Expressing Symmetry Breaking Constraints Using Multiple Viewpoints and Channeling Constraints

Y.C. Law and J.H.M. Lee

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{yclaw,jlee}@cse.cuhk.edu.hk

Abstract. Symmetries have adverse effects on the CSP solving process, because symmetrically equivalent regions of the search tree may be traversed more than once. Symmetry breaking can reduce the search space of the problem and is beneficial. In this paper, we focus on using symmetry breaking constraints statically to remove symmetries in CSPs. In particular, we formalize the ideas of two types of symmetries in CSPs, namely variable symmetries and value symmetries. The former type is easier to express with symmetry breaking constraints than the latter in general. Therefore, we introduce a general principle in devising another viewpoint from a given viewpoint and the two viewpoints are connected together using channeling constraints. By using a second viewpoint, we can transform value symmetries in the original viewpoint into variable symmetries in another so that we can express symmetry breaking constraints more easily. We demonstrate our approach using the social golfer problem by building several models which use multiple viewpoints to break different types of symmetries of the problem. Since we can express symmetry breaking constraints more succinctly in another viewpoint than in the original one, this usually leads to better constraint propagation and fewer total number of constraints, which are the potential sources of speedup. Experimental results confirm that models using multiple viewpoints for symmetry breaking exhibits extra efficiency than those using only single viewpoint.

1 Introduction

The task at hand is to tackle *constraint satisfaction problems* (CSPs) [16], which are, in general, NP-complete. A recent important line of research in the community is to investigate symmetries in CSPs. Symmetries in CSPs are mappings from solutions to solutions, and also non-solutions to non-solutions. They have adverse effects on the CSP solving process, because symmetrically equivalent regions of the search tree may be traversed more than once. Symmetry breaking is beneficial because it can reduce the search space of the CSPs.

There are two main types of methods in breaking symmetries in CSPs. The first approach breaks symmetries *statically* [18]. Symmetry breaking constraints are added to a CSP to make all but one of the symmetrical regions violate the constraints. This approach reformulates the CSP before search to reduce the CSP's initial search space. Symmetrical solutions are removed in the reformulated CSP and only non-symmetric solutions remains. Other examples of this approach include the lexicographic ordering constraints [7] for breaking row and column symmetries in matrix models [6]. The constraints help breaking symmetries of indistinguishable objects, each of which represented by more than one variable in the CSP.

The second approach breaks symmetries *dynamically* [9, 1]. Search algorithms for solving CSPs are modified such that symmetric states are pruned from the search tree as it develops. Examples of such approach are *Symmetry Breaking During Search* (SBDS) [9, 8] and *Symmetry Breaking via Dominance Detection* (SBDD) [4, 19, 2]. Upon backtracking of the search, SBDS adds symmetry breaking constraints to the CSP to remove all the states which are symmetric to the one that causes backtracking. In SBDD, whenever the search algorithm generates a new search node, we check whether it is dominated by another node previously visited. If this is the case, the current search node can be pruned.

Cheng *et al.* [3] introduces redundant modeling, in which two models of the same problem are combined together using channeling constraints. They show increased constraint propagation and efficiency by using this approach. Channeling constraints are used to connect two viewpoints together. In this paper, we focus on expressing static symmetry breaking constraints using multiple viewpoints and channeling constraints. In particular, we formalize the ideas of two types of symmetries in CSPs, namely variable symmetries and value symmetries. The former type is easier to express with symmetry breaking constraints than the latter in general. We introduce a general principle in devising another viewpoint from a given one and the two viewpoints are connected together using channeling constraints. By using a second viewpoint, we can transform value symmetries in the original viewpoint into variable symmetries in another so that we can express symmetry breaking constraints more easily. We demonstrate our approach using the social golfer problem by building several models which use multiple viewpoints to break different types of symmetries of the problem. Since we can express symmetry breaking constraints more succinctly in another viewpoint than in the original one, this usually leads to better constraint propagation and fewer total number of constraints, which are the potential sources of speedup. Experimental results that models using multiple viewpoints for symmetry breaking exhibits extra efficiency than those using only single viewpoint.

The rest of the paper is organized as follows. Section 2 provides the necessary background to the paper. We give definitions to concepts ranging from viewpoints to CSP models. Section 3 introduce our method of transforming value symmetries into variable symmetries. We also give definitions of variable and value symmetries, and a general principle in devising another viewpoint from a given one in the section. We present the experimental results using the social golfer problem in Section 4. Finally, in Section 5, a summary of our ideas and possible directions of future work are presented.

2 Background

There are usually more than one way of formulating a problem P into a CSP. Central to the formulation process is to determine the variables and the domains (associated sets of possible values) of the variables. Different choices of variables and domains are results of viewing the problem P from different angles/perspectives. Following the definition of Law and Lee [15], we define a *viewpoint* to be a pair (X, D_X), where $X = \{x_1, \ldots, x_n\}$ is a set of variables, and D_X is a set containing, for every $x \in X$, an associated domain $D_X(x)$ giving the set of possible values for x.

A viewpoint $V = (X, D_X)$ defines the possible assignments for variables in X. An *assignment* $x \mapsto a$ in V (or in $U \subseteq X$) means that variable $x \in X$ (or U) is assigned the value $a \in D_X(x)$. We overload the $\mapsto$ operator to accept assignments of a set of variables $\{x_{i_1}, \ldots, x_{i_k}\}$ such that $[x_{i_1}, \ldots, x_{i_k}] \mapsto [v_1, \ldots, v_k]$ means $\{x_{i_j} \mapsto v_j | 1 \leq j \leq k\}$.

