Topic 5: Symmetry
(Version of 13th November 2020)

Pierre Flener

Optimisation Group
Department of Information Technology
Uppsala University
Sweden

Course 1DL441:
Combinatorial Optimisation and Constraint Programming,
whose part 1 is Course 1DL451:
Modelling for Combinatorial Optimisation

Outline

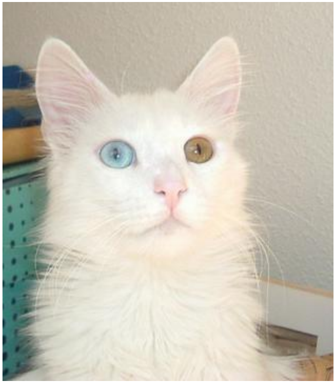
Outline

Symmetry in Nature

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

UPPSALA UNIVERSITET

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 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 k colours to paint the countries of a map such that no neighbour countries have the same colour.

The model where the countries (as decision variables) take colours (as values) has $k!$ value symmetries because any permutation of the colours transforms a (non-)solution into another (non-)solution: the values are not distinguished. (Continued on slide 36)

Example (Partitioned map colouring)

The colours of map colouring are partitioned into subsets, such that only the colours of the same subset are not distinguished. (Continued on slide 36)

Variable Symmetry

Example (*n*-Queens)

The model with a 2d array of decision variables in `0..1` has 4 reflection symmetries and 4 rotation symmetries, which are variable symmetries, as any reflection or rotation of an $n \times n$ board with n queens transforms that (non-) solution into another (non-)solution. (Continued on slide 35)

Example (Subset)

Find an n-element subset of a given set S, such that some constraints are satisfied.

The model encoding the subset as an array of n decision variables of domain S, constrained to take distinct values, has $n!$ variable symmetries as the order of the elements does not matter in a set, but does matter in an array. (Continued on slide 34)

Symmetries can be introduced!

- The symmetries in the (partitioned) map colouring and *n*-queens models are actually problem symmetries: they are detectable in *every* model.
- The symmetries in the subset model are *not* problem symmetries but model symmetries: they are *not* detectable in every model.
- There can also be instance symmetries, which are detectable in the instance data of a problem. Example: cargo boats with the same capacity.

Observation:

A solver may waste a lot of effort on gazillions of (partial) non-solutions that are symmetric to already visited ones, whereas a found solution can be transformed without search into a symmetric solution in polynomial time.

Definition (also see Cohen *et al.* @ *Constraints*, 2006)

A symmetry is a permutation of values or decision variables (or both) that preserves solutions: it transforms (partial) solutions into (partial) solutions, and it transforms (partial) non-solutions into (partial) 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.

Example (Agricultural experiment design, AED)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	1	1	1	0	0	0	0
corn	1	0	0	1	1	0	0
millet	1	0	0	0	0	1	1
oats	0	1	0	1	0	1	0
rye	0	1	0	0	1	0	1
spelt	0	0	1	1	0	0	1
wheat	0	0	1	0	1	1	0

Constraints to be satisfied:

1. Equal growth load: Every plot grows 3 grains.
2. Equal sample size: Every grain is grown in 3 plots.
3. Balance: Every grain pair is grown in 1 common plot.

Instance: 7 plots, 7 grains, 3 grains/plot, 3 plots/grain, balance 1.

General term: balanced incomplete block design (BIBD).

Example (AED and BIBD: the symmetries)

Observation: The grains and plots of an agricultural experiment design are not distinguished:

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	1	1	1	0	0	0	0
corn	1	0	0	1	1	0	0
millet	1	0	0	0	0	1	1
oats	0	1	0	1	0	1	0
rye	0	1	0	0	1	0	1
spelt	0	0	1	1	0	0	1
wheat	0	0	1	0	1	1	0

- The grains / rows can be permuted: 7! variable sym.s
- The plots / columns can be permuted: 7! var. sym.s

All these permutations preserve solutions.
(Continued on slide 28)

Example (The sport scheduling problem, SSP)

Find schedule in $\texttt{Periods} \times \texttt{Weeks} \rightarrow \texttt{Teams} \times \texttt{Teams}$ for

- $|\texttt{Teams}| = \texttt{n}$ and $\texttt{n}$ is even
- $|\texttt{Weeks}| = \texttt{n-1}$
- $|\texttt{Periods}| = \texttt{n}/2$ periods per week

subject to the following constraints:

1. Each possible game is played exactly once.
2. Each team plays exactly once per week.
3. Each team plays at most twice per period.

