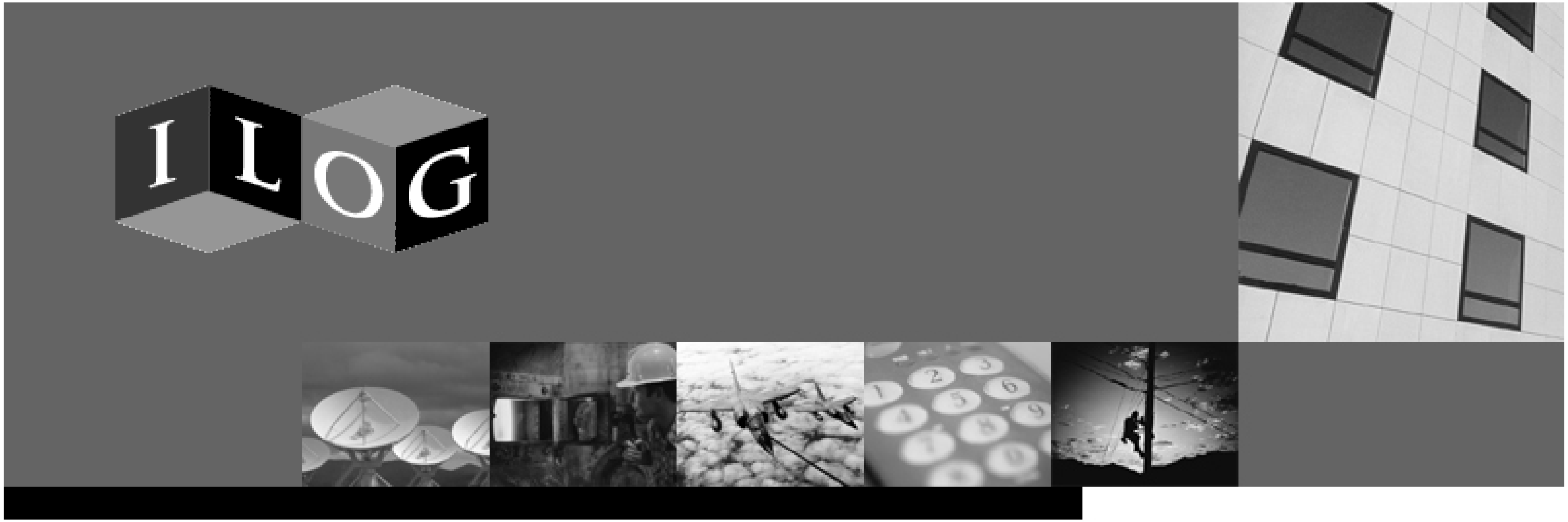


Symmetry breaking revisited

Jean-François Puget

ILOG

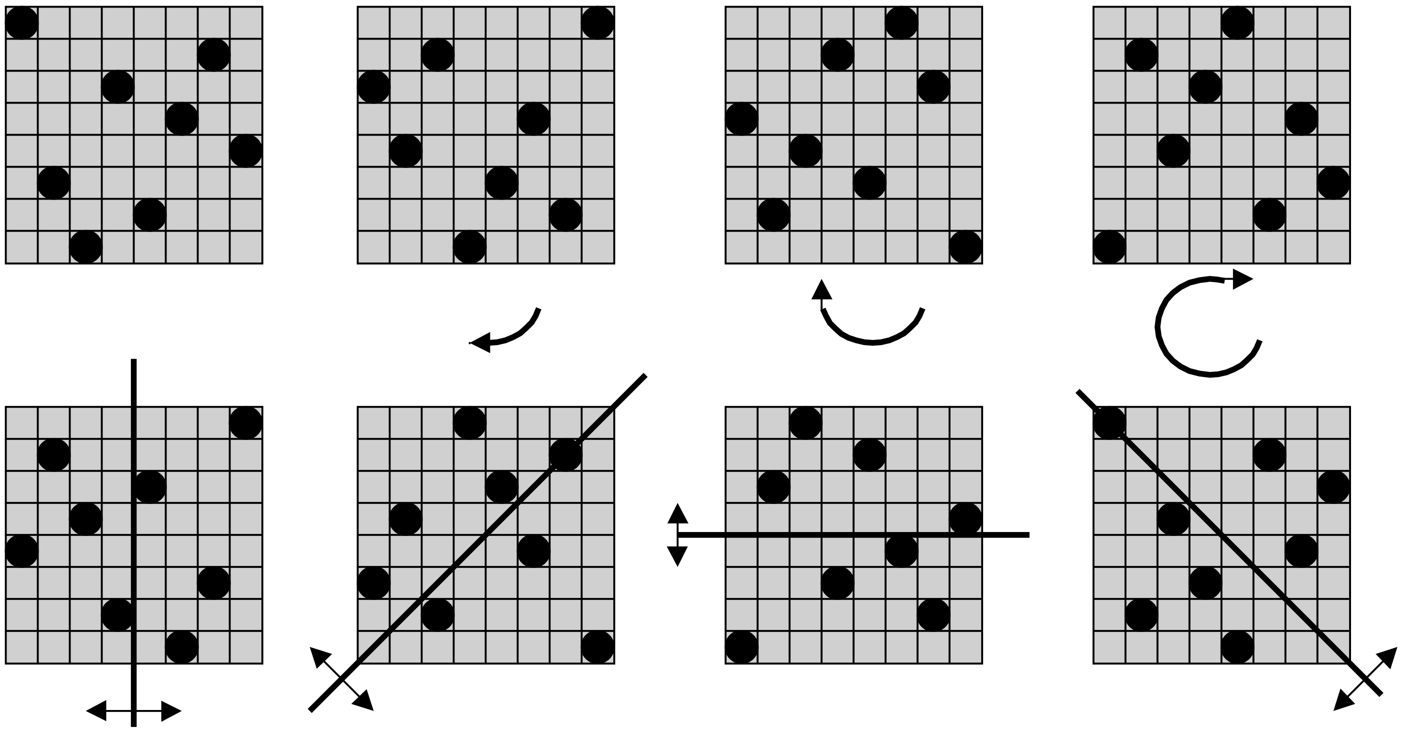
Sept 9, 2002

Outline

- Symmetries in CSP
- Past nodes as nogoods
- Isomorph rejection
- Results
- Future work

