Breaking Symmetries in Matrix Models: A Brief Overview

**Pierre Flener[1], Alan M.Frisch[2], Brahim Hnich[3], Chris Jefferson[2]
Zeynep Kiziltan[4], Ian Miguel[2], Justin Pearson[1] and Toby Walsh[3]**

1 Introduction

Many real life problems can be formulated as constraint satisfaction problems (CSP). When a problem is formulated as a CSP, often there are symmetries, either from the original problem or introduced during the transformation into a CSP. When we attempt to search for a solution to the problem the search tree will be much larger due to having to search symmetric sections of tree (which may or may not contain solutions). There are many different kinds of symmetry that can appear, and there are different ways of trying to remove some or all of the symmetries that appear. Here we are going to overview our work on index symmetry (also known as row and column symmetries of matrices in the two index case) and our methods to break this by adding extra constraints to the problem formulation.

A finite CSP (the type we shall consider here) consists of a finite set of variables each of which has a finite domain of values it can take and a set of constraints, each of which specify allowed assignments of values to some subset of the variables. We shall talk about a *variable assignment* being a mapping to each variable from its domain of possible values. A solution of a CSP is a variable assignment where each constraint is satisfied. Throughout this paper we will refer to comparing values of variables with $\leq$, however any total ordering would do (the most obvious example being $\geq$). Using another ordering may allow easier computation or may fit in better with some other part of the problem.

The most common method of solving a CSP is searching the space of partial assignments, that is we recursively choose a variable and then branch on each value in the domain of that variable. We continue until either we find a solution or we exhaust the search tree. It is possible to improve this search by trying to identifying parts of the tree where there can be no solution, and not search them. The most common way of doing this is by *domain pruning algorithms.* These look at the values the variables can currently take and try to find values which can never be part of a consistent solution. For example if we have two variables, x which can take the values $\{1, 2, 3\}$ and y which can take the values $\{0, 1, 2, 3, 4\}$ and the constraint $x < y$ then clearly y can never take the values 0 or 1 whatever x is.

For a single constraint the strongest form of domain reduction is *generalized arc consistency* (GAC), where we remove every value from the domains of the variables in that constraint which can never satisfy that constraint given the current domains of the other variables. Note that this does not mean that the search for a solution becomes trivial; this does not tell us there is some assignment of values to variables which satisfy all constraints.

A *variable symmetry* of a CSP is a bijective mapping of the set of variables to itself which maps solutions of the CSP to solutions and non-solutions to non-solutions. It is simple to show that the set of variable symmetries of a CSP form a group under function composition [1].

We define a *symmetry class* of an assignment to the variables of a CSP by the set of other assignments which are equivalent to it. There are various methods of trying to reduce the amount of symmetry in the problem. The one we shall consider here is imposing extra constraints. Clearly we do not want to lose solutions, so we shall define a set of symmetry breaking constraints to be *consistent* if at least one variable assignment from each class of symmetries satisfies the constraints. A set of symmetry breaking constraints is *total* if it allows only one variable assignment from each class of symmetries.

Often all or a subset of the variables of a CSP have a natural representation as indexed variables (that is they can be represented as a set $\{x_i\}$, $\{x_{i,j}\}$, etc.) In this situation a common form of symmetry is index symmetry. Index symmetry of the j^{th} index implies that for any two values a and b that index could take, swapping the values of all the variables with a as their j^{th} index and those with b is a variable symmetry.

As an example we can consider a single index problem, where we have 3 variables x_0, x_1 and x_2 where each variable takes a integer value from the range 0 to 9 and together satisfy the constraint $x_0 + x_1 + x_2 = 9$. This problem obviously has index symmetry which leads to symmetric solutions ($x_0 = 1, x_1 = 2, x_2 = 7$ and $x_0 = 7, x_1 = 2, x_2 = 0$ for example) and in fact every solution is symmetric to as many as 5 other solutions. Also we will search large sections of the space of partial solutions which are equivalent. We can reduce both the size of the search and the number of solutions found by imposing the consistent and total symmetry breaking constraints $x_1 \leq x_2$ and $x_2 \leq x_3$.

2 Symmetry in "two index" problems

The most studied variable index representation is where we have two indices, that is we can represent the variables as a set $\{x_{i,j} | 0 \leq i < n, 0 \leq i < m\}$. Note that this also gives us a natural representation of the variables as a matrix, where we place the value of variable $x_{i,j}$ in the ith row and jth column. We can have symmetry on either the first or second index (referred to as row and column

[1]Department of Information Technology, Box 337, SE-751 05 Uppsala, Sweden
[2]AI Group, Department of Computer Science, University of York, York, England
[3]Cork Constraint Computation Center, University College Cork, Ireland
[4]Information Science Department, Box 512, S-751 20 Uppsala, Sweden

symmetry). First index (or row) symmetry means that for any i and j if we swap the values of the pairs of variables $x_{i,k}$ and $x_{j,k}$ for $0 \leq k < m$ in a solution then we also get a solution. Second index symmetry is defined in the obvious manner.

If we have just row symmetry then we can perform total and consistent symmetry breaking in a similar fashion to the single index problem by imposing the constraints $\{(x_{i,0}, ..., x_{i,m-1}) \leq_{lex} (x_{i+1,0}, ..., x_{i+1,m-1}) | 0 \leq i < n-1\}$ where $\leq_{lex}$ constrains the two vectors of variables to be ordered lexicographically. There exists an $O(m)$ algorithm to which can perform GAC on this constraint [4].We can construct a similar set of constraints for column symmetry.