Idea for a model, and a solution for $\texttt{n=8}$ as an array `Game`:

	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

Example (SSP: the symmetries)

Observation: The periods, weeks, game slots, 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 sym.s
- The weeks / columns can be permuted: 7! var. sym.s
- The game slots can be permuted: $2!^{28}$ variable sym.s
- The team names can be permuted: 8! value sym.s

All these permutations preserve solutions.
(Continued on slides 23 and 29)

Example (The social golfer problem, SGP)

Find schedule $\texttt{Weeks} \times \texttt{Groups} \times \texttt{Slots} \to \texttt{Players}$ for

- $|\texttt{Weeks}| = \texttt{w}$
- $|\texttt{Groups}| = \texttt{g}$ groups per week
- $|\texttt{Slots}| = \texttt{s}$ players per group
- $|\texttt{Players}| = \texttt{g} \cdot \texttt{s}$

subject to the following constraint:

1. Any two players are at most once in the same group.

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

	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	$[1,9,11]$	$[2,6,8]$	$[3,4,10]$	$[5,7,12]$

By the way, there is no solution when adding a fifth week!

Example (SGP: the symmetries)

Observation: The weeks, groups, group slots, and players 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 can be permuted within a week: $4!^4$ var. symmetries
- The group slots can be permuted: $3!^{16}$ variable symmetries
- The player names can be permuted: 12! value symmetries

All these permutations preserve solutions.
(Continued on slide 24)

Terminology, for Variable and Value Sym.s

Definitions (Special cases of symmetry)

- Full symmetry: any permutation preserves solutions. The full symmetry group S_n has $n!$ symmetries over a sequence of n elements.
- Partial symmetry: any piecewise permutation preserves solutions. This often occurs in instances. **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 circular sequence of n elements.

Definitions (Special cases of symmetry, end)

- Index symmetry: any permutation of slices of an array of decision variables preserves solutions:
  full vs partial row symmetry, column symmetry, ...
- Conditional or dynamic symmetry: a symmetry that appears while solving a problem.
  Such symmetries are beyond the scope of this topic.

Careful: Index symmetries multiply up!

If there is full row and column symmetry in an $m \times n$ array (that is, if there are $m!$ row sym.s and $n!$ column sym.s), then there are ~~$m! + n!$~~ $m! \cdot n!$ compositions of symmetries, and at most $m! \cdot n! - 1$ symmetric solutions per solution.

For example, none of the $2^{1 \cdot 4} = 16$ Boolean 1×4 arrays can have $1! \cdot 4! - 1 = 23$ distinct symmetric arrays.

Challenges Raised by Symmetries

Definition

Symmetry handling has two aspects:

- 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 effort is spent on the solving: multiple symmetric representations of a solution are avoided.

Automated detection is beyond the scope of this topic.

UPPSALA UNIVERSITET

Classification of Symmetry Breaking

Definition

A symmetry class is an equivalence class of solutions under all the considered symmetries, including their compositions.

Aim: While solving, keep ideally one member per symmetry class, as this may make a problem "less intractable":

- Symmetry breaking by reformulation:
  the elimination of ~~the~~ symmetries detectable in model.
- Static symmetry breaking:
  the elimination of symmetric solutions by constraints.
- Dynamic symmetry breaking:
  the elimination of symmetric solutions by search.
  This is beyond the scope of this topic:
  ☞ see Topic 15: Search (in Part 2).

Definition

Structural symmetry breaking exploits the combinatorial structure of a problem by using the key strengths of constraint-based modelling — namely constraint predicates and search strategies — towards eliminating, ideally in low polynomial time and space, all symmetric solutions, 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 symmetries: see slide 34.
- The cyclic group C_n has only n symmetries, which are more difficult to break.

UPPSALA UNIVERSITET

Outline

Symmetry Breaking by Reformulation

Example (The sport scheduling problem over n teams)

Let the **domain** of the decision variables of an $\frac{n}{2} \times n$ array called `Game` be $\{f \cdot n + s \mid 1 \leq f < s \leq n\}$: the game between teams f and s is uniquely identified by $f \cdot n + s$.