Examples

- N queen : 8 symmetries of the square

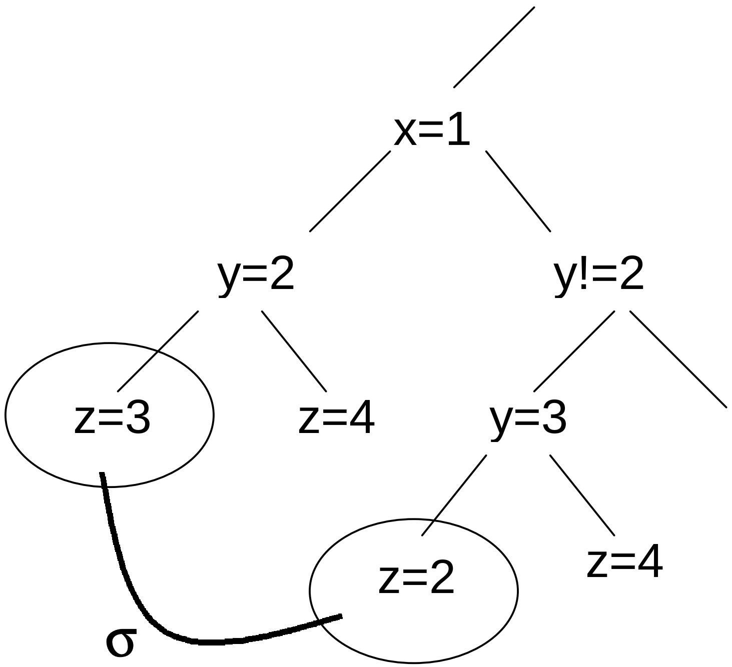


Symmetries

- **Isomorphisms**
  - 1-1 Mappings (bijections) that preserve problem structure.
- **Uniquely defined by how unary decisions are mapped**
  - $\sigma : \quad x_i = a_i \quad \rightarrow \quad x_i' = a_i'$
  - Variables can be permuted
  - Values can be permuted
  - Both
- **Map solutions to solutions**
  - Potentially large number of isomorph variants
- **Map trees search to tree search**
  - The same failure will be repeated many times

Example

- Alldiff(x,y,z), x,y,z in {1, 2, 3, 4}
- Variables can be permuted



x=1,y=3,z=2

is isomorph to

x=1,y=2,z=3

σ:

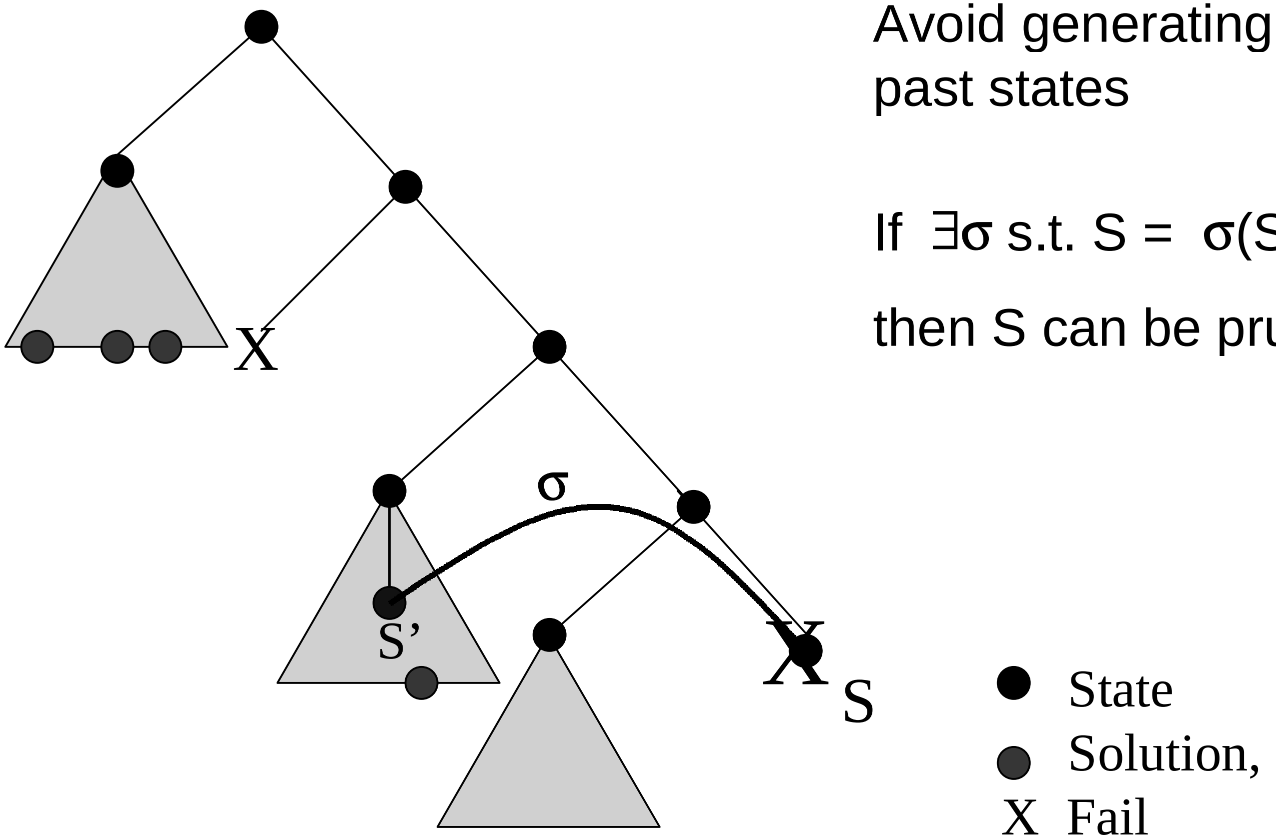
$\forall a \quad x=a \rightarrow x=a$

$\forall a \quad y=a \rightarrow z=a$

$\forall a \quad z=a \rightarrow y=a$

Past states as nogoods

Focacci&Milano, Fahle&al [CP'01]