When formulating a problem P into a CSP, the choice of viewpoints is not arbitrary. Suppose $sol(P)$ is the set of all solutions of P (in whatever notations and formalism). We say that viewpoint V is *proper* [15] for P if and only if we can find a subset S of the set of all possible complete assignments in V so that there is a one-one mapping between S and $sol(P)$. In other words, each solution of P must correspond to a distinct complete assignment in V.

A *CSP model* M (or simply *model* or *CSP* hereafter) of a problem P is a pair (V, C), where V is a proper viewpoint of P and C is a set of constraints in V for P. A *constraint* can be considered a predicate among a subset of the variables in V that maps to $true$ or $false$. A *solution* of $M = (V, C)$ is a set of assignments of all variables in V such that the assignments maps all the constraints to $true$, i.e., all the constraints are satisfied.

3 Symmetry Breaking via Channeling

In this section, we introduce a method to break symmetries in CSPs using multiple viewpoints [15, 14] of a problem and channeling constraints. In the following subsections, we first formally define two types of symmetries in CSPs, namely variable symmetry and value symmetry. Then, we describe a general principle in devising another viewpoint from a given one. The general form of channeling constraints to connect the two viewpoints are also given. Finally, we present our method of transforming value symmetry into variable symmetry so that we can express symmetry breaking constraints more easily.

3.1 Types of Symmetry

In this subsection, we define variable symmetry and value symmetry in CSPs. Given a solution, variable symmetry allows swapping the values assigned to some variables to obtain another one, while value symmetry allows permutating some domain values on a subset of variables to obtain another solution.

Variable Symmetry A variable symmetry [5] of a CSP is a bijective mapping of the set of variables X to itself, $\sigma : X \rightarrow X$, which maps solutions of the CSP to solutions and non-solutions to non-solutions. That means two (or more) indistinguishable objects in the problem appear as variables in the CSP. Variable symmetry can be found in many problems which have symmetries. The social golfer problem, "prob010" in CSPLib [10], is an example.

> *32 social golfers play golf once a week, and always in groups of 4. No golfer can play in the same group as any other golfer on more than one occasion. How many weeks can the golfers play for?*

The problem can be generalized to that of finding a w-week schedule of g groups, each of which contains s golfers, such that no two golfers can play together more than once. Therefore, the total number of golfers is $n = g \times s$. We denote an instance of the problem as (g, s, w).

The social golfer problem is highly symmetric [4, 2]:

1. Players can be exchanged inside groups.
2. Groups can be exchanged inside weeks.
3. Weeks of schedule can be exchanged.
4. Players can be permutated among the $n!$ combinations.

To model the social golfer problem into a CSP, consider the viewpoint $V_1 = (X, D_X)$ which contains a variable $p_{i,k}$ for each golfer i in week k with $0 \leq i < n$ and $0 \leq k < w$. The domain of the variables $D_X(p_{i,k}) = \{0, \ldots, g-1\}$ contains the group numbers that a golfer can play in. Using viewpoint V_1, the constraints of the problem can be expressed accordingly. Fig. 1 shows a solution of the $(3, 3, 3)$ instance.

week \ golfer	0	1	2	3	4	5	6	7	8
0	0	0	0	1	1	1	2	2	2
1	0	1	2	0	1	2	0	1	2
2	0	1	2	1	2	0	2	0	1

Fig. 1. A Solution of the $(3, 3, 3)$ Instance of the Social Golfer Problem

Symmetry 1 is broken implicitly using V_1 because we simply use a group number to represent a group and do not distinguish different positions within a group. Symmetry 4, however, exists in V_1, because all the golfers are indistinguishable objects, but we name the golfers from 0 to $n-1$ and use different variables for them. Given a solution of the problem, we can always exchange the values assigned to any two golfers and yet obtain another solution. Since the golfers appear as variables in V_1, such symmetry is variable symmetry. For example, consider the solution in Fig. 1. For golfers 0 and 1, we have $[p_{0,0}, p_{0,1}, p_{0,2}] \mapsto [0, 0, 0]$ and $[p_{1,0}, p_{1,1}, p_{1,2}] \mapsto [0, 1, 1]$. We can exchange the values assigned to golfers 0 and 1 and obtain another solution with

$[p_{0,0}, p_{0,1}, p_{0,2}] \mapsto [0,1,1]$ and $[p_{1,0}, p_{1,1}, p_{1,2}] \mapsto [0,0,0]$. In this example, we have the bijective mapping σ_1 as the identity mapping except $\sigma_1(p_{0,k}) = p_{1,k}$ and $\sigma_1(p_{1,k}) = p_{0,k}$ with $k = 0, 1, 2$.

Symmetry 3 is also variable symmetry in V_1. In the problem, the weeks are indistinguishable, but we name the weeks from 0 to $w-1$ and use different variables for the weeks. Given a solution, we can again exchange the values assigned to any two weeks of golfers and obtain another solution. Since the weeks appear as variables in V_1, symmetry 3 is also variable symmetry. For example, in the solution in Fig. 1, we have $[p_{0,0}, \ldots, p_{8,0}] \mapsto [0,0,0,1,1,1,2,2,2]$ and $[p_{0,1}, \ldots, p_{8,1}] \mapsto [0,1,2,0,1,2,0,1,2]$. By exchanging the values assigned to the variables of week 0 and 1, we obtain another solution with $[p_{0,0}, \ldots, p_{8,0}] \mapsto [0,1,2,0,1,2,0,1,2]$ and $[p_{0,1}, \ldots, p_{8,1}] \mapsto [0,0,0,1,1,1,2,2,2]$. In this example, we have another bijective mapping σ_2 which is the identity mapping except $\sigma_2(p_{i,0}) = p_{i,1}$ and $\sigma_2(p_{i,1}) = p_{i,0}$ for all $0 \leq i < 9$.

