solution with $X_j = n - i + 1$. We can adapt the generalized lex leader method to deal with such symmetries in a straight forward way.

We define a *variable/value symmetry* of a constraint optimization problem as a bijection σ on variable assignments that preserves feasibility. For example, in the n-queens problem, the variable assignment $X_i = j$ maps onto $X_j = n - i + 1$. Let $\sigma(X_1, \ldots, X_n)$ be the mapping of the complete assignment $X_1, \ldots, X_n$ by the variable/value symmetry σ. Then, we can break symmetry by posting the generalized lex leader constraints:

$$[f(X_1, \ldots, X_n), X_1, \ldots, X_n] \leq_{\text{lex}}$$

$$[f(\sigma(X_1, \ldots, X_n)), \sigma(X_1, \ldots, X_n)]$$

For each σ in the set of symmetries, where X_1 to X_n is some fixed order on the variables.

8 Case Studies

To illustrate these methods for breaking symmetry in constraint optimization, we present two case studies.

8.1 Bin packing

Consider a simple model of bin packing problems where we have a decision variable for each item, and the value taken is the number of the bin into which the item is packed. The goal is to minimize the number of values used. If all bins are the same size, then we can swap the items in any two bins. Hence, we have a value symmetry in which values are interchangeable. The objective function counts the number of values used. This is invariant to the symmetry of interchanging values. Hence, the generalized lex leader constraints simply ensures that we construct the assignment which is lexicographically least. In [Walsh, 2006], I prove that this is equivalent to the global PRECEDENCE constraint.

Suppose we have two items of the same size. Then we can swap the two items in any packing. This corresponds to a variable symmetry which interchanges the corresponding two decision variables. The objective function is invariant to this type of variable symmetry. The generalized lex leader constraints therefore again ensures that we construct the assignment which is lexicographically least. In [Flener *et al.*, 2006], it is shown that this can be ensured using lexicographical ordering constraints on the "signatures" of interchangeable values. All such variable and value symmetry in a bin packing optimization problem can therefore be broken using existing methods.

8.2 Car sequencing

We consider an optimization version of car sequencing (prob001 in CSPLib) in which each car has a due date, and the objective is to minimize the sum of the tardiness. Production capacity constraints remain the same (e.g. only 1 in 2 cars at any point along the production line can have the sun roof option). A simple model of this problem has a decision variable for each position on the production line, and the value taken is the car being produced. Suppose we have two cars, j and k with the same options. Without loss of generality, suppose that car j is due before car k where $j < k$. This gives a value symmetry since we can interchange the corresponding two values and preserve feasibility of the solution. However, the objective function is not, in general, invariant to this symmetry.

Consider the generalized lex leader constraints corresponding to this value symmetry. There are three cases. In the first case, both cars are produced before j's due date. Then the objective function is invariant to symmetry. Hence, the generalized lex leader constraints ensures that we find the assignment which is lexicographically smaller than its symmetry. That is, the assignment where j is used before k. In the second case, both cars are produced after k's due date. The objective function is again invariant to symmetry, and the generalized lex leader constraints ensures that we find the assignment which is lexicographically smaller than its symmetry. That is, the assignment where j is used before k. In the third case, at least one car is produced between j and k's due dates. There are three subcases. In the first, both cars are produced between j and k's due dates. Here, the generalized lex leader constraints ensures that we find the assignment whose objective value is less than its symmetry. This requires j to be used before k. The other two subcases are similar. In each, the value j must be used before k. Thus, in every case, the generalized lex leader constraints ensure that the value j must be used before k. This implied constraint breaks the symmetry of these interchangeable values.

9 Dynamic Methods

Another way to deal with symmetry is to adapt the search method to avoid exploring symmetric branches. For instance, after exploring a branch, the Symmetry Breaking During Search method (SBDS) adds a symmetry breaking constraint to avoid visiting any branches which are symmetric [Gent and Smith, 2000]. More precisely, suppose the current partial assignment is A and extending this with the variable assignment $X_i = j$ results in backtracking, then we can add the implied constraint, $\sigma(A) \Rightarrow \neg\sigma(X_i = j)$ where σ is any symmetry not already broken by A.

The advantage of such a method is that is does not conflict with the branching heuristic. Static symmetry breaking constraints, on the other hand, eliminate particular symmetric solutions, and these might be the precise solutions which the branching heuristic is directing the backtrack search procedure towards. To adapt SBDS to deal with symmetries in constraint optimization, we observe that we only want to eliminate those symmetric states with an equal or higher objective value[1]. Hence, suppose the current partial assignment is A and extending this with the variable assignment $X_i = j$ results in backtracking, then we can add the implied constraint, $(\sigma(A) \wedge f_{lb}(\sigma(A)) \geq f_{ub}(A)) \Rightarrow \neg\sigma(X_i = j)$ where f_{lb} and f_{ub} are lower and upper bounds on the objective function respectively, and σ is some symmetry not already broken by A.