A **round-robin schedule** breaks many of the other sym.s:

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

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

We must determine the period of each game, not its week!

Example (The social golfer problem, SGP)

Break the slot symmetries (slide 16) within each group by switching from a 3d $\mathtt{w} \times \mathtt{g} \times \mathtt{s}$ array of integer 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 $\mathtt{w} \times \mathtt{g}$ array of set variables
(see Topic 2: Basic Modelling):

	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 constraint that all sets must be of size $\mathtt{s}$.

UPPSALA UNIVERSITET

Outline

Useful Predicates: Lexicographic Ordering

Example

`lex_lesseq([1,2,34,5,678], [1,2,36,45,78])`, because `34 < 36`, even though `678` $\not\leq$ `78`: this $\leq_{\text{lex}}$ order is induced by $\leq$ and is not the point-wise ordering.

Definition

A `lex_lesseq(A,B)` constraint, where `A` and `B` are same-length 1d arrays of variables, say both with indices in `1..n`, holds iff `A` is lexicographically at most equal to `B`:

- either `n=0`,
- or `A[1]<B[1]`,
- or `A[1]=B[1]` & `lex_lesseq(A[2..n],B[2..n])`.

Variant predicates exist.

Symmetry Breaking by Constraints

Classification:

- Lex-leader scheme (Crawford *et al.*, *KR'96*) (slide 31): state one lexicographic constraint per var. symmetry. This is general, but takes exponential space if there are exponentially many symmetries, as is often the case.
- Structural symmetry breaking by constraints (slide 36): exploit the combinatorial structure of a problem for stating fewer symmetry-breaking constraints, and not necessarily lexicographic ones. This has already been worked out for some common combinations of variable symmetries, value symmetries, or both.

Careful: Symmetry-breaking constraints should harmonise with the choices of dummy values (see Topic 4: Modelling, and slide 36) and search-strategy annotations (see Topic 8: Inference & Search in CP & LCG).

Lexicographic ordering constraints along one dimension of an array break the index symmetry of that dimension.

Example (Balanced incomplete block design, BIBD)

The following constraints break all the row and column symmetries (see slide 12), but not all their compositions:

- Each row is `lex_greater` than the next, if any. Note that rows cannot be equal, as that would lead to violating the (so far unstated) incompleteness condition $2 \leq \texttt{blockSize} < |\texttt{Varieties}|$ on the parameters.
- Each column is `lex_greatereq` than the next, if any. Note that columns can be equal when $\texttt{balance} \geq 2$.

The use of `lex_greatereq` (as opposed to `lex_lesseq`) will be justified in Topic 8: Inference & Search in CP & LCG.

Lexicographic ordering constraints along one dimension of an array break the index symmetry of that dimension.

Example (The sport scheduling problem, SSP)

The following constraints simplify the row and column lexicographic constraints on the $\frac{n}{2} \times (n-1)$ array `Game`:

- each game in the first col. is less than the next, if any,
- each game in the first row is less than the next, if any,

as the values of array `Game` are necessarily all different.

With the following constraint on an $\frac{n}{2} \times n \times 2$ array `Team`:

- the first team of each game has a smaller number than the second team of the game (this constraint can also be enforced by the definition on slide 23 of the domain of the `Game[p,w]` decision variables),

and channelling, this breaks all the variable symmetries (of slide 14, including all their compositions) in this case, as the values of array `Game` are necessarily all different.

Lexicographic ordering constraints along every dimension with index symmetry of an array have two properties:

+ No symmetry class is lost.
– In general, not all symmetry classes are of size 1, except if the values of the array are all different, etc.

Counterexample

Assume full row symmetry and full column symmetry: the arrays $\begin{array}{|c|c|c|}\hline 0 & 0 & 1 \\ \hline 1 & 1 & 0 \\ \hline \end{array}$ and $\begin{array}{|c|c|c|}\hline 0 & 1 & 1 \\ \hline 1 & 0 & 0 \\ \hline \end{array}$ have lexicographically ordered rows and columns, but are symmetric: they can be transformed into each other by simultaneously swapping their two rows and swapping their first and last columns.

