# Symmetry

Pierre Flener

ASTRA Research Group on CP
Uppsala University
Sweden

ACP Summer School
Aussois (France), 5 and 6 May 2010

# Outline

# Outline

# Symmetry in Nature

Johannes Kepler, *On the Six-Cornered Snowflake*, 1611: six-fold rotational symmetry of snowflakes, role of symmetry in human perception and the arts, fundamental importance of symmetry in the laws of physics.

# Broken Symmetry in Nature

The Angora cat originated in the Turkish city of Ankara. It is admired for its long silky coat and quiet graceful charm. It is often bred to favour a pale milky colouring, as well as one blue and one amber eye. (*Turkish Daily News*, 14 Oct 2001)

# The Future?

## The Nobel Prize in Physics 2008

"for the discovery of the mechanism of spontaneous broken symmetry in subatomic physics"

"for the discovery of the origin of the broken symmetry which predicts the existence of at least three families of quarks in nature"

Photo: University of Chicago

**Yoichiro Nambu**

© The Nobel Foundation
Photo: U. Montan

**Makoto Kobayashi**

© The Nobel Foundation
Photo: U. Montan

**Toshihide Maskawa**

# Value Symmetry

## Example (Map colouring)

Use $n$ colours to paint the countries of a map such that no two neighbour countries have the same colour.
A model assigning colours (as values) to the countries (as decision variables) has $n!$ value symmetries because any permutation of the colours of a (non-)solution transforms that solution into another (non-)solution: the values (the colours) are not distinguished. ▸ Solution 1 ▸ Solution 2

## Example (Partitioned Map Colouring)

The available colours are partitioned into subsets, such that only colours of the same subset are not distinguished.
▸ Solution 1 ▸ Solution 2

# Variable Symmetry

## Example (Subsets)

Suppose an $n$-element subset of a given set has to be found, subject to some constraints.
A model encoding the subset as an array of $n$ distinct decision variables has $n!$ variable symmetries because the order of the elements does not matter in a set, but does matter in an array. ▸ Solution 1 ▸ Solution 2

## Careful: symmetries can be introduced!

Contrary to the first two examples, the symmetries identified in the third example are **not** symmetries of the problem itself, but symmetries of the model of the problem.

# Definitions

## Definition (Symmetry; also see Cohen *et al.*, *CP'05*)

A symmetry is a permutation of values or decision variables (or both) that preserves (non-)solutions.

Symmetries form a group:

- The inverse of a symmetry is a symmetry.
- The identity permutation is a symmetry.
- The composition of two symmetries is a symmetry.

(Computational) group theory is the way to study symmetry.

## Difficulty

A solver may waste a lot of effort exploring symmetric (partial) assignments, be they (partial) solutions or not.

# The Sport Scheduling Problem (SSP)

## Example (SSP: the problem)

Find schedule in $Periods \times Weeks \rightarrow Teams \times Teams$ for:

- $|Teams| = n$
- $|Weeks| = n - 1$
- $|Periods| = n/2$

subject to the following constraints:

1. Each team plays exactly once against each other team.
2. Each team plays exactly once per week.
3. Each team plays at most twice per period.

Intuitive idea for a matrix model and a solution for $n = 8$:

| | Wk 1 | Wk 2 | Wk 3 | Wk 4 | Wk 5 | Wk 6 | Wk 7 |
|---|---|---|---|---|---|---|---|
| P 1 | 1 vs. 2 | 1 vs. 3 | 2 vs. 6 | 3 vs. 5 | 4 vs. 7 | 4 vs. 8 | 5 vs. 8 |
| P 2 | 3 vs. 4 | 2 vs. 8 | 1 vs. 7 | 6 vs. 7 | 6 vs. 8 | 2 vs. 5 | 1 vs. 4 |
| P 3 | 5 vs. 6 | 4 vs. 6 | 3 vs. 8 | 1 vs. 8 | 1 vs. 5 | 3 vs. 7 | 2 vs. 7 |
| P 4 | 7 vs. 8 | 5 vs. 7 | 4 vs. 5 | 2 vs. 4 | 2 vs. 3 | 1 vs. 6 | 3 vs. 6 |

# The Sport Scheduling Problem (SSP)

## Example (SSP: the symmetries)

Observation: The periods, weeks, and teams of a sport schedule are not distinguished:

| | Wk 1 | Wk 2 | Wk 3 | Wk 4 | Wk 5 | Wk 6 | Wk 7 |
|---|---|---|---|---|---|---|---|
| P 1 | 1 vs. 2 | 1 vs. 3 | 2 vs. 6 | 3 vs. 5 | 4 vs. 7 | 4 vs. 8 | 5 vs. 8 |
| P 2 | 3 vs. 4 | 2 vs. 8 | 1 vs. 7 | 6 vs. 7 | 6 vs. 8 | 2 vs. 5 | 1 vs. 4 |
| P 3 | 5 vs. 6 | 4 vs. 6 | 3 vs. 8 | 1 vs. 8 | 1 vs. 5 | 3 vs. 7 | 2 vs. 7 |
| P 4 | 7 vs. 8 | 5 vs. 7 | 4 vs. 5 | 2 vs. 4 | 2 vs. 3 | 1 vs. 6 | 3 vs. 6 |

- The periods/rows can be permuted (4! variable syms).
- The weeks/columns can be permuted (7! var syms).
- The teams of a game can be permuted ($2!^{28}$ var syms).
- The team names can be permuted (8! value syms).

