AI Communications 17 (2004) 167–169
IOS Press

Thesis

Symmetry breaking ordering constraints

Zeynep Kiziltan
Computer Science Division, Department of Information Science, Uppsala University, Sweden
E-mail: Zeynep.Kiziltan@dis.uu.se

The thesis defended in this dissertation is that: *Row and column symmetry is a common type of symmetry in constraint programming. Ordering constraints can effectively break this symmetry. Efficient global constraints can be designed for propagating such ordering constraints.*

A wide range of problems originating from combinatorics, design, configuration, scheduling, timetabling, bioinformatics, and code generation can be modelled as constraint satisfaction problems. Often such models have a matrix of decision variables. We observe that such matrix modelling provides an effective way of representing these diverse problems and there are two patterns that commonly arise in matrix models: row and column symmetry, and value symmetry.

A 2-d matrix has row (resp. column) symmetry iff its rows (resp. columns) represent indistinguishable objects. An $n \times m$ matrix with row and column symmetry has $n!m!$ symmetries which increase super-exponentially. Thus, it can be very costly to visit all the symmetric branches in a tree search. Symmetry breaking methods such as SES [1], SBDS [7], and SBDD [2,4] cut off all the symmetric parts of the search tree and are applicable to any class of symmetries. However, SES and SBDS treat each symmetry individually, which is impractical when the number of symmetries is large. The dominance checks of SBDD can be very expensive in the presence of many symmetries. We therefore need special techniques to deal with row and column symmetry effectively.

A matrix has value symmetry iff the values in the domain of the variables are indistinguishable. Even though value symmetry is not confined to matrix models, value symmetry in a matrix can be transformed to, for instance, row symmetry. This is another advantage of developing effective techniques for dealing with row and column symmetries.

$$
\begin{array}{ccccccc}
 & \multicolumn{6}{c}{\longrightarrow \text{column permutations} \longrightarrow} \\
 & \overset{*}{\begin{pmatrix}0&1&0\\0&2&3\\1&0&1\end{pmatrix}} & \overset{\star}{\begin{pmatrix}0&0&1\\0&3&2\\1&1&0\end{pmatrix}} & \begin{pmatrix}1&0&0\\2&0&3\\0&1&1\end{pmatrix} & \begin{pmatrix}1&0&0\\2&3&0\\0&1&1\end{pmatrix} & \begin{pmatrix}0&0&1\\3&0&2\\1&1&0\end{pmatrix} & \begin{pmatrix}0&1&0\\3&2&0\\1&0&1\end{pmatrix} \\
 & \overset{\dagger}{\begin{pmatrix}0&1&0\\1&0&1\\0&2&3\end{pmatrix}} & \overset{\bullet}{\begin{pmatrix}0&0&1\\1&1&0\\0&3&2\end{pmatrix}} & \begin{pmatrix}1&0&0\\0&1&1\\2&0&3\end{pmatrix} & \begin{pmatrix}1&0&0\\0&1&1\\2&3&0\end{pmatrix} & \begin{pmatrix}0&0&1\\1&1&0\\3&0&2\end{pmatrix} & \begin{pmatrix}0&1&0\\1&0&1\\3&2&0\end{pmatrix} \\
\downarrow & & & & & & \\
\text{row} & \begin{pmatrix}0&2&3\\0&1&0\\1&0&1\end{pmatrix} & \begin{pmatrix}0&3&2\\0&0&1\\1&1&0\end{pmatrix} & \begin{pmatrix}2&0&3\\1&0&0\\0&1&1\end{pmatrix} & \begin{pmatrix}2&3&0\\1&0&0\\0&1&1\end{pmatrix} & \begin{pmatrix}3&0&2\\0&0&1\\1&1&0\end{pmatrix} & \begin{pmatrix}3&2&0\\0&1&0\\1&0&1\end{pmatrix} \\
\text{permu-} & & & & & & \\
\text{tations} & \begin{pmatrix}0&2&3\\1&0&1\\0&1&0\end{pmatrix} & \begin{pmatrix}0&3&2\\1&1&0\\0&0&1\end{pmatrix} & \begin{pmatrix}2&0&3\\0&1&1\\1&0&0\end{pmatrix} & \begin{pmatrix}2&3&0\\0&1&1\\1&0&0\end{pmatrix} & \begin{pmatrix}3&0&2\\1&1&0\\0&0&1\end{pmatrix} & \begin{pmatrix}3&2&0\\1&0&1\\0&1&0\end{pmatrix} \\
\downarrow & & & & & & \\
 & \begin{pmatrix}1&0&1\\0&1&0\\0&2&3\end{pmatrix} & \begin{pmatrix}1&1&0\\0&0&1\\0&3&2\end{pmatrix} & \overset{\star}{\begin{pmatrix}0&1&1\\1&0&0\\2&0&3\end{pmatrix}} & \begin{pmatrix}0&1&1\\1&0&0\\2&3&0\end{pmatrix} & \begin{pmatrix}1&1&0\\0&0&1\\3&0&2\end{pmatrix} & \begin{pmatrix}1&0&1\\0&1&0\\3&2&0\end{pmatrix} \\
 & \begin{pmatrix}1&0&1\\0&2&3\\0&1&0\end{pmatrix} & \begin{pmatrix}1&1&0\\0&3&2\\0&0&1\end{pmatrix} & \begin{pmatrix}1&0&1\\3&2&0\\0&1&0\end{pmatrix} & \begin{pmatrix}1&1&0\\3&0&2\\0&0&1\end{pmatrix} & \begin{pmatrix}0&1&1\\2&0&3\\1&0&0\end{pmatrix} & \begin{pmatrix}0&1&1\\2&3&0\\1&0&0\end{pmatrix}
\end{array}
$$

