Symmetry in SAT (and ASP and CP): Breaking the right symmetries

Bart Bogaerts

Aalto University

December 8, 2015

Main Reference

Jo Devriendt, Bart Bogaerts, and Maurice Bruynooghe.
BreakIDGlucose: On the importance of row symmetry.
In *Proceedings of the Fourth International Workshop on the Cross-Fertilization Between CSP and SAT (CSPSAT)*, 2014.

Take Home Message

Take Home Message

Don't go implementing any of the methods I describe in this talk!
(unless you really have to)

Content

1. Motivation
2. Symmetries
   - Constraint Programming
   - SAT Solving
3. Row Interchangeability Detection: Take 1
4. Row Interchangeability Detection: Take 2
5. Row Interchangeability Detection: Take 3
6. Final thoughts

Motivation

- 2012
- "Symmetry Propagation" for SAT [Devriendt et al., 2012]
- Add symmetric variants of learnt clauses lazily
- Performance: good (compared to symmetry breaking)

Motivation

- 2012
- "Symmetry Propagation" for SAT [Devriendt et al., 2012]
- Add symmetric variants of learnt clauses lazily
- Performance: good (compared to symmetry breaking)

Except...

- On the pigeonhole problem
- Unexplainable difference: (bad) luck?

Motivation

- 2012
- "Symmetry Propagation" for SAT [Devriendt et al., 2012]
- Add symmetric variants of learnt clauses lazily
- Performance: good (compared to symmetry breaking)

Except...

- On the pigeonhole problem
- Unexplainable difference: (bad) luck?

Except...

- If we permute the variables...

Content

1 Motivation

2 Symmetries
- Constraint Programming
- SAT Solving

3 Row Interchangeability Detection: Take 1

4 Row Interchangeability Detection: Take 2

5 Row Interchangeability Detection: Take 3

6 Final thoughts

CP perspective

- CSP: (V, D, C)
- assignment $\alpha : (V \to D)$
- solution satisfies constraints

For example:

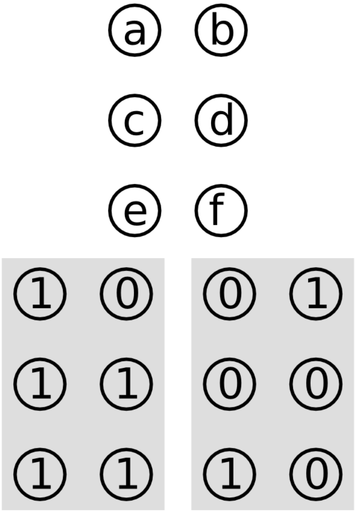
- $V = \{a, b, c, d, e, f\}$
- $D = \{0, 1\}$
- $C = \{b \leq 0\}$
- $\alpha = \{a = 1, b = 0, c = 1,$
  $d = 1, e = 1, f = 1\}$
- $\alpha' = \{a = 0, b = 1, c = 0,$
  $d = 0, e = 1, f = 0\}$

CP perspective

- CSP: (V, D, C)
- assignment $\alpha : (V \rightarrow D)$
- solution satisfies constraints

For example:

- $V = \{a, b, c, d, e, f\}$
- $D = \{0, 1\}$
- $C = \{b \leq 0\}$
- $\alpha = \{a = 1, b = 0, c = 1,$
  $d = 1, e = 1, f = 1\}$
- $\alpha' = \{a = 0, b = 1, c = 0,$
  $d = 0, e = 1, f = 0\}$

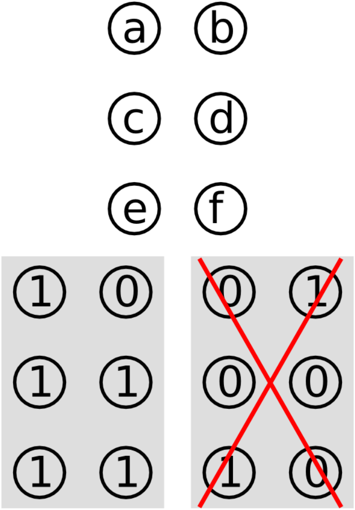


CP perspective

- CSP: (V, D, C)
- assignment $\alpha : (V \to D)$
- solution satisfies constraints

For example:

- $V = \{a, b, c, d, e, f\}$
- $D = \{0, 1\}$
- $C = \{b \leq 0\}$
- $\alpha = \{a = 1, b = 0, c = 1, d = 1, e = 1, f = 1\}$
- $\alpha' = \{a = 0, b = 1, c = 0, d = 0, e = 1, f = 0\}$