All these permutations do not affect whether any given (partial) assignment is a (partial) solution or not.

# The Social Golfer Problem (SGP)

## Example (SGP: the problem)

Find schedule in $\mathit{Weeks} \times \mathit{Groups} \times \mathit{Slots} \rightarrow \mathit{Golfers}$ for:

- $|\mathit{Weeks}| = w$
- $|\mathit{Groups}| = g$
- $|\mathit{Slots}| = s$
- $|\mathit{Golfers}| = g \cdot s$

subject to the following constraint:

1. Any two golfers play at most once in the same group.

Idea for matrix model and a solution for $\langle w, g, s \rangle = \langle 4, 4, 3 \rangle$:

| | Group 1 | Group 2 | Group 3 | Group 4 |
|---|---|---|---|---|
| Week 1 | [1, 2, 3] | [4, 5, 6] | [7, 8, 9] | [10, 11, 12] |
| Week 2 | [1, 4, 7] | [2, 5, 10] | [3, 8, 11] | [6, 9, 12] |
| Week 3 | [1, 8, 10] | [2, 4, 12] | [3, 5, 9] | [6, 7, 11] |
| Week 4 | [1, 9, 11] | [2, 6, 8] | [3, 4, 10] | [5, 7, 12] |

# The Social Golfer Problem (SGP)

## Example (SGP: the symmetries)

Observation: The weeks, groups, slots, and golfers of a social golfer schedule are not distinguished:

| | Group 1 | Group 2 | Group 3 | Group 4 |
|---|---|---|---|---|
| Week 1 | [1, 2, 3] | [4, 5, 6] | [7, 8, 9] | [10, 11, 12] |
| Week 2 | [1, 4, 7] | [2, 5, 10] | [3, 8, 11] | [6, 9, 12] |
| Week 3 | [1, 8, 10] | [2, 4, 12] | [3, 5, 9] | [6, 7, 11] |
| Week 4 | [1, 9, 11] | [2, 6, 8] | [3, 4, 10] | [5, 7, 12] |

- The weeks/rows can be permuted (4! variable symmetries).
- The groups/col.s can be permuted within a week ($4!^4$ var syms).
- The slots of a group can be permuted ($3!^{16}$ variable symmetries).
- The golfer names can be permuted (12! value symmetries).

All these permutations do not affect whether any given (partial) assignment is a (partial) solution or not.

# Terminology, for Variables and Values

**Definition (Special cases of symmetry)**

- Full symmetry: any permutation (i.e., bijection) preserves solutions. The full symmetry group $S_n$ has $n!$ symmetries over a sequence of $n$ elements.
- Partial symmetry: any piecewise permutation preserves solutions.
  Examples: weekdays vs weekend; same-size boats.
- Wreath symmetry: any wreath permutation preserves solutions. Example: the composition of the first two variable symmetries of the social golfer problem.
- Rotation symmetry: any rotation preserves solutions. The cyclic symmetry group $C_n$ has $n$ symmetries over a sequence of $n$ elements.

# Terminology (continued)

## Definition (Special cases of symmetry, continued)

- Index symmetry: any permutation of slices of a matrix of decision variables preserves solutions: full / partial row symmetry, column symmetry, ...
- Conditional / dynamic symmetry: a symmetry that appears while solving a problem. Example: after a few decisions, warehouses of initially different capacities may have the same remaining capacity. (Conditional symmetries are beyond the scope of this lecture.)

## Careful: symmetries multiply up!

If there is row and column symmetry in an $m \times n$ matrix (i.e., if there are $m!$ row syms and $n!$ column syms), then there are ~~$m! + n!$~~ $m! \cdot n!$ compositions of symmetries.

# Challenges Raised by Symmetries

## Definition (Symmetry handling)

- Detecting ~~the~~ symmetries of the problem (in a model) as well as ~~the~~ symmetries introduced when modelling.
- Breaking (better: exploiting) ~~the~~ detected symmetries, so that less search effort is spent.

## Scope of lecture

- This lecture is about symmetry breaking when solving combinatorial problems by systematic search that is interleaved with inference (here: propagation).
- When solving combinatorial problems by local search, the idea is often rather to exploit the presence of any symmetries by doing nothing, rather than by making the search space smaller.

# Outline

# Classification of Symmetry Breaking

## Definition (Symmetry equivalence class)

A symmetry class is an equivalence class of assignments under all the symmetries (including their compositions).

In each symmetry class, visit (at least) one member during search, as this may make a problem “more tractable”:

- Symmetry breaking by reformulation: elimination of ~~the~~ symmetries detected in a model.
- Static symmetry breaking: already when posting the problem constraints.
- Dynamic symmetry breaking: during search only.

## Careful: risky combination of SB methods!

Symmetry-breaking methods rarely combine without losing symmetry classes (and hence solutions).

# Classification of Symmetry Breaking

## Definition (Structural symmetry breaking)

Structural symmetry breaking is about exploiting the combinatorial structure of a problem (as well as the key strengths of CP, namely global constraints if not search procedures) toward eliminating, ideally in polynomial time and space, all symmetric sub-trees at every node explored (even if there are exponentially many symmetries).