Another way to adapt the search method is Symmetry Breaking by Dominance Detection (SBDD) [Fahle *et al.*, 2001]. In SBDD, check is performed at each node to see if it is symmetric to one already explored. The key idea in SBDD is the definition of *dominance*. In constraint satisfaction problems, a node in the search tree is dominated by another iff the

[1] Recall that we consider just minimization problems.

variable assignments of the second are symmetric to a subset of the first. SBDD uses a dominance test to ensure only undominated nodes are explored. This definition of dominance needs to be weakened for constraint optimization as we may want to visit symmetric nodes provided they have a smaller objective value. More precisely, in constraint optimization, a node in the search tree is dominated by another iff the variable assignments of the second are symmetric to a subset of the first *and* a lower bound on the objective value for the first is greater than or equal to an upper bound on the objective value for the second. With this weakened definition of dominance, SBDD can then be used in branch and bound style algorithms.

10 Related Work

Puget proved that symmetric solutions can be eliminated from constraint satisfaction problems by the addition of suitable constraints [Puget, 1993]. Crawford *et al.* presented the first general method for constructing variable symmetry breaking constraints within constraint satisfaction problems [Crawford *et al.*, 1996]. Petrie and Smith adapted this method to value symmetries by posting a lexicographical ordering constraint for each value symmetry [Petrie and Smith, 2003]. Puget and Walsh independently proposed propagators for such value symmetry breaking constraints [Puget, 2006; Walsh, 2006]. Aloul *et al.* have adapted such symmetry breaking methods to Boolean optimization [Aloul *et al.*, 2005]. When the constraints and objective function have similar sets of symmetries, they statically break symmetries in the intersection of these two sets. When the intersection of these two sets of symmetries is small, they use a dynamic method to break symmetries of any infeasible assignments that are discovered. This permits any symmetry of the constraints to be used.

Within branch and bound and dynamic programming algorithms, *dominance relations* are often used to prune symmetric search states. Let S be the set of search states (typically, partial assignments), and $f(s)$ for $s \in S$ be the minimum cost feasible solution that is an extension of s. A dominance relation, $\prec$ is a partial ordering[2] over search states satisfying $s_i \prec s_j$ implies $f(s_i) \leq f(s_j)$. Thus, if we have already expanded s_i, we can prune s_j. The formal definition of dominance relation given in [Kohler and Steiglitz, 1974] also includes consistency with the lower-bound function (namely, $s_i \prec s_j$ implies $L(s_i) \leq L(s_j)$ where $L(s)$ is a lower bound on the cost of any feasible solution beneath s). Ibaraki proved conditions under which a stronger dominance relation is guaranteed to give a smaller branch-and-bound search tree [Ibaraki, 1977]. Finally, Yu and Wah used machine learning techniques to propose new dominance relations for 0/1 knapsack, scheduling and related problems [Yu and Wah, 1988].

One way to use dominance relations is to keep track of some or all previous states and only explore new states that are not dominated. Dominance relations are thus a generalization of lower-bound pruning. Another way to us dominance relations is to construct constraints that prune dominated search states [Prestwich and Beck, 2004]. Let S_d be some subset of dominated states. That is, $s \in S_d$ implies there exists $s_j \in S$ such that $s_j \prec s_i$. We need a constraint C such that $C(s)$ holds iff $s \in S_d$. However, there is no general method to construct such constraints. Such dominance constraints can, however, do more pruning than the symmetry breaking constraints proposed here. They can prune all feasible states with an equivalence class of assignments if the minimum value of the objective function in this equivalence class is larger than the value taken by some other feasible assignment.

[2] A partial order is transitive, reflexive ($s \prec s$) and antisymmetric ($s_i \prec s_j$ and $s_j \prec s_i$ implies $s_i = s_j$).

11 Conclusion and Future Work

We have shown how the lex leader method for symmetry breaking can be adapted to work with constraint optimization problems. The key idea is to post symmetry breaking constraints which eliminate within each equivalence class of assignments, all but those assignments with the best objective value, and between those assignments with the best objective value, the single assignment which is lex ordered the least. There are many directions for future research. First, we should consider special classes of symmetries like interchangeable values, or row and column symmetries, where symmetry breaking may be more tractable. Second, we might identify other situations where the generalized lex leader constraints can be simplified. Third, we should consider how these generalized lex leader constraints interact with the problem constraints. For instance, how do they interact with an all-different constraint over the decision variables? Fourth, we might develop specialized propagators for reasoning about the symmetries of the objective function. Fifth, we should test how these methods work in practice on constraint optimization problems.

References

[Aloul *et al.*, 2002] F.A. Aloul, A. Ramani, I. Markov, and K.A. Sakallah. Solving difficult SAT instances in the presence of symmetries. In *Proceedings of the Design Automation Conference*, pages 731–736, 2002.