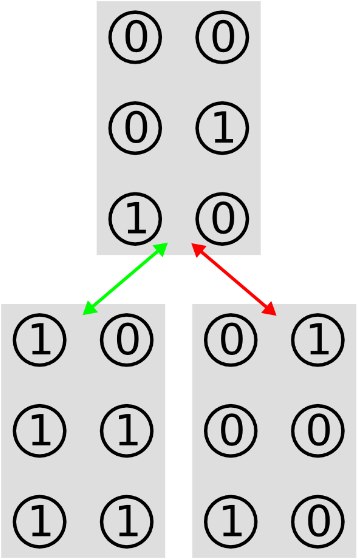


CP perspective on symmetry

$S : (V \to D) \to (V \to D)$
symmetry S is a permutation of the set of assignments preserving satisfaction to all constraints
Recall:

- $V = \{a, b, c, d, e, f\}$
- $D = \{0, 1\}$
- $C = \{b \leq 0\}$

CP perspective on symmetry

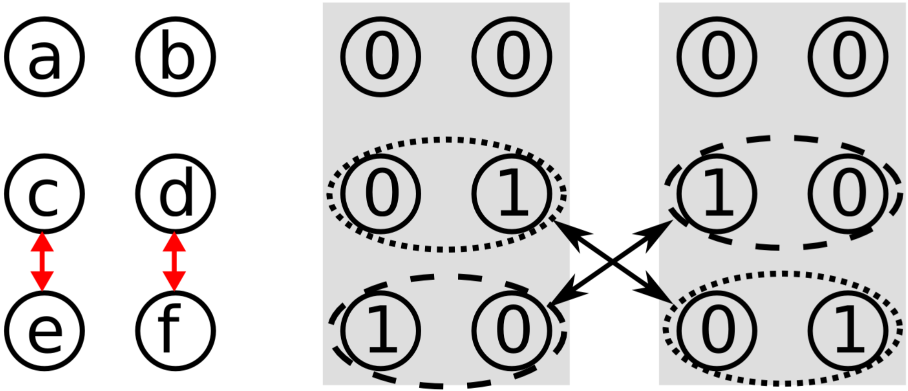


$S : (V \to D) \to (V \to D)$
symmetry S is a permutation of the set of assignments preserving satisfaction to all constraints
Recall:

- $V = \{a, b, c, d, e, f\}$
- $D = \{0, 1\}$
- $C = \{b \leq 0\}$

CP perspective on symmetry

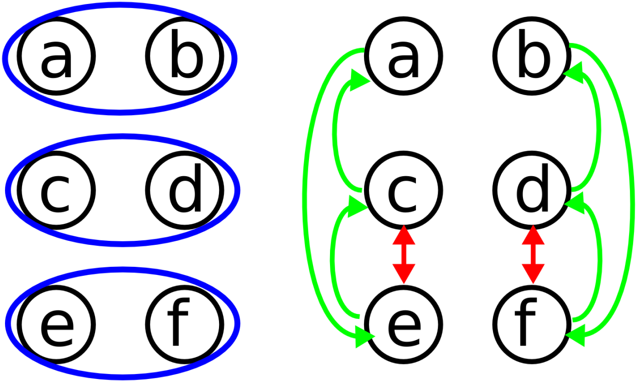
Common restriction:
variable symmetry $S_P : \alpha \mapsto \alpha \circ P$
induced by variable permutation $P : V \to V$
For example: $P = (ce)(df)$



CP perspective on row symmetry

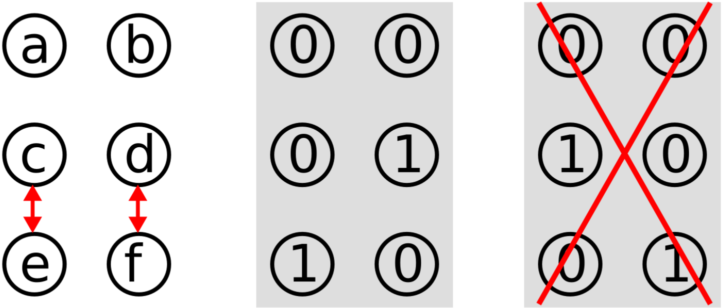
Row symmetry:

- special case of variable symmetry
- assumption: V is ordered as a matrix M
- P permutes the rows of M
- CSP is row-interchangeable for M iff all permutations on the rows of M induce a symmetry



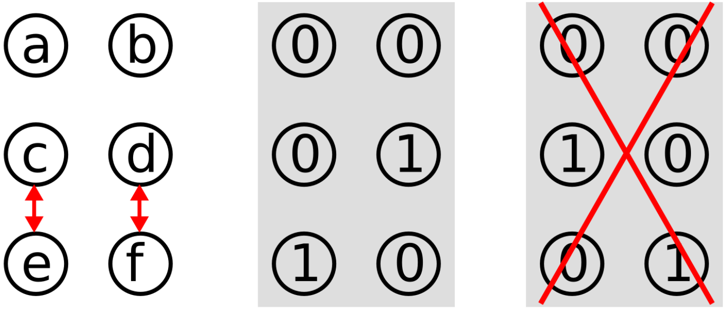
CP perspective on row symmetry

Symmetry breaking: speed up search by adding extra constraints to remove symmetrical assignments from the assignment space.



CP perspective on row symmetry

Symmetry breaking: speed up search by adding extra constraints to remove symmetrical assignments from the assignment space.



Complete symmetry breaking: maximal number of symmetric assignments removed while retaining soundness.

CP result: row interchangeability can be broken completely by enforcing lexicographic order on rows. [Flener et al., 2002]

SAT perspective

- SAT instance: $(V, \{0,1\}, C)$, C is a CNF
- assignment $\alpha : (V \rightarrow D)$
- solution satisfies C

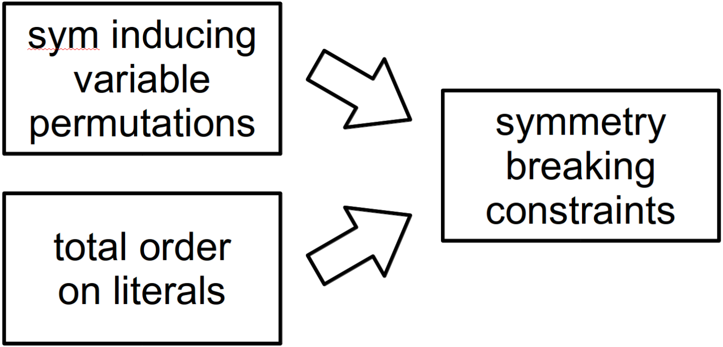
SAT perspective on symmetry

- SAT instance: $(V, \{\mathbf{t}, \mathbf{f}\}, C)$, C is a CNF
- assignment $\alpha : (V \to D)$
- solution satisfies C
- A *variable symmetry* is a permutation P of V such that $\alpha \models C$ iff $\alpha \circ P \models V$

(this is exactly the same as in CP)

SAT perspective on breaking symmetry

General symmetry breaking in SAT as per Saucy+Shatter [Aloul et al., 2006]:

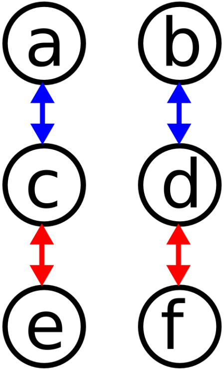


For each variable permutation P (**in some set of generators**) and for each variable v, add the lex-leader constraint
$(\forall v' \prec v : v' = P(v')) \Rightarrow v \leq P(v)$.

Can Shatter completely break row interchangeability?

- variable permutations: P_{12}, P_{23}
  that induce row interchangeability
- order on variables: $a \prec b \prec c \prec d \prec e \prec f$
- symmetry breaking constraints:
  $a \leq c$, $a = c \Rightarrow b \leq d$,
  $c \leq e$, $c = e \Rightarrow d \leq f$

These constraints actually state that the rows of each solution must be lexicographically ordered!

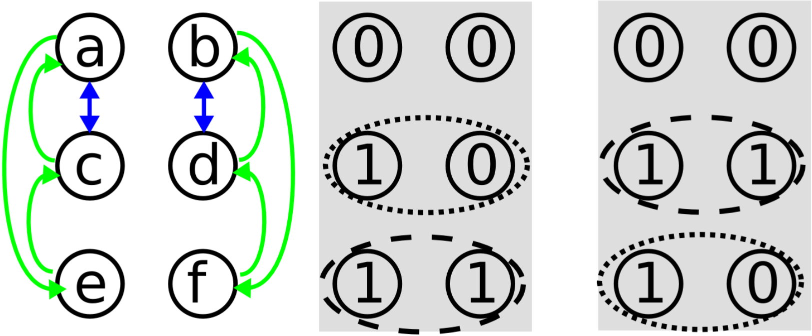


Row interchangeability can be completely broken by standard SAT symmetry breaking methods, given the right variable permutations and variable ordering.