## Careful: size does not matter!

A number of symmetries is no indicator of the difficulty of breaking them! For example, consider variable symmetry:

- The full group $S_n$ has $n!$ easily broken syms. ▸ Solution 2
- The cyclic group $C_n$ has only $n$ symmetries, which are much more difficult to break. ▸ Solution 1

# Symmetry Breaking by Reformulation

## Example (The sport scheduling problem)

Let the domain of the decision variables of an $\frac{n}{2} \times n$ matrix be $\{(f-1) \cdot n + s \mid 1 \leq f < s \leq n\}$: the game between teams $f$ and $s$ is uniquely identified by $(f-1) \cdot n + s$.

A round-robin schedule breaks many of the other syms:

- Fix the games of the first week to the set $\{(1,2)\} \cup \{(t+1, n+2-t) \mid 1 < t \leq n/2\}$
- For the other weeks, transform each $(f, s)$ into $(f', s')$:

$$f' = \begin{cases} 1 & \text{if } f = 1 \\ 2 & \text{if } f = n \\ f+1 & \text{otherwise} \end{cases}, \text{ and } s' = \begin{cases} 2 & \text{if } s = n \\ s+1 & \text{otherwise} \end{cases}$$

Determine the period of each game, but not its week!

# Symmetry Breaking by Reformulation

## Example (The social golfer problem)

Break the slot symmetries within each group by switching from the 3D $w \times g \times s$ matrix of scalar decision variables:

| | Group 1 | Group 2 | Group 3 | Group 4 |
|---|---|---|---|---|
| Week 1 | [1, 2, 3] | [4, 5, 6] | [7, 8, 9] | [10, 11, 12] |
| Week 2 | [1, 4, 7] | [2, 5, 10] | [3, 8, 11] | [6, 9, 12] |
| Week 3 | [1, 8, 10] | [2, 4, 12] | [3, 5, 9] | [6, 7, 11] |
| Week 4 | [1, 9, 11] | [2, 6, 8] | [3, 4, 10] | [5, 7, 12] |

to a 2D $w \times g$ matrix of set decision variables:

| | Group 1 | Group 2 | Group 3 | Group 4 |
|---|---|---|---|---|
| Week 1 | {1, 2, 3} | {4, 5, 6} | {7, 8, 9} | {10, 11, 12} |
| Week 2 | {1, 4, 7} | {2, 5, 10} | {3, 8, 11} | { 6, 9, 12} |
| Week 3 | {1, 8, 10} | {2, 4, 12} | {3, 5, 9} | { 6, 7, 11} |
| Week 4 | {1, 9, 11} | {2, 6, 8} | {3, 4, 10} | { 5, 7, 12} |

and adding the constraints that all sets must be of size $g$.

UPPSALA UNIVERSITET



# Outline

# Dynamic Symmetry Breaking (DSB)

## Definition (Dynamic symmetry breaking)

DSB = no addition of constraints to the problem model

## Classification

- Via the addition of constraints by the search procedure.
- Via a problem-specific search procedure.

## Benefit

No interference with dynamic variable / value orderings, especially problem-specific ones!

UPPSALA UNIVERSITET



# State of the Art

Two dual approaches, with large bodies of research:

- Symmetry breaking during search (SBDS, ...): after finding a no-good node in the search tree, add constraints preventing its symmetric nodes from being visited in the future.
- Symmetry breaking by dominance detection (SBDD, GCF, ...): before expanding a node, check whether a symmetric node thereof has been visited in the past.

The SBD$\star$ schemes are general and may take exponential time or space if there are exponentially many symmetries (and are beyond the scope of this lecture). Hence:

- Dynamic structural symmetry breaking (DSSB): exploit the combinatorial structure of the problem for designing a symmetry-free search procedure (in SBDD style).

# Dynamic Structural Symmetry Breaking

## Example (Map colouring: full value symmetry)

Consider two symmetric partial assignments:

$$\{Portugal \mapsto green, Spain \mapsto blue, France \mapsto green\}$$

$$\{Portugal \mapsto blue, Spain \mapsto red, France \mapsto blue\}$$

Not the values, but the **clustering** of the variables matters!
Compact representation, using (new) global constraints:

$$allEqual(Portugal, France) \quad \& \quad allEqual(Spain) \quad \& \quad allDifferent(Portugal, Spain)$$

This is an abstract no-good, based on **one** representative var in an *allDifferent* constraint for each symmetry class.

# Dynamic Structural Symmetry Breaking

## Example (Map colouring: symmetry-free search)

Given a partial colouring with $u$ colours, only $u + 1$ colours need to be considered for the next country $c$:

- Colour $c$ with one of the $u$ already used colours.
- Colour $c$ with an arbitrary unused colour (if any left).

In practice: The already used colours are the first $u$ colours, say $1..u$, so that the new colour to be considered is $u + 1$. This breaks all the $n!$ value symmetries in constant time and constant space overhead at every node explored! We say that it takes constant amortised time & space.

## Applications (Van Hentenryck [& Michel])

- Scene allocation (*INFORMS J. of Computing*, 2002)
- Steel mill slab design (*CPAIOR'08*)

# Partial Value Symmetry (*IJCAI'03*)

## Example (Partial value symmetry)

Weekdays vs the weekend days; same-capacity boats.

## Abstract no-goods

Let $D = D_1 \cup D_2 \cup \cdots \cup D_m$ be the domain of the variables, where the values in each set $D_i$ are fully interchangeable (full value sym for $m = 1$): variable clustering for each $D_i$.

## Search procedure: constant amortised time & space

In each set $D_i$, only the values already used and one so far unused value need to be tried.