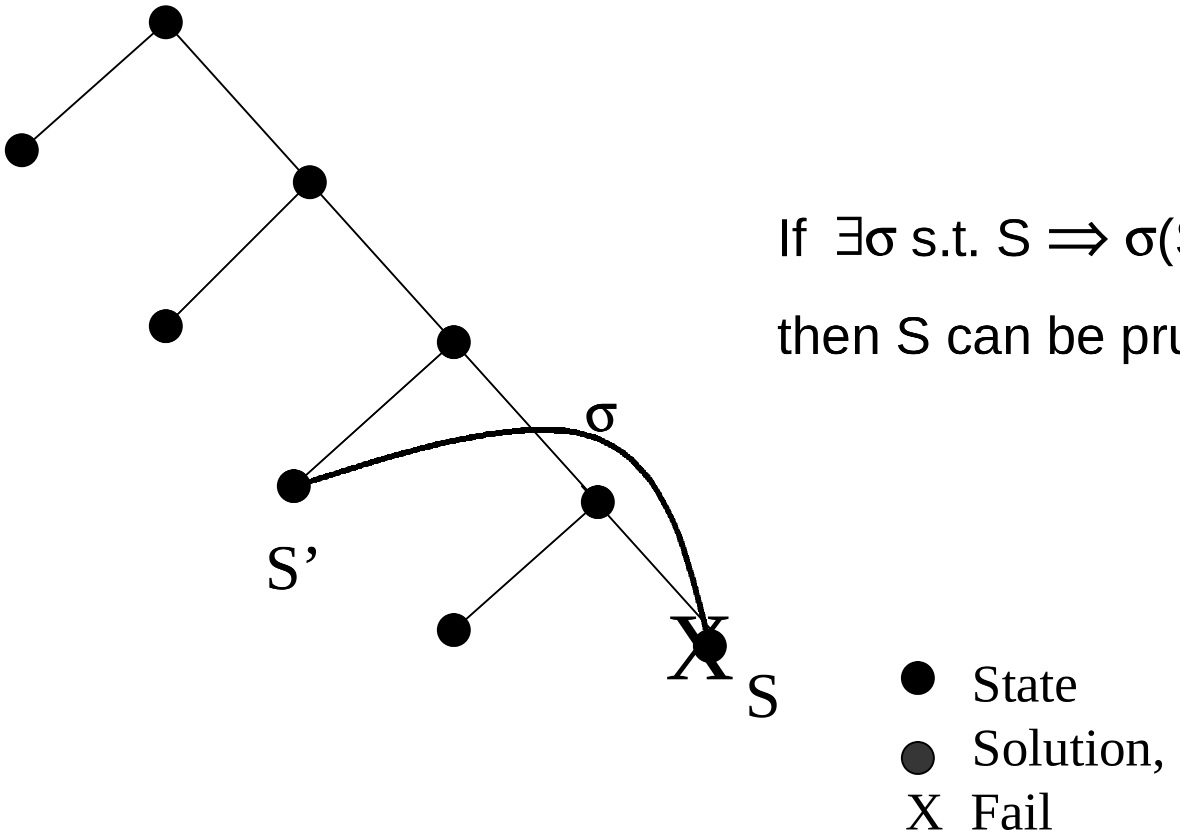
Avoid generating states isomorph to past states