What can go wrong?

- variable permutations: P_{12}, P_{321} that induce row interchangeability
- order on variables: $a \prec b \prec c \prec d \prec e \prec f$
- symmetry breaking constraints:
  $a \leq c$, $a = c \Rightarrow b \leq d$,
  $a \leq e$, $a = e \Rightarrow b \leq f$,
  $a = e \wedge b = f \Rightarrow c \leq a$,$a = e \wedge b = f \wedge c = a \Rightarrow d \leq b$

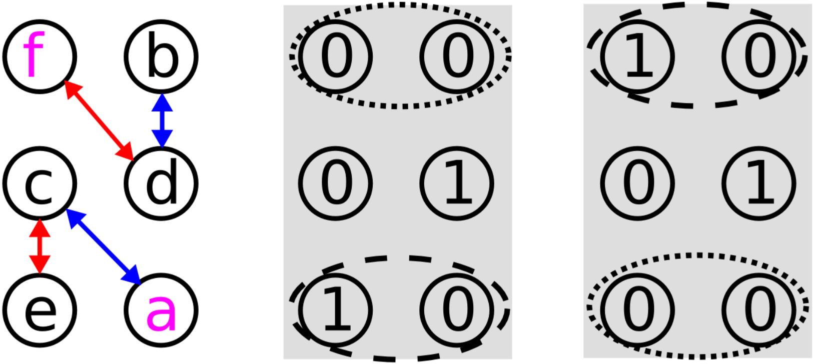
These do not represent lexicographic row ordering constraint



What can go wrong?

- variable permutations: P_{12}, P_{23}
  that induce row interchangeability
- order on variables: $f \prec b \prec c \prec d \prec e \prec a$
- symmetry breaking constraints:
  $b \leq d$, $b = d \Rightarrow c \leq a$,
  $f \leq d$, $f = d \Rightarrow c \leq e$

These do not represent lexicographic row ordering constraint



SAT perspective: conclusion

To completely break row interchangeability:

- need for right generator symmetries
- need for right ordering on variables

Easy when you know the matrix structure of the variables...

How to detect matrix structure of variables in a CNF?
How to detect row interchangeability in a CNF?

Motivation

- 2012
- "Symmetry Propagation" for SAT [Devriendt et al., 2012]
- Add symmetric variants of learnt clauses lazily
- Performance: good (compared to symmetry breaking)

Except...

- On the pigeonhole problem
- Unexplainable difference: (bad) luck?

Except...

- If we permute the variables...

Pigeon Hole

- SAT encoding of the CSP (V, D, C)
- $V = \{1..n\}, D = \{1..n-1\}$
- $C = \{\forall d \in D : \#\{v \mid \alpha(v) = d\} \leq 1\}$.
- (Boolean encoding of $\alpha(i)$ is a "row")

Pigeon Hole

- SAT encoding of the CSP (V, D, C)
- $V = \{1..n\}, D = \{1..n-1\}$
- $C = \{\forall d \in D : \#\{v \mid \alpha(v) = d\} \leq 1\}$.
- (Boolean encoding of $\alpha(i)$ is a "row")

- Any permutation of variables induces a symmetry
- Saucy: generators $(12), (23), (34), (45), \ldots$
- with order on variables $1 \prec 2 \prec 3 \prec \ldots$: breaking constraints

$$\alpha(1) \leq \alpha(2), \alpha(2) \leq \alpha(3), \alpha(3) \leq \alpha(4) \ldots$$

- Breaks the symmetry group completely (no two symmetric assignments satisfy this constraint)

Pigeon Hole

- SAT encoding of the CSP (V, D, C)
- $V = \{1..n\}, D = \{1..n-1\}$
- $C = \{\forall d \in D : \#\{v \mid \alpha(v) = d\} \leq 1\}$.
- (Boolean encoding of $\alpha(i)$ is a "row")