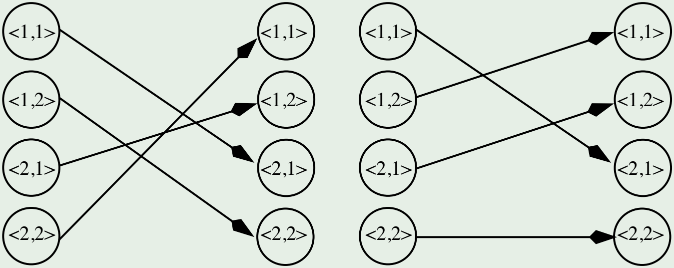
## Application (Michel, . . . , Van Hentenryck, *CPAIOR'08*)

- Eventually-serialisable data service deployment

# Wreath Value Symmetry (*IJCAI'03*)

## Example (Wreath value symmetry)

Schedule meetings in (day, room) pairs,
where the days are interchangeable,
and the rooms are interchangeable within each day:




Wreath permutation Not a wreath permutation!

# Wreath Value Symmetry (*IJCAI'03*)

## Abstract no-goods

Let $D = D_1 \times D_2$ be the domain of the decision variables, where the values in each set $D_i$ are fully interchangeable (full value sym for $|D_2| = 1$): one abstract no-good on $D_1$, and $m$ abstract no-goods on $D_2$ when $m$ values of $D_1$ are used, with variable clustering as for full value symmetry.

## Search procedure: constant amortised time & space

1. For the first value component, in set $D_1$, only the values already used and one so far unused value need to be tried. Let $d_1 \in D_1$ be the chosen value.
2. For the second value component, in set $D_2$, only the values already used with $d_1$ and one so far unused value need to be tried.

# Selected Other DSSB Results

Consider a combinatorial problem with *n* decision variables over a domain of *k* values:

- Generalisation to any value symmetry: group equivalence (GE) trees (Roney-Dougal *et al.*, *ECAI'04*)
  ☞ $O(n^4)$ time overhead at every node explored.
- Partial variable symmetry + partial value symmetry (Sellmann & Van Hentenryck, *IJCAI'05*)
  ☞ $O(k^{2.5} + n \cdot k)$ time at every node explored.
  ☞ Coinage of the term structural symmetry breaking.
  ☞ Can be specialised for full variable symmetry only.

UPPSALA UNIVERSITET



# Tractability of DSSB: State of the Art

| value symmetry \ variable symmetry | none | full | partial | wreath | |
|---|---|---|---|---|---|
| none | | P<br>**P** | P<br>**P** | P<br>**P** | scalar problem<br>**set problem** |
| full | P<br>**P** | P<br>**NP** | P<br>**NP** | NP<br>**NP** | scalar problem<br>**set problem** |
| partial | P<br>**P** | P<br>**NP** | P<br>**NP** | NP<br>**NP** | scalar problem<br>**set problem** |
| wreath | P<br>**P** | P<br>**NP** | P<br>**NP** | NP<br>**NP** | scalar problem<br>**set problem** |
| any | P | | | | scalar problem<br>**set problem** |

P: All symmetric sub-trees can be eliminated, say by DSSB, with a polynomial time & space overhead at every node explored.
NP: Dominance-detection schemes (in SBDD style) are NP-hard.

# In Mathematics: Combinatorial Generation

The listing, ranking, unranking, and random selection of objects of some combinatorial structure (combination, partition, permutation, subset, tree, etc) w.r.t. some order:

- Constant amortised time (CAT): in time proportional to the number of objects listed (after some initialisations).
- Backtracking ensuring success at terminals (BEST): every leaf of the backtracking tree is a desired object.
- Loopless: the next object is constructed without executing any loop.
- Memoryless: the next object is constructed without using any global variables (can start from any object).

# Combinatorial Objects

Consider sequences of $n = 3$ variables over $k = 2$ values:

| tuples | unlabelled tuples | necklaces | unlabelled necklaces |
|---|---|---|---|
| 000 | 000 | 000 | 000 |
| 001 | 001 | 001 | 001 |
| 010 | 010 | | |
| 011 | 011 | 011 | |
| 100 | | | |
| 101 | | | |
| 110 | | | |
| 111 | | 111 | |
| no symmetry | full value symmetry $S_k$ on values | rotation variable symmetry $C_n$ on variables | rot var + full val symmetry $C_n \times S_k$ |

UPPSALA UNIVERSITET



# Combinatorial Objects

Consider sequences of $n = 3$ variables over $k = 2$ values:

| tuples | unlabelled tuples | necklaces | unlabelled necklaces |
|---|---|---|---|
| 000 | 000 | 000 | 000 |
| 001 | 001 | 001 | 001 |
| 010 | 010 | | |
| 011 | 011 | 011 | ~~011~~ |
| 100 | | | |
| 101 | | | |
| 110 | | | |
| 111 | | 111 | |
| no symmetry | full value symmetry $S_k$ on values | rotation variable symmetry $C_n$ on variables | rot var + full val symmetry $C_n \times S_k$ |

# Combinatorial Generation

**Algorithm: full value symmetry (Er, *Computer J.* 1988)**

**procedure** $list(j, u : \textbf{integer})$ {$u =$ the largest used value}
**var** $i$ : **integer**
**if** $j > n$ **then**
  **return true** {all sym broken at all nodes!}
**else**
  **try all** $i = 0$ **to** $\min(u + 1, k - 1)$ **do**
    $X[j] \leftarrow i$;
    $list(j + 1, \max(i, u))$