Variable symmetry in CSPs can be broken by introducing symmetry breaking constraints to impose an ordering on the indistinguishable variables. In the above example of the social golfer problem, we can break symmetry 4 of golfers 0 and 1 by imposing an ordering to restrict that the values assigned to golfer 0 must be *lexicographically smaller* [6] than that assigned to golfer 1, i.e., $[p_{0,0}, p_{0,1}, p_{0,2}] <_{lex} [p_{1,0}, p_{1,1}, p_{1,2}]$. In general, the symmetry breaking constraints are $[p_{i,0}, \ldots, p_{i,w-1}] <_{lex} [p_{i+1,0}, \ldots, p_{i+1,w-1}]$ for all $0 \leq i < n-1$. Similarly, we can break symmetry 3 of weeks 0 and 1 by the symmetry breaking constraints $[p_{0,0}, \ldots, p_{8,0}] <_{lex} [p_{0,1}, \ldots, p_{8,1}]$, and in general $[p_{0,k}, \ldots, p_{n-1,k}] <_{lex} [p_{0,k+1}, \ldots, p_{n-1,k+1}]$ for all $0 \leq k < w-1$. The above two kinds of symmetry breaking constraints correspond to breaking the row and column symmetries in matrix models [6]. Flener *et al.* [6] show that in a matrix model with row and column symmetries, while lexicographically ordering all the rows (columns) breaks all row (column) symmetries, lexicographically ordering both the row and columns fails to break all the compositions of the row and column symmetries. However, there is always a solution with the rows and columns both lexicographically ordered. The solution in Fig. 1 has all the rows and columns lexicographically ordered. Exchanging any two rows or columns breaks the lexicographic ordering constraints. Therefore, the symmetric solution after exchanging will not be accepted as a solution of the resultant CSP.

Value Symmetry Another type of symmetry in CSPs is value symmetry [6]. A value symmetry acts on a subset $X' \subseteq X$ of the variables in the viewpoint (X, D_X) of a CSP where $D_X(x) = D_X(x')$ for all $x, x' \in X'$. It is a bijective mapping on the set of domain values, $\sigma : D_X(x) \to D_X(x)$ where $x \in X'$, which maps solutions of the CSP to solutions and non-solutions to non-solutions. That means two (or more) indistinguishable objects in the problem appear as domain values in the CSP.

Value symmetries, alongside with variable symmetry, is also very common in CSPs with symmetries. In the social golfer problem, symmetry 2 is an example of value symmetry in V_1. Given a solution, consider the set of variables $X' =$

$\{p_{0,0}, \ldots, p_{n-1,0}\} \subseteq X$ representing all the golfers in week 0. We can permutate all the values assigned to X' from 0 to 1, from 1 to 2, and from 2 to 0 to obtain another solution symmetric to the original one. For example, consider the solution in Fig. 1, we have $[p_{0,0}, \ldots, p_{8,0}] \mapsto [0,0,0,1,1,1,2,2,2]$. If we follow the above mapping, we obtain another solution with $[p_{0,0}, \ldots, p_{8,0}] \mapsto [1,1,1,2,2,2,0,0,0]$.

Value symmetry is difficult to tackle using symmetry breaking constraints in general, because we do not know beforehand which variable will be assigned what particular value. Therefore, we do not know how an ordering can be imposed. In practice, value symmetries are handled by pre-assigning the affected variables as far as possible with some values without loss of generality. These pre-assignments must be able to be extended to solutions. However, whenever we cannot pre-assign some variables (which is usually the case), there is the chance of not breaking part of the value symmetries but wasting search efforts. For example, in the social golfer problem, we can always pre-assign, without loss of generality, the variables $[p_{0,0}, \ldots, p_{n-1,0}] \mapsto [\underbrace{0, \ldots, 0}_{s}, \underbrace{1, \ldots, 1}_{s}, \ldots, \underbrace{g-1, \ldots, g-1}_{s}]$ and $[p_{0,k}, \ldots, p_{s-1,k}] \mapsto [0, \ldots, s-1]$ for all $k \geq 1$. The former breaks the value symmetries for week 0, and the latter breaks part of the value symmetries in week 1 and so on. While it is possible to pre-assign variables to break value symmetries of values 0 to $s-1$ starting from week 1, the value symmetries of values s to $g-1$ remains unbroken. Therefore, fewer value symmetries can be broken when $s \ll g$.

Sometimes, it is possible to express symmetry breaking constraints, although unnaturally, for value symmetries as well. Consider the value symmetries in V_1 of the social golfer problem. We devise the following symmetry breaking constraints to break the value symmetries:

$$p_{i,k} = j \rightarrow \bigvee_{i'=0}^{i-1} p_{i',k} = j'$$

for all $0 \leq i < n$, $0 \leq j' < j < g$, and $0 \leq k < w$. The constraints are unnatural because they make use of disjunctions, which are handled less efficiently in many CSP solvers. Besides, the number of constraints is large. We need a total of $O(ng^2w) = O(g^3sw)$ constraints to break the value symmetries, which can impose a large overhead for solving.

3.2 Multiple Viewpoints and Channeling Constraints

In this subsection, we describe a general principle in devising another viewpoint with a given one. We also give the general form of channeling constraints for connecting the two viewpoints together.

There are usually more than one way of formulating a problem into a CSP. The formulation process is to determine the variables and the domains of the variables. We observe that in many problems, we can devise two different viewpoints which are reciprocal of each other in the sense that in one viewpoint, objects of type X is assigned to objects of type Y, while in another viewpoint, objects of type Y is assigned to objects of type X. For example, in a generic job-shop scheduling problem, we can either assign jobs to machines or assign machines to jobs. Note that the objects appear as different roles in the two viewpoints. An object which appears as a variable in one viewpoint becomes a domain value in the other viewpoint, and one which appears as a domain value in one viewpoint becomes a variable in another.