Things become much less trivial if we consider having both row and column symmetry at the same time. This would mean we could swap any two rows, or any two columns and transform solutions into solutions.

Looking at the work we have already done the most obvious solution would be to try to add lexicographic constraints to both the rows and the columns. The problem with this is that the lexicographic order of the rows of a matrix is effected by lexicographically ordering the columns. Is constraining both the rows and columns a consistent or total set of symmetry breaking constraints?

It turns out that we can always find a matrix that satisfies lexicographic ordering on both the rows and columns as long as we constrain the rows and the columns either both with the lexicographically largest first, or lexicographically least first. If we constrain them in different ways the constraints are not consistent (proofs and examples of this are included in [3]). However even when this is a consistent symmetry breaking method it turns out to not be total in all cases (although there are some cases where it is).

We may wonder if there is a total symmetry breaking set of constraints for where we have both row and column symmetry. There is in fact a method of consistently and totally breaking any variable symmetry by the introduction of lexicographic constraints [6], however the method of breaking the symmetry described in that paper (which is the current best for this, and many other symmetry groups) involves the introduction of $O(v!)$ constraints where v is the number of variables involved in the symmetry.

We have proved that for all i, the set of constraints $\{(x_{0,0}, ..., x_{0,m-1}) \leq Y | Y$ is a permutation of $(y_{i,0}, ..., y_{i,m-1})\}$ along with $\leq_{lex}$ on both the rows and columns is consistent (but still not total). This set of constraints is of factorial size so appears too large to impose. However we have created an efficient ($O(m \log m)$) algorithm for enforcing arc consistency on the entire set of constraints $\{X \leq_{lex} Y' | Y'$ is a perm of $Y\}$ for two equal length vectors of variables X and Y, which we shall refer to as $X \leq_{perm} Y$. Unfortunately we cannot apply $\leq_{perm}$ to both the rows and the columns at the same time without losing consistency.

While applying $\leq_{lex}$, and possibly $\leq_{perm}$ as well, creates an algorithm for breaking many of the symmetries quickly, it is difficult to analyze exactly which elements of the symmetry classes are removed by these constraints as the $\leq_{lex}$ constraints on the rows, the $\leq_{lex}$ constraints on the columns and the $\leq_{perm}$ constraints effect each other and interact in complex ways. It is therefore worth looking to see if there are other algorithms we could use to break symmetries.

Instead of constraining each index lexicographically, we could instead use constraint the vectors of variables under the multiset ordering (referred to by the constraint $\leq_{ms}$) [2]. At first glance multiset looks less efficient. Even if we only have row (or column) symmetry, imposing multiset ordering on the rows (or columns) is not sufficient to break all the symmetries. However the major advantage of multiset is that constraining one index under the multiset ordering does not directly place any constraint on any other index, as the multiset order does not use the order of the elements it constrains. Therefore trivially we know we will get a consistent system when we apply multiset over multiple indices. While the current best algorithm for imposing multiset ordering on two vectors of variables is not as efficient as the algorithm for lexicographic ordering, it is still efficient enough to be used in large problems.

This work has been generalized to more than two indices. The best set of total symmetry breaking constraints we know of is still $O(v!)$ in size, but it can be shown we can impose $\leq_{lex}$ or $\leq_{ms}$ on all of the indices (if there is symmetry there of course) and have a consistent although not total set of symmetry breaking constraints, and we can impose $\leq_{perm}$ as well as $\leq_{lex}$ on just one of the indices.

3 Partial Symmetry

Many problems have only partial rather than full index symmetry, that is there is one or more disjoint subsets of the values some index can take for which we can interchange the values and map solutions to solutions. For example in the steel mill problem [5] the rows represent slabs of steel, some of which are equivalent. We have shown that these partial symmetries can also be broken by using the $\leq_{lex}$ and $\leq_{ms}$ constraints discussed earlier but now rather than applying these between all adjacent pairs of index values we apply them only to pairs from each of the sets of interchangeable index values. It is also possible to use the $\leq_{perm}$ ordering here although only on one of the sets of interchangeable values.

Acknowledgments

This project and the sixth author are supported by EPSRC Grant GR/N16129. The eighth author is supported by Science Foundation Ireland. Authors one, three, five and seven are supported by VR grant 221-99-369.

References

[1] J. Crawford. A theoretical analysis of reasoning by symmetry in first-order logic, 1992.

[2] P. Flener, A. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, J. Pearson, and T. Walsh. Breaking row and column symmetries in matrix models, 2002.

[3] Pierre Flener, Alan Frisch, Brahim Hnich, Zeynep Kiziltan, Ian Miguel, Justin Pearson, and Toby Walsh. Symmetry in matrix models. Technical Report 2001-022, 2001.

[4] Alan Frisch, Brahim Hnich, Zeynep Kiziltan, Ian Miguel, and Toby Walsh. Global constraints for lexicographic orderings. In *Proceedings of CP'02*, 2002.

[5] Alan M. Frisch, Ian Miguel, and Toby Walsh. Symmetry and implied constraints in the steel mill slab design problem.

[6] E. Luks and A. Roy. Symmetry breaking in constraint satisfaction, 2002.