Initial call: $list(1, -1)$

Complexity: Constant amortised time & space:

$$\#objects = \#unlabelled\ tuples$$

Property: Lexicographic enumeration (by var & val orders)

# Combinatorial Generation

**Algo: rot var sym (Ruskey & Sawada, *COCOON'00*)**

**procedure** $list(j, p : \textbf{integer})$ {$p = $ #positions to replicate}
**var** $i$ : **integer**
**if** $j > n$ **then**
  **return** $n \bmod p = 0$ {not all sym broken at all nodes!}
**else**
  **try all** $i = X[j-p]$ **to** $k-1$ **do**
    $X[j] \leftarrow i$;
    $list(j+1, \textbf{if } i = X[j-p] \textbf{ then } p \textbf{ else } j)$

Initial call: $X[0] \leftarrow 0$; $list(1,1)$, where $X[0]$ is a dummy var
Complexity: Constant amortised time & space:

$$\#objects \leq \#necklaces \cdot (k/(k-1))^2$$

Property: Lexicographic enumeration (by var & val orders)

# Outline

# Useful Constraints: Lexicographic Ordering

## Example

- $[C, o, n, s, t, r, a, i, n, t] \leq_{lex} [C, o, n, s, t, r, u, c, t, s]$ because letter '*a*' precedes '*u*' in the Latin alphabet.
- $[1, 2, 34, 5, 678] \leq_{lex} [1, 2, 36, 45, 78]$ because $34 \leq 36$ among the natural numbers.

## Definition (Lexicographic order)

A sequence $X = [x_1, \ldots, x_n]$ is lexicographically at most a sequence $Y = [y_1, \ldots, y_n]$ of the same type $T$ and the same size $n$, which is denoted by $X \leq_{lex} Y$, if and only if:

- either $n = 0$
- or $x_1 <_T y_1$
- or $x_1 = y_1$ **and** $[x_2, \ldots, x_n] \leq_{lex} [y_2, \ldots, y_n]$

# Static Symmetry Breaking (SSB)

## Definition (Static symmetry breaking)

SSB = addition of ordering constraints to the problem model

## Classification

- Lex-leader scheme (Crawford *et al.*, *KR'96*): post a $\leq_{lex}$ ordering constraint for each symmetry.

The lex-leader scheme is general and may take exponential space if there are exponentially many symmetries. Hence:

- Static structural symmetry breaking (SSSB): exploit the combinatorial structure of the problem for posting fewer (not necessarily $\leq_{lex}$) symmetry-breaking constraints.

## Careful

Potential interference with dynamic var / val orderings!

# Static Symmetry Breaking

Lexicographic ordering along one dimension of a matrix breaks all the index symmetries of that dimension.

## Example (The sport scheduling problem)

Breaking all the variable symmetries of the $\frac{n}{2} \times n$ matrix:

- Each row is lexicographically at most the next, if any.
- Each col. is lexicographically at most the next, if any.
- The first team of each game has a smaller number than the second team of the game (this constraint can also be enforced by a suitable definition of the domain of the decision variables).

This breaks all the variable symmetries in this case, because the matrix values are all different.

# Static Symmetry Breaking

When lexicographically ordering a matrix along every dimension with index symmetry:

- No symmetry class is of size 0.
- However, in general, not all sym classes are of size 1, except if all the matrix values are different, etc.

## Counterexample

Symmetric matrices with lex ordered rows and columns:

| | |
|---|---|
| 0 | 1 |
| 0 | 1 |
| 1 | 0 |

→

Swap the columns

Swap row 1 and 3

←

| | |
|---|---|
| 0 | 1 |
| 1 | 0 |
| 1 | 0 |

# The Lex-Leader Scheme

## Construction of lex-leader constraints

For any group $G$ of variable symmetries on decision variables $\{x_1, \ldots, x_n\}$ of domain / type $T$:

1. Choose a variable ordering, say $\langle x_1, \ldots, x_n \rangle$.
2. Choose a total value ordering on $T$, say $\leq_T$.
3. Choose a lexicographic order induced by $\leq_T$, say $\leq_{lex}$.
4. For every symmetry $\sigma \in G$, add the constraint

   $$[x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}]$$

   to the problem model.
5. Simplify the resulting constraints, locally and globally.

This yields exactly one solution per symmetry class.

UPPSALA UNIVERSITET



# The Lex-Leader Scheme

## Example (Row & column symmetry on a 2 $\times$ 3 matrix)

The group of row & column symmetries of $\begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \end{pmatrix}$ can be generated by 3 permutations (of the first two columns, of the last two columns, and of the two rows):

$$(1,2)(4,5) \qquad (2,3)(5,6) \qquad (1,4)(2,5)(3,6)$$

This group contains the following 12 permutations:

| | | |
|---|---|---|
| () | (1,2)(4,5) | (2,3)(5,6) |
| (1,4)(2,5)(3,6) | (1,6,2,4,3,5) | (1,5,3,4,2,6) |
| (1,4)(2,6)(3,5) | (1,5)(2,4)(3,6) | (1,6)(2,5)(3,4) |
| (1,3)(4,6) | (1,2,3)(4,5,6) | (1,3,2)(4,6,5) |

Symmetry $\{x_1 \mapsto x_2,\ x_2 \mapsto x_3,\ x_3 \mapsto x_1,\ x_4 \mapsto x_5,\ x_5 \mapsto x_4\}$ is here denoted by the the cycle notation $(1,2,3)(4,5)$.