Given two models $M_1 = ((X, D_X), C_X)$ and $M_2 = ((Y, D_Y), C_Y)$. Cheng *et al.* [3] define a *channeling constraint* c to be a constraint that relates variables not just in X, in Y, but in $X \cup Y$. Thus, c relates M_1 and M_2 by limiting the combination of values that their variables can take. Cheng *et al.* show how a collection of channeling constraints can be used to connect two mutually redundant models of the same problem to form a combined model, which exhibits increased constraint propagation and thus improved efficiency. We note in the definition of channeling constraints that the constraints in the two models are immaterial. Channeling constraints relate actually the viewpoints of the models. In other words, channeling constraints set forth a relationship between the possible assignments of the two viewpoints.

The channeling constraints for connecting two mutually reciprocal viewpoints can be stated generally. Suppose there are two objects in a problem and objects 1 and 2 is paired in a solution. Object 1 is associated with variable x in the first viewpoint, and with value v in the second viewpoint. Object 2 is associated with value u in the first viewpoint, and with variable y in the second. The channeling constraints for connecting the two viewpoints is then in the form:

Variable x has value u if and only if variable y has value v.

Note that we cannot simply state $x = u$ or $y = v$, because x and y can be set variables [11, 12], which can be assigned multiple values instead of one value in integer variables.

Take again the social golfer problem as an example. In viewpoint V_1, we are assigning groups to the golfers. The groups appear as values and the golfers appear as variables. We can devise another viewpoint V_2 in which golfers are to be assigned to groups. In such case, the groups appear as variables and the golfers appear as values in V_2. In viewpoint V_2, we use variables $G_{j,k}$ for each group j in week k. Since a group contains multiple golfers, the variables $G_{j,k}$ are set variables and their domains are the set of all possible sets of the n players, i.e., $D_Y(G_{j,k}) = 2^{\{0,\ldots,n-1\}}$, where 2^X is the power set of X. Fig. 2 shows the same solution as in Fig. 1, but expressed in V_2.

group \ week	0	1	2
0	{0,1,2}	{0,3,6}	{0,5,7}
1	{3,4,5}	{1,4,7}	{1,3,8}
2	{6,7,8}	{2,5,8}	{2,4,6}

Fig. 2. A Solution of the $(3, 3, 3)$ Instance, Expressed in V_2

In order to channel the viewpoints V_1 and V_2, we need to specify the channeling constraints. In this example, the channeling constraints are $p_{i,k} = j \Leftrightarrow i \in G_{j,k}$ for all $0 \leq i < n$, $0 \leq j < g$, and $0 \leq k < w$. Note that we use the membership operator "$\in$" for the set variables. The channeling constraints are of the same form as in the general case we state before.

3.3 Transforming Value Symmetry into Variable Symmetry

In this subsection, we exploit another use of channeling constraints and model channeling, which is to help breaking value symmetries in CSPs. In particular, we transform value symmetries of a CSP into variable symmetries, so that symmetry breaking constraints can be expressed more easily.

In the previous subsection, we introduce a general principle in coming up with a second viewpoint. Domain values in the given viewpoint become variables in the other viewpoint. Value symmetry in the given viewpoint also becomes variable symmetry in the other. Recall that variable symmetries are easier to express with symmetry breaking constraints than value symmetries. It is a good idea to transform value symmetries in one viewpoint into variable symmetries in another. Suppose we are given a viewpoint V_1, and V_2 is another viewpoint where domain values of the variables in V_1 become variables in V_2. By using channeling, we connect the two viewpoints V_1 and V_2 together. The original symmetry breaking constraints for the variable symmetries in V_1 can be used as usual. The value symmetries in V_1 can be tackled by symmetry breaking constraints for breaking the corresponding variable symmetries in V_2. Therefore, the two types of symmetries can be broken by symmetry breaking constraints altogether. Note that the viewpoint V_2 is solely used for expressing the symmetry breaking constraints for the value symmetries in V_1. We need not express the problem constraints there because they are expressed in V_1 already.

Consider symmetry 2 in the social golfer problem again, which is a value symmetry in V_1. The groups inside a week are indistinguishable objects. In V_2, we are using a set variable $G_{j,k}$ for each group j in week k. Therefore, symmetry 2 is a variable symmetry in V_2. To break the symmetry, we can impose an order on the groups in each week. Since for each week, each of the n golfers must be assigned to one of the g groups once, the groups in a week must be disjoint sets of each other. Therefore, we can use the smallest numbered golfer in each group as a key for ordering the groups. The symmetry breaking constraints are then $\min G_{j,k} < \min G_{j+1,k}$ for $0 \leq j < g-1$ and $1 \leq k < w$. Using these constraints and the channeling constraints, the value symmetry in V_1 can be broken as variable symmetry in V_2. Fig. 2 shows a solution of the $(3, 3, 3)$ instance which satisfies the symmetry breaking constraints in V_2.

To further illustrate our ideas, we give another example of transforming value symmetries into variable symmetries for the social golfer problem using a third viewpoint V_3. In V_3, we use a variable $z_{i,k,j}$ for each golfer i in group j of week k. The domain of the variables $z_{i,k,j}$ is $\{0, 1\}$. Variable $z_{i,k,j}$ takes value 1 if golfer i is in group j of week k, and value 0 otherwise. The channeling constraints for connecting V_1 and V_3 can be expressed as $p_{i,k} = j \Leftrightarrow z_{i,k,j} = 1$ for all $0 \leq i < n$, $0 \leq j < g$, and $0 \leq k < w$.