If ∃σ s.t. S = σ(S'), S' past state

then S can be pruned

Generalized nogoods

Only look at the roots of left subtrees



If $\exists\sigma$ s.t. $S \Rightarrow \sigma(S')$, S' left child

then S can be pruned

Nogood entailment

- **Previous work rely on state inclusion**
  - For each node S, check if there exists σ and nogood S' s.t

    $\forall$ x, (domain of x in S) $\subseteq$ σ(domain of x in S')

- **We check if symmetric decisions are entailed :**

  $\exists \sigma , S \Rightarrow \sigma(\bigwedge_i c_i)$

  Where c_i are the decisions leading to the nogood S'

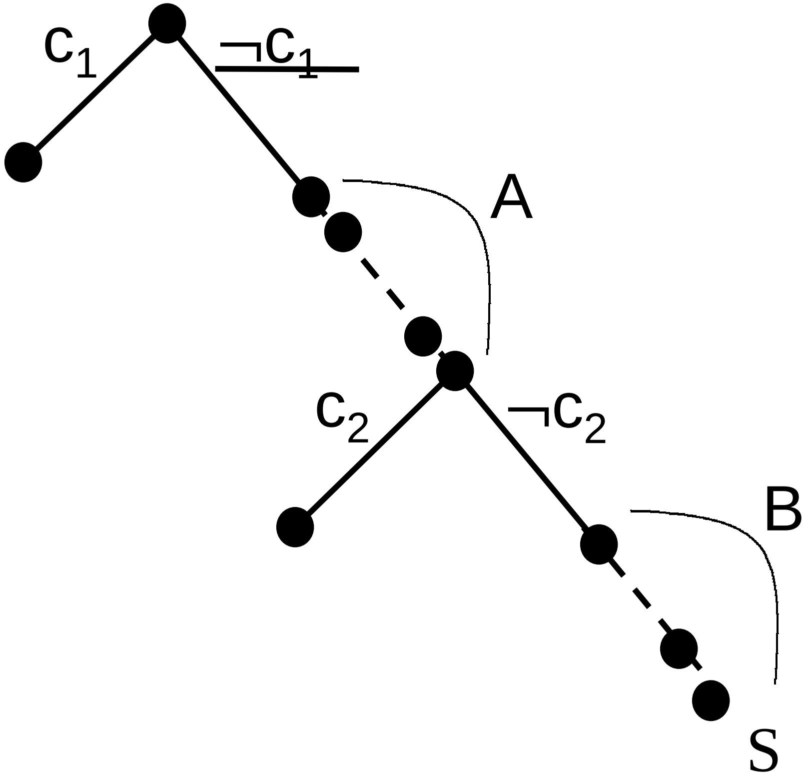
- **Nogood entailment must be checked at each node, for each nogood.**

Decision set as nogoods

Assume 2 nogoods only:

$\mathbf{c_1}$

$\neg \mathbf{c_1} \wedge \mathbf{A} \wedge \mathbf{c_2}$



S is pruned iff

$\exists\sigma_1 \;\; S \Rightarrow \sigma_1 (\mathbf{c_1})$

$\vee$

$\exists\sigma_2 \;\; S \Rightarrow \sigma_2 (\neg \mathbf{c_1} \wedge \mathbf{A} \wedge \mathbf{c_2})$

Can get rid of negative decisions :
S is pruned iff

$\exists\sigma_1 \;\; S \Rightarrow \sigma_1 (\mathbf{c_1})$

$\vee$

$\exists\sigma_2 \;\; S \Rightarrow \sigma(\mathbf{A} \wedge \mathbf{c_2})$

$\mathbf{a \vee (\neg a \wedge b) \;\; \equiv \;\; a \vee b}$

Theoretical results

- **Symmetry breaking search is complete.**
  - For each solution of the original problem, it finds a solution isomorph to it.
- **Symmetry breaking search is correct .**
  - It never finds two isomorph solutions.
- **The proofs do not depend on the search strategy nor on the constraint propagation algorithm**
  - Can be used in conjunction with symmetry breaking constraints
  - Non DFS, parallel search