# The Lex-Leader Scheme

## Example (2 × 3 matrix, continued)

Constraints for var ordering $\langle x_1, x_2, x_3, x_4, x_5, x_6\rangle$:

$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_2,x_1,x_3,x_5,x_4,x_6]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_1,x_3,x_2,x_4,x_6,x_5]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_4,x_5,x_6,x_1,x_2,x_3]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_6,x_4,x_5,x_3,x_1,x_2]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_5,x_6,x_4,x_2,x_3,x_1]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_4,x_6,x_5,x_1,x_3,x_2]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_5,x_4,x_6,x_2,x_1,x_3]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_6,x_5,x_4,x_3,x_2,x_1]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_3,x_2,x_1,x_6,x_5,x_4]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_2,x_3,x_1,x_5,x_6,x_4]$$
$$[x_1,x_2,x_3,x_4,x_5,x_6] \leq_{lex} [x_3,x_1,x_2,x_6,x_4,x_5]$$

# The Lex-Leader Scheme

## Example (2 × 3 matrix, continued)

Simplified constraints for var ordering $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle$:

$$[x_1, x_4] \leq_{lex} [x_2, x_5]$$
$$[x_2, x_5] \leq_{lex} [x_3, x_6]$$
$$[x_1, x_2, x_3] \leq_{lex} [x_4, x_5, x_6]$$
$$[x_1, x_2, x_3] \leq_{lex} [x_6, x_4, x_5]$$
$$[x_1, x_2, x_3, x_4] \leq_{lex} [x_5, x_6, x_4, x_2]$$
$$[x_1, x_2, x_3] \leq_{lex} [x_4, x_6, x_5]$$
$$[x_1, x_2, x_3] \leq_{lex} [x_5, x_4, x_6]$$
$$[x_1, x_2, x_3] \leq_{lex} [x_6, x_5, x_4]$$

# The Lex-Leader Scheme

## Example (Full variable symmetry)

For the $n!$ symmetries of the full symmetry group $S_n$, the $n!$ $n$-ary $\leq_{lex}$ constraints (over lists of length $n$) simplify into $n - 1$ binary $\leq$ constraints (over scalars):

$$x_1 \leq x_2 \leq \cdots \leq x_{n-1} \leq x_n$$

## In practice

Breaking all the symmetries will reduce the search effort, but at the expense of increased propagation effort:

- Break only some symmetries, but which ones?
- Double-lex (*lex*$^2$) often works well: pick the symmetries that swap adjacent rows or columns of a 2D matrix with full row & column symmetry. ▸ Example

UPPSALA UNIVERSITET



# The Lex-Leader Scheme

## Adaptation for value symmetries (Walsh, *CP'06*)

For any group $G$ of value symmetries on decision variables $\{x_1, \ldots, x_n\}$ of domain / type $T$:

1. Choose a variable ordering, say $\langle x_1, \ldots, x_n \rangle$.
2. Choose a total value ordering on $T$, say $\leq_T$.
3. Choose a lexicographic order induced by $\leq_T$, say $\leq_{lex}$.
4. For every symmetry $\sigma \in G$, add the constraint

$$[x_1, \ldots, x_n] \leq_{lex} [\sigma(x_1), \ldots, \sigma(x_n)]$$

to the problem model.

This yields exactly one solution per symmetry class.

Uppsala Universitet



# Static Structural Symmetry Breaking

## Example (Full / partial value sym; Law & Lee, *CP'04*)

Consider decision variables $X$ in domain $D = 0, \ldots, k-1$.

Full value symmetry-breaking constraints:
The first occurrences of the domain values are ordered:

$$\mathit{firstPos}(0, X) < \mathit{firstPos}(1, X) < \cdots < \mathit{firstPos}(k-1, X)$$

Global constraint for the conjunction of these constraints:

$$\mathit{intValuePrecedeChain}(X, D)$$

Partial value symmetry over domain $D = D_1 \cup D_2 \cup \cdots \cup D_m$:

$$\bigwedge_{i=1}^{m} \mathit{intValuePrecedeChain}(X, D_i)$$

# Static Structural Symmetry Breaking

## Example (Partial variable sym + full value sym; *CP'06*)

- Make study groups for two sets of five indistinguishable students each. There are six indistinguishable tables.
- The decision variables $\{f_1, \ldots, f_5\} \cup \{m_6, \ldots, m_{10}\}$ correspond to the students and are to be assigned table values from the ordered domain $\{t_1, \ldots, t_6\}$.
- Constraints breaking the variable symmetries:

$$f_1 \leq f_2 \leq f_3 \leq f_4 \leq f_5 \quad \& \quad m_6 \leq m_7 \leq m_8 \leq m_9 \leq m_{10}$$

- Constraints computing the signatures (counter pairs):

$$\begin{array}{l} \mathit{cardinality}([f_1, \ldots, f_5],\ [t_1, \ldots, t_6],\ [c_1^f, \ldots, c_6^f]) \quad \& \\ \mathit{cardinality}([m_6, \ldots, m_{10}],\ [t_1, \ldots, t_6],\ [c_1^m, \ldots, c_6^m]) \end{array}$$

- Constraints breaking the value symmetries:

$$[c_1^f, c_1^m] \geq_{lex} \cdots \geq_{lex} [c_6^f, c_6^m]$$

UPPSALA UNIVERSITET



# Static Structural Symmetry Breaking

## Example (Partial variable sym + full value sym; *CP'06*)

Consider the satisfying assignment

$$\{f_1 \mapsto t_1, f_2 \mapsto t_1, f_3 \mapsto t_2, f_4 \mapsto t_3, f_5 \mapsto t_4, m_6 \mapsto t_1, m_7 \mapsto t_2, m_8 \mapsto t_2, m_9 \mapsto t_3, m_{10} \mapsto t_5\}.$$