Fig. 1. A set of symmetric assignments.

Given an assignment to an $n \times m$ matrix with row and column symmetry, we can permute its rows and columns in $n!m!$ different ways, and obtain a set of symmetric assignments. We show an example of this in Fig. 1 on an assignment:

$$
\begin{pmatrix} 0 & 1 & 0 \\ 0 & 2 & 3 \\ 1 & 0 & 1 \end{pmatrix}
$$

to a 3×3 matrix of variables. By permuting its rows and columns, we obtain $3!3! - 1 = 35$ more assignments which are symmetric to the original assignment. These 36 assignments form an equivalence class.

The assignments in an equivalence class are indistinguishable from each other in terms of satisfiability. One of them is a solution iff the remaining as-

0921-7126/04/$17.00

signments are all solutions. In search for solutions, we want to discard the symmetric assignments. So, how can we distinguish between these indistinguishable assignments? As the rows and columns of a matrix are vectors, we can characterise the assignments according to how their row and column vectors are ordered. One ordering of vectors is lexicographic ordering, which is also used to order the words in a dictionary. A vector $\vec{x} = \langle x_0, \ldots, x_{n-1} \rangle$ is lexicographically less than a vector $\vec{y} = \langle y_0, \ldots, y_{n-1} \rangle$ iff there is an index k above which the subvectors are equal, and $x_k < y_k$. For instance, $\langle 0, 2, 3 \rangle$ is less than $\langle 0, 3, 1 \rangle$ in the lexicographic order. We say that the rows (resp. columns) of a matrix of values are lexicographically ordered if each row (resp. column) is no greater than the rows (resp. columns) below (resp. to the right of) it.

Lexicographic ordering is very focused on positions and it ignores values beneath the position where the vectors differ. Multiset ordering, on the other hand, ignores positions but focuses on values. A multiset is a set in which repetition is allowed. A multiset $\{\!\{\mathbf{x}\}\!\}$ is less than a multiset $\{\!\{\mathbf{y}\}\!\}$ iff the largest value in $\{\!\{\mathbf{x}\}\!\}$ is less than the largest value in $\{\!\{\mathbf{y}\}\!\}$, or the largest values are the same and, if we eliminate one occurrence of the largest value from both $\{\!\{\mathbf{x}\}\!\}$ and $\{\!\{\mathbf{y}\}\!\}$, the resulting two multisets are ordered. For instance, $\{\!\{1, 1, 1\}\!\}$ is less than $\{\!\{0, 0, 2\}\!\}$ in the multiset order. Even though the rows and columns of a matrix are vectors, it may be useful to ignore the positions but rather concentrate on the values by treating the vectors as multisets. We say that the rows (resp. columns) of a matrix of values are multiset ordered if each row (resp. column), as a multiset, is no greater than the rows (resp. columns) below (resp. to the right of) it.

Let us now analyse the assignments of the equivalence class shown in Figure 1. There are exactly two assignments in this class, marked as $\star$, where the rows and columns are lexicographically ordered. On the other hand, there is exactly one assignment, marked as $\dagger$, where the rows and columns are multiset ordered. Similarly, there is exactly one assignment, marked as $*$, where the rows are lexicographically ordered and the columns are multiset ordered; and exactly one assignment, marked as $\bullet$, where the rows are multiset ordered and the columns are lexicographically ordered. Hence, we are able to distinguish the assignments marked as $\star$, $\dagger$, $*$, and $\bullet$ from the rest of the assignments.