The lex-leader scheme (next slide) generates lexicographic ordering constraints that are not necessarily along the dimensions of an array of decision variables. It guarantees that all the compositions of all var. symmetries are broken.

The Lex-Leader Scheme

For any group G of variable symmetries on the indices of the decision variables `x1,...,xn` of domain D, which are not necessarily arranged into a 1d array:

1. Choose a variable ordering, say `[x1,...,xn]`.
2. Choose a total value ordering $\preceq$ on D.
3. Choose a lexicographic-ordering predicate induced by $\preceq$, say `lex_lesseq` (induced by $\leq$).
4. For every symmetry $\sigma \in G$, add the constraint

   `lex_lesseq([x1,...,xn],[x`σ`(1),...,x`σ`(n)])`

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

This yields exactly one solution per symmetry class.

Example (2 × 3 array with full row and column sym)

Consider the array

x1	x2	x3
x4	x5	x6

with full row and column symmetry:

$2! \cdot 3! - 1$ constraints for the variable ordering `x1,x2,x3,x4,x5,x6`:

```
lex_lesseq([x1,x2,x3,x4,x5,x6],[x2,x1,x3,x5,x4,x6])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x1,x3,x2,x4,x6,x5])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x4,x5,x6,x1,x2,x3])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x6,x4,x5,x3,x1,x2])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x5,x6,x4,x2,x3,x1])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x4,x6,x5,x1,x3,x2])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x5,x4,x6,x2,x1,x3])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x6,x5,x4,x3,x2,x1])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x3,x2,x1,x6,x5,x4])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x2,x3,x1,x5,x6,x4])
lex_lesseq([x1,x2,x3,x4,x5,x6],[x3,x1,x2,x6,x4,x5])
```

Example (2 × 3 array with full row and column sym)

Consider the array

x1	x2	x3
x4	x5	x6

with full row and column symmetry:

simplified constraints for the variable ordering `x1,x2,x3,x4,x5,x6`:

```
1  lex_lesseq([x1      ,x4      ],[x2      ,x5      ])
2  lex_lesseq([   x2      ,x5   ],[   x3      ,x6   ])
3  lex_lesseq([x1,x2,x3         ],[x4,x5,x6         ])
4  lex_lesseq([x1,x2,x3         ],[x6,x4,x5         ])
5  lex_lesseq([x1,x2,x3,x4      ],[x5,x6,x4,x2      ])
6  lex_lesseq([x1,x2,x3         ],[x4,x6,x5         ])
7  lex_lesseq([x1,x2,x3         ],[x5,x4,x6         ])
8  lex_lesseq([x1,x2,x3         ],[x6,x5,x4         ])
9  % constraints 1&2: lexicographic ordering on columns
10 % constraint  3:    lexicographic ordering on rows
11 % constraint  4 bans right array of counterex on s.30
```

Example (Full variable symmetry)

For the $n!$ symmetries of the full symmetry group S_n, the $n! - 1$ n-ary `lex_lesseq` constraints (over arrays of size n) simplify into $n - 1$ binary $\leq$ constraints (over integers):

```
x1 ≤ x2 /\ x2 ≤ x3 /\ ... ≤ xn
```

Let the chosen variable ordering form a 1d array `A`, such as in the model for the subset problem of slide 8: rather use `increasing(A)`.

In practice:

Breaking all the symmetries may increase the solving time:

- Break only some symmetries, but which ones?
- Double-lex often works well on a 2d array `A` with full row and column symmetry: the `lex2(A)` constraint breaks the row symmetries and column symmetries, but not all their compositions: see the examples on slides 28 to 30.

Example (Rotation and reflection symmetry)

A magic square of order n is an $n \times n$ array containing all integers 1 to n^2 exactly once, so that the sums of the rows, columns, and main diagonals are equal (namely $\frac{n^2 \cdot (n^2+1)}{2 \cdot n}$).