Indeed, the variable-symmetry constraints are satisfied:

$$f_1 \leq f_2 \leq f_3 \leq f_4 \leq f_5 \quad \& \quad m_6 \leq m_7 \leq m_8 \leq m_9 \leq m_{10}$$

and the value-symmetry constraints are satisfied:

$$[2, 1] \geq_{lex} [1, 2] \geq_{lex} [1, 1] \geq_{lex} [1, 0] \geq_{lex} [0, 1] \geq_{lex} [0, 0]$$

Note that a pointwise ordering would not have sufficed.

# Static Structural Symmetry Breaking

## Example (Partial variable sym + full value sym; *CP'06*)

If student $m_{10}$ moves from table $t_5$ to table $t_6$, producing a symmetrically equivalent assignment because the tables are fully interchangeable:

$$\{f_1 \mapsto t_1, f_2 \mapsto t_1, f_3 \mapsto t_2, f_4 \mapsto t_3, f_5 \mapsto t_4, m_6 \mapsto t_1, m_7 \mapsto t_2, m_8 \mapsto t_2, m_9 \mapsto t_3, m_{10} \mapsto t_6\}$$

then the value-symmetry constraints are violated:

$$[2,1] \geq_{lex} [1,2] \geq_{lex} [1,1] \geq_{lex} [1,0] \geq_{lex} [0,0] \not\geq_{lex} [0,1]$$

# Static Structural Symmetry Breaking

## Example (Partial variable sym + full value sym; *CP'06*)

If students $m_9$ and $m_{10}$ swap their assigned tables, producing a symmetrically equivalent assignment because both students are male:

$$\{f_1 \mapsto t_1, f_2 \mapsto t_1, f_3 \mapsto t_2, f_4 \mapsto t_3, f_5 \mapsto t_4, m_6 \mapsto t_1, m_7 \mapsto t_2, m_8 \mapsto t_2, m_9 \mapsto t_5, m_{10} \mapsto t_3\}$$

then the signatures do not change and hence the value-symmetry constraints remain satisfied, but the variable-symmetry constraints are violated, because

$$m_6 \leq m_7 \leq m_8 \leq m_9 \not\leq m_{10}$$

# Selected Other SSSB Results (*CP'06*)

Consider a combinatorial problem with $n$ decision variables over a domain of $k = \ell \cdot m$ values:

- Partial variable symmetry + partial value symmetry:
  $O(n + k)$ constraints break $O(n! \cdot k!)$ symmetries
- Generalisation:
  Partial variable symmetry + wreath value symmetry:
  $O(n + k)$ constraints break $O(n! \cdot (m!)^{\ell} \cdot \ell!)$ symmetries

UPPSALA UNIVERSITET



# Outline

# Classification of Symmetry Detection

## Careful

General symmetry detection schemes are graph-isomorphism complete (and are beyond the scope of this lecture).

## Definition (Structural symmetry detection)

Structural symmetry detection is about exploiting the combinatorial structure of a problem toward deriving, ideally in polynomial time and space, ~~the~~ symmetries of the model (even if there are exponentially many derived symmetries):

- Static structural sym detection: when posting.
- Dynamic structural sym detection: when searching.

# Outline

# Static Structural Symmetry Detection

## Bottom-up derivation (*SARA'05*)

Key insight: Once the symmetries of (global) constraints and functions are identified (manually), the symmetries of a model with these constraints and functions can be derived compositionally, automatically, and efficiently:

- Symmetry identification
- Symmetry composition

A subset of our results turned out to be in (Roy & Pachet, *ECAI'98 Workshop on Non-Binary Constraints*).

# Static Structural Symmetry Detection



## Symmetry identification

Consider a problem with variables $X$ over domain $D$:

- Constraint $\mathit{allDifferent}(x_1, \ldots, x_n)$ has full value sym.
- Function $\mathit{nbDistinct}(x_1, \ldots, x_n)$ has full value symmetry.
- Constraint $\mathit{atMost}(m, d, [x_1, \ldots, x_n])$ has partial value symmetry over the partition $\{d\} \cup (D \setminus \{d\})$ of $D$ (at most $m$ occurrences of $d$ among the variables $x_i$).
- Constraint $x_1 < x_2$ has partial variable symmetry over the partition $\{x_1\} \cup \{x_2\} \cup (X \setminus \{x_1, x_2\})$ of $X$.

Similarly for row and column symmetries.

☞ Extend the Global Constraint Catalogue accordingly!

UPPSALA UNIVERSITET



# Static Structural Symmetry Detection

## Symmetry composition

Consider a problem with variables $X$ over $D = \{a, \ldots, h\}$:

- If the constraints $c_1$ and $c_2$ have full value symmetry, then their conjunction $c_1 \ \& \ c_2$ has full value symmetry.
- Extension to functions. Example: The expression $3 \cdot \mathit{nbDistinct}(x_1, x_2, x_3) + 4 \cdot \mathit{nbDistinct}(x_4, x_5, x_6)$ has full value symmetry.
- Generalisation to partial symmetry (PS). Example: $\mathit{atMost}(i, a, X)$ has PS over $\{a\} \cup \{b, c, \ldots, h\}$, and $\mathit{atMost}(j, b, X)$ has PS over $\{b\} \cup \{a, c, \ldots, h\}$, so their conj. has PS over $\{a\} \cup \{b\} \cup \{c, \ldots, h\}$ if $i \neq j$, but PS over $\{a, b\} \cup \{c, \ldots, h\}$ if $i = j$:
  ☞ Need for aggregation into $\mathit{atMost}([i, j], [a, b], X)$.