Symmetry 2 of the problem becomes variable symmetries in V_3. We can use the symmetry breaking constraints $[z_{0,k,j}, \ldots, z_{n-1,k,j}] >_{lex} [z_{0,k,j+1}, \ldots, z_{n-1,k,j+1}]$ for all $0 \leq j < g-1$ and $0 \leq k < w$ to tackle such symmetries. Note that we order the array of variables in decreasing order instead of increasing order. This is because the latter conflicts with the other lexicographic ordering constraints in V_1. For example, consider the ordering constraint $[p_{0,0}, p_{0,1}, p_{0,2}] <_{lex} [p_{1,0}, p_{1,1}, p_{1,2}]$ in V_1 of the $(3,3,3)$ instance. Suppose $[p_{0,0}, p_{0,1}, p_{0,2}] \mapsto [0,0,0]$ and $[p_{1,0}, p_{1,1}, p_{1,2}] \mapsto [0,1,1]$ (as in the solution in Fig. 1). These assignments correspond to $\boldsymbol{z}_0 \mapsto [1,0,0,1,0,0,1,0,0]$ and $\boldsymbol{z}_1 \mapsto [1,0,0,0,1,0,0,1,0]$ respectively, where $\boldsymbol{z}_0 = [z_{0,0,0}, z_{0,0,1}, z_{0,0,2}, z_{0,1,0}, z_{0,1,1}, z_{0,1,2}, z_{0,2,0}, z_{0,2,1}, z_{0,2,2}]$ and $\boldsymbol{z}_1 = [z_{1,0,0}, z_{1,0,1}, z_{1,0,2}, z_{1,1,0}, z_{1,1,1}, z_{1,1,2}, z_{1,2,0}, z_{1,2,1}, z_{1,2,2}]$. Note that the former is lexicographically larger than the latter, i.e., $\boldsymbol{z}_0 >_{lex} \boldsymbol{z}_1$. Therefore, the row ordering constraints in V_1 impose a decreasing order on the array of variables in V_3. The situation is similar for the column ordering constraints in V_1. Hence, we have to impose a decreasing order instead of an increasing order for tackling symmetry 2 in V_3.

Flener *et al.* [6] suggest that it is always possible to transform a viewpoint with variable and value symmetries into another viewpoint that only contains variable symmetries. They suggest transforming a n-dimensional matrix of variables into a $(n+1)$-dimensional matrix where the $(n+1)$-st dimension corresponds to the domain values of the original matrix variables. For example, if x is a variable in the original matrix having domain $\{v_0 \ldots, v_{m-1}\}$, we introduce variables $[x_0, \ldots, x_{m-1}]$ in the new viewpoint. The new variables have domain $\{0,1\}$, with the semantics $x = v_i \Leftrightarrow x_i = 1$. In the transformed viewpoint, there are no value symmetries anymore because 0 and 1 are clearly distinguishable. Symmetry breaking constraints are then expressed in the new viewpoint to break all kinds of symmetries of the problem. The viewpoint V_3 in the social golfer problem can be viewed as a transformed viewpoint of V_1 with an extra dimension on the groups. We extend Flener *et al.*'s suggestion by allowing multiple viewpoints coexist for symmetry breaking. Different symmetry breaking constraints can be expressed in different viewpoints, whichever is easier to express. As we shall see, expressing symmetry breaking constraints in multiple viewpoints is more beneficial than that in a single viewpoint.

When transforming value symmetry in one viewpoint into variable symmetry in another, we should be careful that the symmetry breaking constraints expressed in V_2 does not conflict with those expressed in V_1. That means in each set of mutually symmetric solutions of a problem, there should be at least one element satisfying all the symmetry breaking constraints. If all the elements in the set violate some of the symmetry breaking constraints, we lose a unique solution of the problem and this is not desired. Ideally, there should be exactly one element in each set satisfying the symmetry breaking constraints, so that all the symmetries are broken and all the solutions of the resultant model are unique ones of the problem. However, stating symmetry breaking constraints for breaking all symmetries in a CSP may be costly and the effort of propagating the constraints may outweigh the gain of reducing the search space. Hence, practically, partial symmetry breaking [17] is acceptable and popular. For example, the method used by Flener *et al.* [6] for breaking row and column symmetries in matrix models also falls into the partial symmetry breaking category.

4 Experimental Results

In this section, we present experimental results in supporting our idea using the social golfer problem. The experiments are run using ILOG Solver 4.4 [13] on a Sun Ultra 5/400 workstation with 256MB memory. We use the global constraint by Frisch *et al.* [7] for lexicographic ordering[1] which maintains *generalized arc consistency* (GAC) of the constraint.

In the previous section, we give two examples of the social golfer problem to break value symmetries in V_1 using viewpoints V_2 and V_3 and channeling constraints. We call the models using viewpoints V_1 and V_2 `golferV1V2` and V_1 and V_3 `golferV1V3` respectively in subsequent discussions.

Having two models `golferV1V2` and `golferV1V3` that use multiple viewpoints, we can also build models using a single viewpoint only. Note that besides symmetry 2, symmetries 3 and 4 also appear as variable symmetries in V_3. Therefore, it is possible to use V_3 alone to model the problem and also express all the symmetry breaking constraints. Although we impose a decreasing order for symmetry 2 in `golferV1V3`, we can still express the ordering constraints either all in increasing order or all in decreasing order when using V_3 alone. We call the models with symmetry breaking constraints in V_3 all in increasing order and all in decreasing order `golferV3` and `golferV3'` respectively.

Finally, we have another model that uses a single viewpoint V_1 only. The variable symmetries are broken using lexicographic ordering constraints as usual. The value symmetries are broken using the symmetry breaking constraints expressed directly in V_1, which are introduced in the previous section. We call the model using single viewpoint V_1 `golferV1`.