[Aloul *et al.*, 2003] F.A. Aloul, K.A. Sakallah, and I.L. Markov. Efficient symmetry breaking for Boolean satisfiability. In *Proceedings of the 18th International Joint Conference on AI*, pages 271–276. International Joint Conference on Artificial Intelligence, 2003.

[Aloul *et al.*, 2005] F.A. Aloul, A. Ramani, I. Markov, and K.A. Sakallah. Dynamic symmetry-breaking for improved boolean optimization. In *Proceedings of the Asia South Pacific Design Automation Conference (ASPDAC)*, pages 445–450, 2005.

[Cohen *et al.*, 2006] D. Cohen, P. Jeavons, C. Jefferson, K.E. Petrie, and B.M. Smith. Symmetry definitions for constraint satisfaction problems. *Constraints*, 11(2–3):115–137, 2006.

[Crawford *et al.*, 1996] J. Crawford, G. Luks, M. Ginsberg, and A. Roy. Symmetry breaking predicates for search

problems. In *Proceedings of the 5th International Conference on Knowledge Representation and Reasoning, (KR '96)*, pages 148–159, 1996.

[Fahle *et al.*, 2001] T. Fahle, S. Schamberger, and M. Sellmann. Symmetry breaking. In T. Walsh, editor, *Proceedings of 7th International Conference on Principles and Practice of Constraint Programming (CP2001)*, pages 93–107. Springer, 2001.

[Flener *et al.*, 2002] P. Flener, A. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, J. Pearson, and T. Walsh. Breaking row and column symmetry in matrix models. In *8th International Conference on Principles and Practices of Constraint Programming (CP-2002)*. Springer, 2002.

[Flener *et al.*, 2006] P. Flener, J. Pearson, M. Sellmann, and P. Van Hentenryck. Static and dynamic structural symmetry breaking. In *Proceedings of 12th International Conference on Principles and Practice of Constraint Programming (CP2006)*. Springer, 2006.

[Gent and Smith, 2000] I.P. Gent and B.M. Smith. Symmetry breaking in constraint programming. In W. Horn, editor, *Proceedings of ECAI-2000*, pages 599–603. IOS Press, 2000.

[Hentenryck *et al.*, 2003] P. Van Hentenryck, M. Agren, P. Flener, and J. Pearson. Tractable symmetry breaking for CSPs with interchangeable values. In *Proceedings of the 18th International Conference on AI*. International Joint Conference on Artificial Intelligence, 2003.

[Ibaraki, 1977] T. Ibaraki. The power of dominance relations in branch-and-bound algorithms. *Journal of the Association for Computing Machinery*, 24(2):264–279, 1977.

[Kohler and Steiglitz, 1974] W.H. Kohler and K. Steiglitz. Characterization and theoretical comparison of branch-and-bound algorithms for permutation problems. *Journal of the Association for Computing Machinery*, 21(1):140–156, 1974.

[Law and Lee, 2004] Y.C. Law and J.H.M. Lee. Global constraints for integer and set value precedence. In *Proceedings of 10th International Conference on Principles and Practice of Constraint Programming (CP2004)*, pages 362–376. Springer, 2004.

[Petrie and Smith, 2003] Karen E. Petrie and Barbara M. Smith. Symmetry Breaking in Graceful Graphs. Technical Report APES-56a-2003, APES Research Group, June 2003.

[Prestwich and Beck, 2004] S.D. Prestwich and C.J. Beck. Exploiting dominance in three symmetric problems. In *Proceedings of 4th International Workshop on Symmetry in Constraint Satisfaction Problems (SymCon-04)*, pages 63–70, 2004. Held alongside 10th International Conference on Principles and Practice of Constraint Programming (CP2004).

[Puget, 1993] J.-F. Puget. On the satisfiability of symmetrical constrained satisfaction problems. In J. Komorowski and Z.W. Ras, editors, *Proceedings of ISMIS'93*, LNAI 689, pages 350–361. Springer-Verlag, 1993.

[Puget, 2005] J-F. Puget. Breaking all value symmetries in surjection problems. In P. van Beek, editor, *Proceedings of 11th International Conference on Principles and Practice of Constraint Programming (CP2005)*. Springer, 2005.

[Puget, 2006] J-F. Puget. An efficient way of breaking value symmetries. In *Proceedings of the 21st National Conference on AI*. American Association for Artificial Intelligence, 2006.

[Roney-Dougal *et al.*, 2004] C. Roney-Dougal, I. Gent, T. Kelsey, and S. Linton. Tractable symmetry breaking using restricted search trees. In *Proceedings of ECAI-2004*. IOS Press, 2004.

[Walsh, 2006] T. Walsh. General symmetry breaking constraints. In *12th International Conference on Principles and Practices of Constraint Programming (CP-2006)*. Springer-Verlag, 2006.

[Yu and Wah, 1988] C.-F. Yu and B.W. Wah. Learning dominance relations in combined search problems. *IEEE Transactions on Software Engineering*, 14(8):1155–1175, 1988.