For instance, a magic square `M` of order 3 has row sum 15:

2	9	4
7	5	3
6	1	8

Rotation and reflection symmetry-breaking constraint:

```
M[1,1] < M[1,n] /\ M[1,n] < M[n,1] /\ M[1,1] < M[n,n]
```

Structural Symmetry Breaking

Recall the (partitioned) map colouring problems of slide 7:

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

Consider decision variables `A` in the domain `D = 1..k`:

- **Full value symmetry** over the domain `D`:
  The first occurrences of the domain values are ordered:

  ```
  forall(i in 1..k-1)(value_precede(i,i+1,A))
  ```

  or, logically equivalently but better:

  ```
  value_precede_chain(D,A)
  ```

- **Partial value symmetry** over the partitioned domain $\mathtt{D} = \mathtt{D[1]} \cup \mathtt{D[2]} \cup \cdots \cup \mathtt{D[m]}$:

  ```
  forall(i in 1..m)(value_precede_chain(D[i],A))
  ```

Think if dummy values are symmetric to non-dummy ones.

Example (Partial var. sym. + full value sym.; *CP 2006*)

- Make study groups for two sets of five indistinguishable students each. There are six indistinguishable tables.
- The arrays `P` and `M` of decision variables, both with indices in `1..5`, correspond to the students and are to be given table values from the domain `1..6`.
- Variable-symmetry-breaking constraint:

```
increasing(P) /\ increasing(M)
```

- Constraint on the signatures, which are integer pairs:

```
global_cardinality_closed(P,1..6,CP) /\
global_cardinality_closed(M,1..6,CM)
```

- Value-symmetry-breaking constraint using signatures:

```
forall(i in 1..(6-1))
  (lex_greatereq([CP[i],CM[i]],[CP[i+1],CM[i+1]]))
```

Example (continued)

Consider the solution

$$\begin{aligned} \texttt{P} &= [1, 1, 2, 3, 4] \\ \texttt{M} &= [1, 2, 2, 3, 5] \end{aligned}$$

The variable-symmetry-breaking constraint is satisfied:

```
increasing(P) /\ increasing(M)
```

and the value-symmetry-breaking constraint is satisfied:

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

Note that a pointwise ordering would not have sufficed.

Example (continued)

If student `M[5]` is at table 6 instead of table 5, producing a symmetrically equivalent solution because the tables are fully interchangeable:

$$\begin{aligned} \mathtt{P} &= [1,1,2,3,4] \\ \mathtt{M} &= [1,2,2,3,6] \end{aligned}$$

then the value-symmetry-breaking constraint is violated:

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

Example (end)

If students `M[4]` and `M[5]` swap their tables, producing a symmetrically equivalent solution because those students are indistinguishable:

$$\begin{aligned} \mathtt{P} &= [1, 1, 2, 3, 4] \\ \mathtt{M} &= [1, 2, 2, 5, 3] \end{aligned}$$

then the signatures do not change and hence the value-symmetry-breaking constraint remains satisfied, but the variable-symmetry-breaking constraint is violated, because

$$\mathtt{M[1]} \leq \mathtt{M[2]} \leq \mathtt{M[3]} \leq \mathtt{M[4]} \not\leq \mathtt{M[5]}$$

Symmetry Breaking in MiniZinc

Good practice in MiniZinc:
Flag symmetry-breaking constraints using the `symmetry_breaking_constraint` predicate.
This allows backends to handle them differently, if wanted (see Topic 9: Modelling for CBLS):

```
predicate symmetry_breaking_constraint(var bool: c) = c;
```
vs
```
predicate symmetry_breaking_constraint(var bool: c) = true;
```

Examples

```
constraint symmetry_breaking_constraint(increasing(X));
constraint symmetry_breaking_constraint(lex_lesseq(A,B));
```

Especially for MIP backends, try commenting away some symmetry-breaking constraints, as the first definition above of `symmetry_breaking_constraint` is currently used.

Outline

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 proof. I do not have the time.)

In Practice

- Are there 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 = abs(x-y)` can be simplified into `z = x-y` if symmetry breaking requires `x >= y`, thereby eliminating an undesirable disjunction, but upon `symmetry_breaking_constraint(x >= y)` this is incorrect for backends dropping such constraints.

Bibliography

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.

P. Flener, J. Pearson, and M. Sellmann.
Static and dynamic structural symmetry breaking.
Annals of Mathematics and Artif. Intell., 57(1):37–57, Sept. 2009.
(Supersedes our CP 2006 paper with P. Van Hentenryck.)