Table 1 shows the experimental results using the models `golferV1`, `golferV3`, `golferV3'`, `golferV1V2`, and `golferV1V3` of the social golfer problem. The top table shows results for solving for the first solution, while the bottom one shows results for solving for all solutions. We report the number of fails, number of choice points, and CPU time for each execution of each instance. The first three models use single viewpoint only, while the latter two use multiple viewpoints. A cell labeled with "-" means that execution does not terminate within 2 hours of CPU time. All models are solved using default variable and value ordering (i.e., smallest variable indices first and smallest value first) except `golferV3'`. The model `golferV3'` is solved using default variable ordering coupled with the value ordering heuristic of trying value 1 before 0 (i.e., largest value first). This configuration is used because it mimics the default variable and value ordering as

[1] We acknowledge The University of York for providing the source of the global constraint for lexicographic ordering for our reference.

in `golferV1`, `golferV1V2`, and `golferV1V3`. Trying 1 first for the variable with smallest indices in V_3 is equivalent to trying the smallest value for the variable with smallest indices in V_1. We do not use this configuration in `golferV3` because `golferV3` uses a different lexicographic ordering from `golferV3'`. There is no need to mimic the variable and value ordering in the other models.

Experimental results show that model `golferV1V3`, which uses multiple viewpoints, is the fastest model. Model `golferV1V2`, another model that uses multiple viewpoint, together with `golferV1` have the same and fewest number of fails and choice points. The figures are slightly smaller than those of `golferV1V3`. Although `golferV1` have the same number of fails and choice points as `golferV1V2`, its execution is much slower than that of `golferV1V2`. This is because the symmetry breaking constraints for the value symmetries in V_1 are expressed cumbersomely. There are $O(ng^2w)$ such symmetry breaking constraints in `golferV1`. However, in `golferV1V2`, we only need $O(gw)$ symmetry breaking constraints plus $O(ngw)$ channeling constraints for connecting V_1 and V_2. Thus, only a total of $O(gw + ngw) = O(ngw)$ constraints are required. Since `golferV1V2` uses fewer constraints than `golferV1` by an order, its execution is faster.

The models `golferV3` and `golferV3'` differs only in the increasing or decreasing ordering of the symmetry breaking constraints and the value ordering. However, `golferV3'` is more efficient than `golferV3`, which shows that ordering the arrays of variables in V_3 in decreasing order (and thus in V_1 in increasing order) is a better choice than that in increasing order.

Furthermore, `golferV1V3` and `golferV3'` follow the same lexicographic order and equivalent variable and value orderings. However, the former has fewer number of fails and choice points than the latter. This is because propagating a lexicographic ordering constraint in V_3 obtains less pruning information than in V_1. For example, consider the constraint $\boldsymbol{x}_0 <_{lex} \boldsymbol{x}_1$ on two arrays of variables $\boldsymbol{x}_0 = [x_{0,0}, x_{0,1}, x_{0,2}]$ and $\boldsymbol{x}_1 = [x_{1,0}, x_{1,1}, x_{1,2}]$ with the following domains:

$\boldsymbol{x}_0$	{1,2}	{0,1,2}	{2}
$\boldsymbol{x}_1$	{1}	{0,1,2}	{0,1}

Maintaining GAC on the constraint will prune values from the variable domains:

$\boldsymbol{x}_0$	{1}	{0,1}	{2}
$\boldsymbol{x}_1$	{1}	{1,2}	{0,1}

When using a 0/1 viewpoint, we use the variables $z_{i,k,j}$ for $i = 0, 1$, $0 \leq j < 3$, and $0 \leq k < 3$. Let $\boldsymbol{z}_0 = [z_{0,0,0}, z_{0,0,1}, z_{0,0,2}, z_{0,1,0}, z_{0,1,1}, z_{0,1,2}, z_{0,2,0}, z_{0,2,1}, z_{0,2,2}]$ and $\boldsymbol{z}_1 = [z_{1,0,0}, z_{1,0,1}, z_{1,0,2}, z_{1,1,0}, z_{1,1,1}, z_{1,1,2}, z_{1,2,0}, z_{1,2,1}, z_{1,2,2}]$. The initial domains of the $z_{i,k,j}$ variables become:

$\boldsymbol{z}_0$	{0}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0}	{0}	{1}
$\boldsymbol{z}_1$	{0}	{1}	{0}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0}

By propagating the constraint $\boldsymbol{z}_0 >_{lex} \boldsymbol{z}_1$ and the constraints $\sum \boldsymbol{z}_0 = 1$ and $\sum \boldsymbol{z}_1 = 1$ which ensure that only one of $\boldsymbol{z}_0$ and one of $\boldsymbol{z}_1$ can be 1, we obtain the following result:

Table 1. Experimental Results Using Various Models of the Social Golfer Problem

1st sol.	golferV1			golferV3			golferV3'			golferV1V2			golferV1V3		
g, s, w	fails	choices	time	fails	choices	time	fails	choices	time	fails	choices	time	fails	choices	time
6,2,11	**142**	**180**	0.57	166	425	0.32	166	259	0.32	**142**	**180**	0.2	166	259	**0.13**
7,2,13	**1371**	**1428**	9.63	1525	1966	2.97	1469	1604	3.08	**1371**	**1428**	1.94	1469	1604	**1.47**
6,3,5	-	-	-	4795797	4795983	3327.19	3776112	3776179	3072.41	**3759093**	**3759135**	3066.65	3763739	3763806	**2066.14**
7,3,4	**687**	**729**	3.33	853	1084	0.81	702	768	0.73	**687**	**729**	0.72	694	760	**0.41**
8,3,5	-	-	-	4332617	4332989	5451.74	1711560	1711655	2234.02	**1700505**	**1700565**	2060.44	1706740	1706835	**1330.29**
5,4,4	-	-	-	-	-	-	3497965	3498018	3823.11	**3472724**	**3472761**	3270.73	3478063	3478116	**2297.76**
6,4,3	-	-	-	-	-	-	6238656	6238711	5560.54	**5880269**	**5880309**	5868.36	5903505	5903560	**3730.93**
7,4,3	-	-	-	-	-	-	2281975	2282042	2485.94	**2070539**	**2070588**	2720.35	2094709	2094776	**1699.63**
8,4,9	**22**	**142**	7.36	27	668	1.52	24	207	1.33	**22**	**142**	0.44	24	207	**0.3**
7,5,2	**48794**	**48838**	1121.54	-	-	-	77674	77734	91.34	**48794**	**48838**	127.32	50257	50313	**78.34**
8,5,2	**71463**	**71515**	2691.18	-	-	-	112299	112369	165.61	**71463**	**71515**	213.19	74679	74745	**127.84**
8,8,9	**19**	**182**	41.78	**19**	821	7.59	**19**	245	7.45	**19**	**182**	1.75	**19**	245	**0.93**
5,4,3	**153032**	**153065**	352.82	496380	496484	435.32	160256	160301	126.05	**153032**	**153065**	134.66	153451	153496	**80.83**
5,3,5	**166062**	**166092**	215.26	196432	196547	111.43	166993	167043	102.77	**166062**	**166092**	94.64	166210	166260	**64.27**
5,3,7	**18416**	**18445**	46.76	53824	53954	57.12	18745	18802	23.72	**18416**	**18445**	19.53	18616	18673	**13.96**

all sol.	golferV1			golferV3			golferV3'			golferV1V2			golferV1V3		
g, s, w	fails	choices	time	fails	choices	time	fails	choices	time	fails	choices	time	fails	choices	time
4,2,4	**1283**	**2294**	0.76	1478	2489	0.83	1344	2355	0.6	**1283**	**2294**	0.77	1335	2346	**0.59**
4,2,5	**2779**	**4476**	1.77	2998	4695	1.53	2843	4540	1.55	**2779**	**4476**	1.93	2838	4535	**1.38**
4,2,6	**2368**	**3103**	1.53	2598	3333	1.3	2434	3169	1.41	**2368**	**3103**	1.51	2433	3168	**1.15**
4,2,7	**903**	**1034**	0.62	1145	1276	0.61	974	1105	0.57	**903**	**1034**	0.55	974	1105	**0.41**
4,3,3	**5486**	**5789**	3.55	13440	13743	5.98	5772	6075	2.39	**5486**	**5789**	2.58	5543	5846	**1.81**
4,3,4	**27547**	**27740**	19.16	64027	64220	30.42	27992	28185	13.45	**27547**	**27740**	15.13	27686	27879	**9.75**
4,4,3	**836**	**859**	1.25	27823	27846	22.45	853	876	0.62	**836**	**859**	0.63	853	876	**0.39**
4,4,4	**1084**	**1089**	1.79	106572	106577	88.99	1107	1112	0.91	**1084**	**1089**	0.89	1107	1112	**0.6**
4,4,5	**115**	**116**	0.28	40588	40589	37.33	133	134	0.2	**115**	**116**	0.15	133	134	**0.1**
5,5,4	**708404**	**708619**	4523.75	-	-	-	712654	712869	1172.96	**708404**	**708619**	1006.8	712654	712869	**631.25**
5,5,5	**92221**	**92364**	780.06	-	-	-	92535	92678	193.61	**92221**	**92364**	150.88	92535	92678	**83.14**
5,5,6	**5416**	**5451**	52.03	-	-	-	5512	5547	14.23	**5416**	**5451**	11.3	5512	5547	**6.11**

$\boldsymbol{z}_0$	{0}	{1}	{0}	{0,1}	{0,1}	{0,1}	{0}	{0}	{1}
$\boldsymbol{z}_1$	{0}	{1}	{0}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0}

This corresponds to the following state for $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$:

$\boldsymbol{x}_0$	{1}	{0,1,2}	{2}
$\boldsymbol{x}_1$	{1}	{0,1,2}	{0,1}

We can see that fewer values can be pruned by $\boldsymbol{z}_0 >_{lex} \boldsymbol{z}_1$ than $\boldsymbol{x}_0 <_{lex} \boldsymbol{x}_1$. Hence, `golferV1V3` has better number of fails and choice points than `golferV3'`.

In the social golfer problem, using multiple viewpoints and channeling constraints for symmetry breaking is clearly more beneficial than using models with only single viewpoint. When using a single viewpoint to tackle all the symmetries, we either choose a viewpoint that does not contain value symmetries (like the 0/1 viewpoint V_3) or express symmetry breaking constraints to break the value symmetries in the viewpoint (like model `golferV1`). However, the former makes propagation of the constraints less informative, as we have shown in the previous example. It also incurs an execution overhead because more variables have to be used. The latter has the main difficulty that symmetry breaking constraints for value symmetries are difficult to express. If we have to use a large number of constraints to break the value symmetries (as in `golferV1`), an overhead would be expected. By using multiple viewpoints and channeling constraints, we avoid both drawbacks. We do not have to express constraints in a viewpoint that allows less propagation. We also have better constraint expressiveness because value symmetries in one viewpoint become variable symmetries in another. Symmetry breaking constraints for variable symmetries can be expressed more succinctly. With better constraint expressiveness, it is more possible for extra constraint propagation and fewer total number of constraints, which incurs less overhead. Our experimental results demonstrate such benefits of our approach.