- Any permutation of variables induces a symmetry
- Saucy: generators $(12), (23), (34), (45), \ldots$
- with order on variables $1 \prec 2 \prec 3 \prec \ldots$: breaking constraints

$$\alpha(1) \leq \alpha(2), \alpha(2) \leq \alpha(3), \alpha(3) \leq \alpha(4) \ldots$$

- Breaks the symmetry group completely (no two symmetric assignments satisfy this constraint)
- however... with order $2 \prec 1 \prec 4 \prec 3 \prec \ldots$:

$$\alpha(2) \leq \alpha(1), \alpha(2) \leq \alpha(3), \alpha(4) \leq \alpha(3), \ldots$$

Content

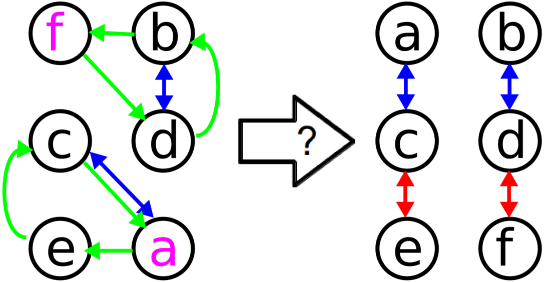
1. Motivation
2. Symmetries
   - Constraint Programming
   - SAT Solving
3. Row Interchangeability Detection: Take 1
4. Row Interchangeability Detection: Take 2
5. Row Interchangeability Detection: Take 3
6. Final thoughts

Row Interchangeability Detection

Problem statement:

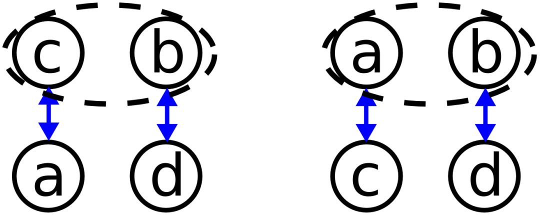
Row Interchangeability Detection (RID)
Given a CNF, find a maximal set of variables that form a variable matrix with interchangeable rows.

Chosen problem perspective: start from detected symmetry group.



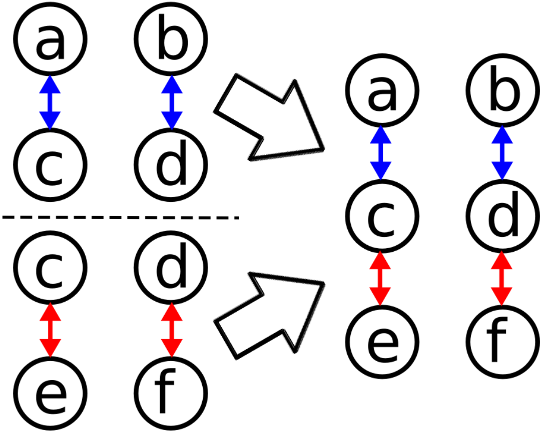
Involution symmetries form rows

A permutation P for which $P^2 = I$, is an involution. Each variable involution forms multiple interchangeable row matrices, with only 2 rows:



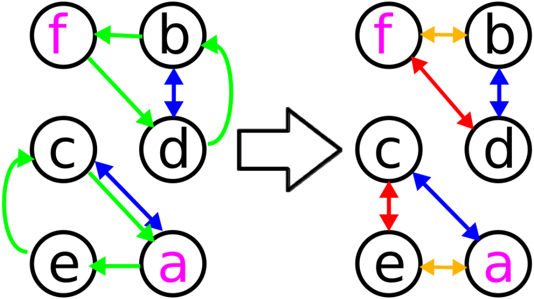
Involution symmetries form rows

Compatible involutions can be combined to row interchangeable matrices with more than 2 rows:



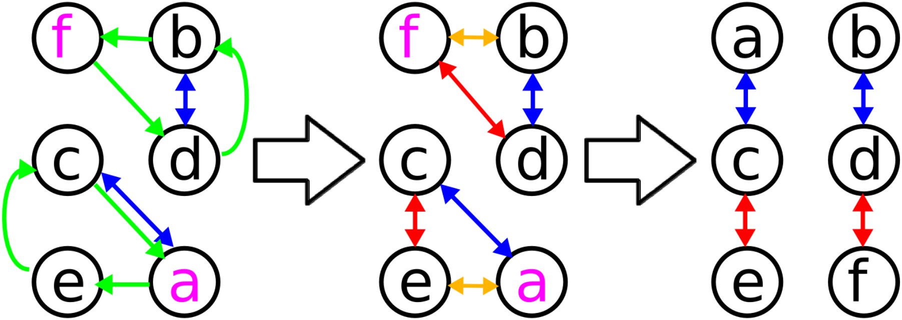
Heuristically search for involutions

Start from generator symmetries returned by Saucy. Compose symmetries to form *small* involutions.