An important result in this dissertation is that in any equivalence class of assignments, there is at least one assignment satisfying the properties of those marked as $\star$, $\dagger$, $*$, or $\bullet$ in Figure 1 [3,6]. Consequently, we can add extra constraints to the model of our problem [9] which enforce such orderings of the rows and the columns. In this way, among the set of symmetric assignments, only those satisfying the ordering constraints are chosen for consideration during search. These constraints can also be used to deal with matrices of arbitrary dimension, partial symmetries, and value symmetry.

To propagate the ordering constraints effectively and efficiently, we design global constraints. Propagating a constraint involves removing inconsistent values from the domains of its variables (i.e. pruning). Due to transitivity property of ordering relations, we focus on propagating the ordering constraint posted on a pair of vectors. We devise efficient linear time propagation algorithms for the lexicographic ordering [5] and the multiset ordering constraints [6]. We provide theoretical and experimental evidence of the value of the algorithms.

By adding ordering constraints to the matrix models, we identify a new pattern in constraint programs: the lexicographic ordering constraint on a pair of vectors of 0/1 variables together with a sum constraint on each vector. We frequently encounter this pattern in problems involving demand, capacity or partitioning that are modelled using matrices with row and/or column symmetry. This motivates us to introduce a new global constraint which combines the lexicographic ordering constraint with two sum constraints. We devise an efficient linear time algorithm to propagate this combination of constraints [8]. Our experimental results show that this new constraint is very useful when the lexicographic ordering constraints conflict with the way we explore the search space. Combining constraints is a step towards tackling one of the drawbacks of using additional constraints to break symmetry.

As theory can only go part of the way in judging the effectiveness of these ordering constraints in breaking row and column symmetries, we finish our research with an empirical study. We perform a wide range of experiments using some of the matrix models we have studied. In each experiment, we have a matrix of decision variables where the rows and/or columns are (partially) symmetric. To break the symmetry, we post ordering constraints on the rows and/or columns, and search for one solution or the best solution according to some criterion. Our results show that these ordering constraints are effective in breaking row and column symmetries as they significantly reduce the size of the search space and the time to solve the problems.

This dissertation is publicly available at the address publications.uu.se/theses/abstract.xsql?dbid=3991.

I am very grateful to my advisors Toby Walsh and Andreas Hamfelt for their supervision and support.

References

[1] R. Backofen and S. Will, Excluding symmetries in constraint-based search, in: *Proc. of CP-99*, J. Jaffar, ed., LNCS 1713, Springer, 1999.

[2] T. Fahle, S. Schamberger and M. Sellmann, Symmetry breaking, in: *Proc. of CP-01*, T. Walsh, ed., LNCS 2239, Springer, 2001.

[3] P. Flener, A.M. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, J. Pearson and T. Walsh, Breaking row and column symmetries in matrix models, in: *Proc. of CP-02*, P. van Hentenryck, ed., LNCS 2470, Springer, 2002.

[4] F. Focacci and M. Milano, Global cut framework for removing symmetries, in: *Proc. of CP-01*, T. Walsh, ed., LNCS 2239, Springer, 2001.

[5] A.M. Frisch, B. Hnich, Z. Kiziltan, I. Miguel and T. Walsh, Global constraints for lexicographic orderings, in: *Proc. of CP-02*, P. van Hentenryck, ed., LNCS 2470, Springer, 2002.

[6] A.M. Frisch, B. Hnich, Z. Kiziltan, I. Miguel and T. Walsh, Multiset ordering constraints, in: *Proc. of IJCAI-03*, G. Gottlob and T. Walsh, eds, Academic Press, 2003.

[7] I.P. Gent and B.M. Smith, Symmetry breaking in constraint programming, in: *Proc. of ECAI-00*, W. Horn, ed., IOS Press, 2000.

[8] B. Hnich, Z. Kiziltan and T. Walsh, Combining symmetry breaking with other constraints: lexicographic ordering with sums, in: *Proc. of AMAI-04*, F. Bacchus and P. van Beek, eds, 2004.

[9] J.F. Puget, On the satisfiability of symmetrical constrained satisfaction problems, in: *Proc. of ISMIS-93*, H.J. Komorowski and Z.W. Ras, eds, LNCS 689, Springer, 1993.