5 Conclusion

In this paper, we introduce a method to break symmetries in CSPs using multiple viewpoints of a problem and channeling constraints. We formalize the ideas of two types of symmetries in CSPs, namely variable and value symmetries. As we find that value symmetries are more difficult to express with symmetry breaking constraints than variable symmetries in general, we introduce a general principle in devising another viewpoint from a given one. By using a second viewpoint, we can transform value symmetries in the original viewpoint into variable symmetries in the other viewpoint so that symmetry breaking constraints can be expressed more easily. The two viewpoints are then connected using channeling constraints to obtain a channeled model with symmetry breaking constraints expressed in two viewpoints. We demonstrate our approach using the social golfer problem by building several models which use multiple viewpoints to break different types of symmetries of the problem. Since we can express symmetry breaking constraints more succinctly in another viewpoint than in the original one, this usually leads to better constraint propagation and fewer total number of constraints, which are the potential sources of speedup. We confirm with experimental results that models using multiple viewpoints for symmetry breaking show extra efficiency over those using single viewpoint only.

There can be several possible directions of future research from our current work. First, symmetry breaking constraints for different symmetries of a problem interact with each other. We have to ensure that different symmetry breaking constraints do not conflict. For example, Flener *et al.* [6] show that for a matrix model with row and column symmetries, each symmetry class must have an element where both the rows and columns of the matrix are lexicographically ordered. Therefore, the row ordering constraints and column ordering constraints do not conflict. When using multiple viewpoints to tackle different symmetries, since the symmetry breaking constraints are expressed in different viewpoints, it is more difficult to ensure that the constraints do not conflict. It is important to study when will (or will not) different symmetry breaking constraints for different symmetries interact adversely.

Second, our work currently shows benefits of using multiple viewpoints for symmetry breaking in the social golfer problem. It would be worthwhile to use the same approach to other problems that contain symmetries to seek for further evidence. Besides, we are conducting more theoretical work to confirm that there is a correspondence between value symmetries in one viewpoint and variable symmetries in another viewpoint. Theoretical results are important for the applicability of our approach.

Third, we define two types of symmetries in CSPs, namely variable and value symmetries. However, these two types of symmetries are not general enough to represent arbitrary symmetries in CSPs. For example, there are seven symmetries in the n-queens problem through rotations and reflections. If we use a variable for each row of the chessboard, we can only capture reflections along the horizontal and vertical axes as variable and value symmetries. The other symmetries of the problem (e.g., reflections along the diagonals and rotations) are neither variable nor value symmetries. It would be interesting to extend our multiple viewpoint and channeling constraints approach to handle arbitrary symmetries.

Fourth, our work currently focuses on adding symmetry breaking constraints to a model before search. An alternative approach to break symmetries is to dynamically add symmetry breaking constraints during search. It would be interesting to investigate how using multiple viewpoints and channeling constraints can be related to symmetry breaking during search. Research can be conducted to design a system for dynamic symmetry breaking making use of multiple viewpoints and/or channeling constraints.

References

1. R. Backofen and S. Will. Excluding symmetries in constraint-based search. *Constraints*, 7:333–349, 2002.

2. N. Barnier and P. Brisset. Solving the Kirkman's schoolgirl problem in a few seconds. In *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 477–491, 2002.
3. B. M. W. Cheng, K. M. F. Choi, J. H. M. Lee, and J. C. K. Wu. Increasing constraint propagation by redundant modeling: an experience report. *Constraints*, 4(2):167–192, 1999.
4. T. Fahle, S. Schamberger, and M. Sellmann. Symmetry breaking. In *Proceedings of the 7th International Conference on Principles and Practice of Constraint Programming*, pages 93–107, 2001.
5. P. Flener, A. M. Frisch, B. Hnich, C. Jefferson, Z. Kiziltan, I. Miguel, J. Pearson, and T. Walsh. Breaking symmetries in matrix models: A brief overview. In *the 10th Workshop on Automated Reasoning*, 2003.
6. 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 *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 462–476, 2002.
7. A. M. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, and T. Walsh. Global constraints for lexicographical orderings. In *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 93–108, 2002.
8. I. P. Gent, W. Harvey, and T. Kelsey. Groups and constraints: Symmetry breaking during search. In *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 415–430, 2002.
9. I. P. Gent and B. M. Smith. Symmetry breaking during search in constraint programming. In *Proceedings of the 14th European Conference on Artificial Intelligence*, pages 599–603, 2000.
10. I. P. Gent and T. Walsh. CSPLib: A benchmark library for constraints. In *Proceedings of the 5th International Conference on Principles and Practice of Constraint Programming*, pages 480–481, 1999. Available at `http://www.csplib.org/`.
11. C. Gervet. Conjunto: Constraint logic programming with finite set domains. In *Proceedings of the International Logic Programming Symposium*, pages 339–358, 1994.
12. C. Gervet. Interval propagation to reason about sets: Definition and implementation of a practical language. *Constraints*, 1(3):191–244, 1997.
13. ILOG. *ILOG Solver 4.4 Reference Manual*, 1999.
14. Y. C. Law and J. H. M. Lee. Algebraic properies of CSP model operators. In *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 750–754, 2002.
15. Y. C. Law and J. H. M. Lee. Model induction: a new source of CSP model redundancy. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pages 54–60, 2002.
16. A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.
17. I. McDonald and B. M. Smith. Partial symmetry breaking. In *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 431–445, 2002.
18. J. F. Puget. On the satisfiability of symmetrical constrained satisfaction problems. In *Proceedings of the 7th International Symposium on Methodologies for Intelligent Systems*, pages 350–361, 1993.
19. J. F. Puget. Symmetry breaking revisited. In *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 446–461, 1993.