Matrix structure detection by involutions

Combining involution generation & matrix extraction solves RID:



Can be extended to the piecewise case, where multiple disjoint row interchangeable variable matrices exist for one problem.

Piecewise Row Interchangeability Detection – the **PRID** problem.

Working implementation: BreakIDGlucose

- extends Shatter with row interchangeability handling
  - use Saucy to detect symmetry inducing variable permutations
  - new: generate involutions to solve PRID
  - new: add row involutions to set of variable permutations
  - new: adjust order on variables as per detected rows
  - add Shatter's lex-leader constraints
- used Glucose 2.1 as SAT solving engine
- obtained gold medal at 2013's SAT competition hard combinatorial track
- could only solve 5 out of 14 two-pigeon-hole instances

Complete row interchangeability breaking is relevant in SAT
Improvement possible with better detection of matrix structure – better answer to PRID.

Content

1. Motivation
2. Symmetries
   - Constraint Programming
   - SAT Solving
3. Row Interchangeability Detection: Take 1
4. Row Interchangeability Detection: Take 2
5. Row Interchangeability Detection: Take 3
6. Final thoughts

Working implementation 2: BreakIDGlucose2

- extends Shatter with row interchangeability handling
  - use Saucy to detect a set G of symmetry inducing variable permutations
  - new: search for $g_1, g_2 \in G$: two "matching" involutions (rows r_1, r_2 and r_3)
  - new: for each $g \in G$, $g(r_i)$ is a candidate row: check whether $r_1 \leftrightarrow g(r_i)$ is a symmetry
  - new: use Saucy to find more permutations that do not permute rows $r_2, r_3, \ldots$
  - new: continue extending the row-interchangeability matrix
  - new: adjust order on variables as per detected rows
  - add Shatter's lex-leader constraints
- used Glucose 4.0 as SAT solving engine
- obtained 10th place out of 28 in the 2015 SAT Race (best of all Glucose variants)

Content

1 Motivation

2 Symmetries
- Constraint Programming
- SAT Solving

3 Row Interchangeability Detection: Take 1

4 Row Interchangeability Detection: Take 2

5 Row Interchangeability Detection: Take 3

6 Final thoughts

Work in progress: tackling the actual problem

Two main directions

- Modify Saucy to give "the right" generators
- Use algebraic tools (e.g., GAP) to modify the set of generators

Content

1 Motivation

2 Symmetries
- Constraint Programming
- SAT Solving

3 Row Interchangeability Detection: Take 1

4 Row Interchangeability Detection: Take 2

5 Row Interchangeability Detection: Take 3

6 Final thoughts

Where does row interchangeability in SAT come from?

SAT encodings of

- variable interchangeable CSP's
- value interchangeable CSP's
- row interchangeable CSP's
- relational model generation problems where relation $R : D_1 \times \ldots \times D_n$ has to be found and some disjoint D_i contains interchangeable elements.
  For example: the IDP system.

Note the triviality of solving the PRID problem in such a high level language!

Take Home Message

Take Home Message

Don't go implementing any of the methods I describe in this talk!
(unless you really have to)

"There are no CNF problems" (P.J. Stuckey, 2013)

Thanks for your attention!
Questions?

Aloul, F. A., Sakallah, K. A., and Markov, I. L. (2006).
Efficient symmetry breaking for Boolean satisfiability.
IEEE Transactions on Computers, 55(5):549–558.

Devriendt, J., Bogaerts, B., De Cat, B., Denecker, M., and Mears, C. (2012).
Symmetry propagation: Improved dynamic symmetry breaking in SAT.
In *IEEE 24th International Conference on Tools with Artificial Intelligence, ICTAI 2012, Athens, Greece, November 7-9, 2012*, pages 49–56.

Flener, P., Frisch, A. M., Hnich, B., Kiziltan, Z., Miguel, I., Pearson, J., and Walsh, T. (2002).
Breaking row and column symmetries in matrix models.
In Hentenryck, P., editor, *Principles and Practice of Constraint Programming - CP 2002*, volume 2470 of *LNCS*, pages 462–477. Springer Berlin Heidelberg.