Isomorph rejection

- **Assume unary decisions**
  - $x_i = a_i$
- **Entailed decisions**

  $\Delta(S) = \{x_i = a_i \mid domain(x_i) = \{a_i\} \text{ in } S\}$
- **Isomorphism test is simpler:**

  $\exists\sigma \quad S \Rightarrow \sigma(c_1 \wedge \ldots \wedge c_k)$

  is equivalent to

  $\exists\sigma \quad \{\sigma(c_1) \wedge \ldots \wedge \sigma(c_k)\} \subseteq \Delta(S)$
- **Complexity**
  - Storage of one nogood is O(1)
  - Number of nogoods is O(nm)

Isomorph rejection (2)

- **For each node, for each nogood for that node, create an auxiliary CSP for computing σ**
  - Variables correspond to decisions of the nogood
  - Values to decisions entailed by the state
  - Constraints restrict σ to be a symmetry of the original CSP
- **Writing symmetries checking as constraint satisfaction is not trivial for the moment.**
  - Subgraph ismorphism on our examples
- **Symmetries are not listed in advance, they are dynamically discovered**

Social Golfer

- **Real world problem : 8-4-10 still open**
  - Smaller instances hard enough
- **Evaluation of symmetry breaking search:**
  - Search for all non isomorph solutions
- **Model**
  - Set variables representing groups of each week.
  - Generation week per week
- **Best or equal results for**

  6-5-6, 6-5-7, 7-3-9, 8-3-10, 9-3-11, 10-3-13, 9-4-8,

  10-4-9, 8-5-5 9-8-3, 10-8-9, 10-9-3, 10-10-3

Results (more results in the paper)

- **Full symmetry breaking (Pentium III 833MHz laptop)**

	5-3-7	5-3-4	5-4-5	5-4-6
solutions	7	13,933	10	0
time (sec)	25.5	3,603	20.4	4.1

- **Partial symmetry breaking**
  - Only used for the first 3 weeks

	5-3-7	5-3-4	5-4-5	5-4-6
solutions	102	353,812	147	0
time (sec)	7.8	105	7.5	3.6

- **Order(s) of magnitude faster than previous work**

ILOG™

bibd(v, b, r, k, λ) : preliminary results

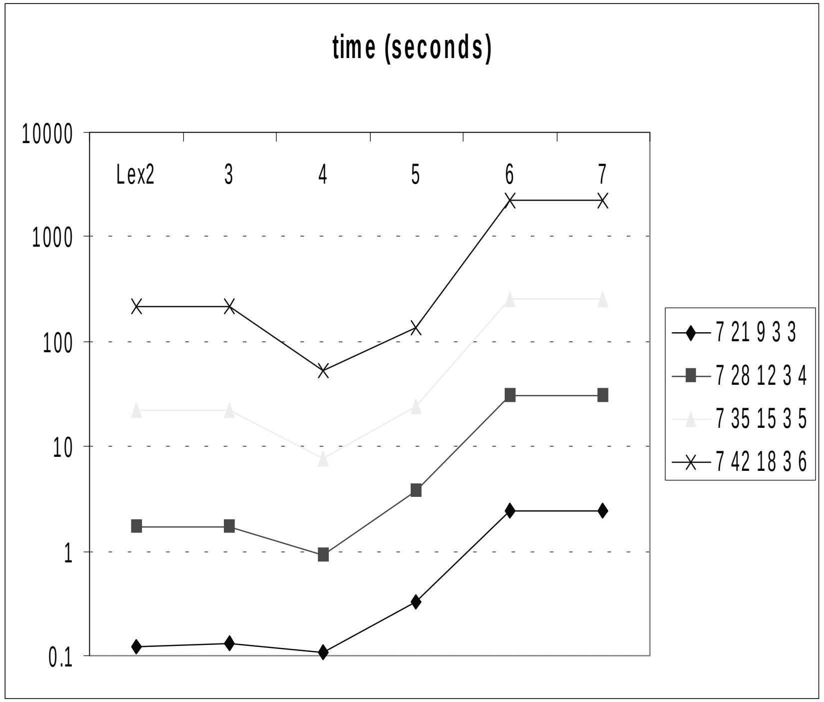
- **Simple model**
  - v x b   matrix of 0-1 variables
  - Sum of each row = r
  - Sum of each column = k
  - Inner product of any two row = λ
  - Row by row generation
  - Rows can be permuted
  - Columns can be permuted