☞ Each composition takes time polynomial in $|X| + |D|$.

# Static Structural Symmetry Detection

## Example (Detected symmetries)

- Scene allocation problem:
  full value symmetry (indistinguishable days)
- Progressive party problem:
  partial row symmetry (same-size guest crews),
  full column symmetry (interchangeable periods), and
  partial value symmetry (same-capacity host boats)

# Outline

# Dynamic Structural Symmetry Detection



## Example

Consider a combinatorial problem with decision variables $X = [x_1, x_2, \ldots, x_n]$ over domain $D = \{a, \ldots, h\}$:

- The constraint $atMost([3, 2], [a, b], X)$ has partial value symmetry over $\{a\} \cup \{b\} \cup \{c, \ldots, h\}$.
- The decision $x_1 = a$ has partial value symmetry over $\{a\} \cup \{b, c, \ldots, h\}$.
- Their conjunction thus has partial value symmetry over $\{a\} \cup \{b\} \cup \{c, \ldots, h\}\}$.
- Projection onto $X \setminus \{x_1\}$ of the original constraint gives $atMost([2, 2], [a, b], [x_2, \ldots, x_n])$, which has partial value symmetry over $\{a, b\} \cup \{c, \ldots, h\}$:
  a new symmetry was dynamically detected!

UPPSALA UNIVERSITET



# Évariste Galois (1811–1832)

Évariste Galois was one of the parents of group theory.
Insight: The structure of the symmetries of an equation determines whether it has solutions or not.

Marginal note in his last paper: "*Il y a quelque chose à compléter dans cette démonstration. Je n'ai pas le temps.*"
(There is something to complete in this demonstration. I do not have the time.)

# Outline

# In Practice

- Few symmetries in real-life problems?
- Keep in mind the objective:
  first solution, all solutions, or best solution?
  Symmetry breaking might not pay off when searching for the first solution.
- Problem constraints can sometimes be simplified in the presence of symmetry-breaking constraints.
  Example: $z = |x - y|$ can be simplified into $z = x - y$ if symmetry breaking requires $x \geq y$.

# Future Work: Other Orders

- Lexicographic order: $12 <_{lex} 13 <_{lex} 21$

  $$000, 001, 010, 011, 100, 101, 110, 111$$

- Co-lexicographic order: $21 <_{colex} 12 <_{colex} 13$

  ☞ Shorter, faster, more elegant and natural algorithms!
- Gray order (Gray, *US Patent* 2632058, 1953):

  $$\underline{0}00, 00\underline{1}, 0\underline{1}1, 01\underline{0}, \underline{1}10, 11\underline{1}, 1\underline{0}1, 10\underline{0}$$

  ☞ Only one value (underlined) changes each time!
- Boustrophedonic order (Flajolet *et al.*, *TCS*, 1994): turning like oxen in ploughing; the writing of alternate lines in opposite directions (Merriam-Webster)

Used for listing objects in combinatorial generation (DSSB), but can / should be turned into constraints (for SSSB)!

# Future Work



- Heuristics for incomplete symmetry breaking.
- Handling of conditional / dynamic symmetries.
- Push symmetry breaking into global constraints.
- Symmetry detection and breaking in CP systems.

# Acknowledgements

STINT The Swedish Foundation for International Cooperation in Research and Higher Education

- CORSA project
  - Uppsala University and Brown University (RI, USA)
  - Institutional grant IG2001-67 of STINT
  - From September 2001 to December 2007
  - Justin Pearson,
    Meinolf Sellmann,
    Pascal Van Hentenryck,
    Magnus Ågren
- SymCon workshop community, esp. Barbara Smith

# Main References by Lecturer


- P. Flener, J. Pearson, and M. Sellmann.
  Static and dynamic structural symmetry breaking.
  *Annals of Mathematics and Artificial Intelligence*, 201x.
  (Supersedes our CP'06 paper with P. Van Hentenryck.)
- P. Flener, J. Pearson, M. Sellmann, P. Van Hentenryck, M. Ågren.
  Dynamic SSB for constraint satisfaction problems.
  *Constraints*, 14(4), December 2009.
  (Supersedes our IJCAI'03 and IJCAI'05 papers.)
- P. Flener and J. Pearson.
  Solving necklace constraint problems.
  *Journal of Algorithms*, 64(2–3):61–73, April–July 2009.
  (Supersedes our ECAI'08 paper.)
- P. Van Hentenryck, P. Flener, J. Pearson, and M. Ågren.
  Compositional derivation of symmetries for constraint satisfaction.
  *Proc. of SARA'05*. LNAI 3607:234–247. Springer-Verlag, 2005.

UPPSALA UNIVERSITET

Appendix

Acknowledgements

References

# Main References by Others


- I.P. Gent, K.E. Petrie, and J.-F. Puget.
  Symmetry in constraint programming.
  Chapter 22 of *Handbook of Constraint Programming*, 2006.

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

- F. Ruskey.
  Combinatorial Generation.
  Unpublished book draft, 2003.