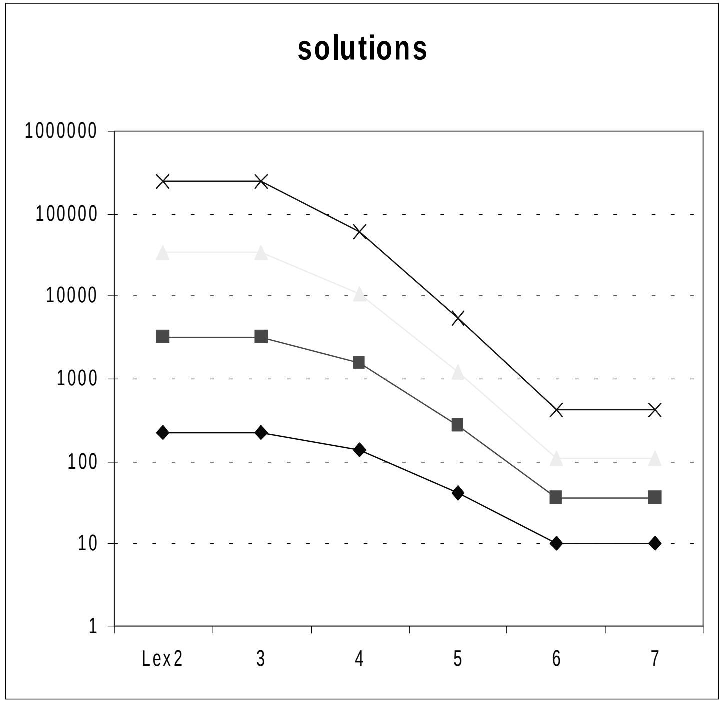
- **Finding one solution is often easy**
  - Solves each instance of [Messeguer&Torras 99] within a 2 seconds

- **Finding all (non isomorph) solutions is harder**
  - Lex^2 is quite effective [Flener & al, CP'02]
  - Finds all solutions of small instances within a second

bibd(7, 7t, 3t, 3, t) family

- ❑ **Lex² + symmetry breaking search on first n rows**





bibd(15, 35, 7, 3, 1) (> 10^{52} symmetries)

- Lex²

Solutions	Nodes	Time (sec)
32,127,296	117,782,182	75,999

> 21 hours

- Lex² + symmetry breaking search

80	76,911	13,721

< 4 hours

- Lex² + symmetry breaking on first 10 rows

157,312	412,312	438

< 8 minutes

Conclusion

- **Simple and powerful formalization**
  - Left children as nogoods
  - Correctness and completeness results
- **Applies to any search strategy and propagation algorithm**
  - Non depth first search, parallel search
  - O(nm) space per open node, O(nm) space for DFS
  - Can be used with symmetry breaking constraints
- **Improves over SBDD and Cut Generation [CP’01]**
- **Improves over Lex^2 on BIBD**
- **Domain filtering instead of generate and test**
  - Can be implemented with an auxiliary CSP
  - On the golfer, reduces number of nodes, but is 2 times slower

Future work

- **Isomorphism test is too costly**
  - Done at each node, it dominates running time
  - Efficient domain specific tests are possible [Barnier & Brisset CP'02]

- **Symmetry definitions**
  - Isomorphism test could use known symmetries
  - Use group generators? [Gent & al CP'02]

- **Combination with SBDS**
  - Domain filtering

- **Other real world problems**
  - Time tabling, rostering

ILOG™