Exploitation of dynamic symmetries for solving SAT problems

Thèse de doctorat de Sorbonne Université

Hakan METIN

Rapporteurs:

PASCAL FONTAINE	Maître de conférences, Université de Liège
LAURE PETRUCCI	Professeur, Université Paris 13

Reviewers:

JEAN-MICHEL COUVREUR	Professeur, Université d'Orléans
EMMANUELLE ENCRENAZ	Maître de conférences, Sorbonne Université

Directors:

SOUHEIB BAARIR	Maître de conférences, Université Paris Nanterre
FABRICE KORDON	Professeur, Sorbonne Université

SCIENCES SORBONNE UNIVERSITÉ

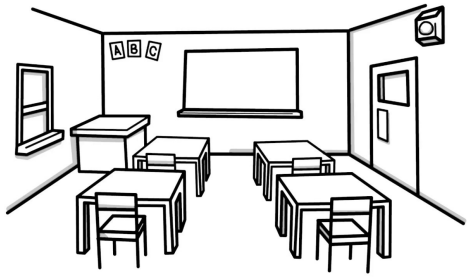
LIP6

Motivation

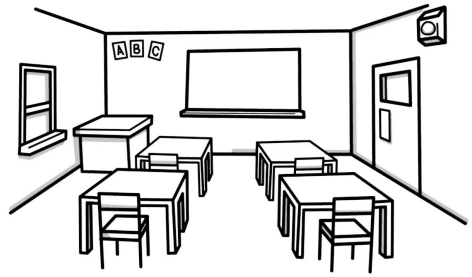
SATisfiability is widely used in different domains

- Artificial intelligence (planning, games, ...)
- Bioinformatics (haplotype inference, ...)
- Security (cryptanalysis, inversion attack on hash , ...)
- Computationally hard problems (graph coloring, ...)
- Formal methods (hardware model checking, ...)

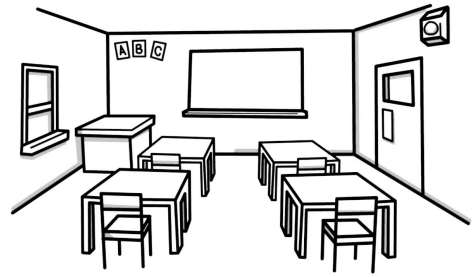
SAT an example



1

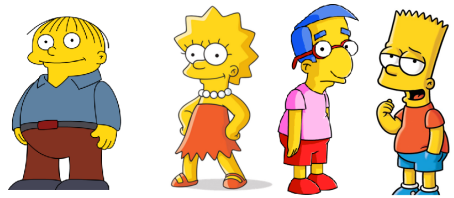


2

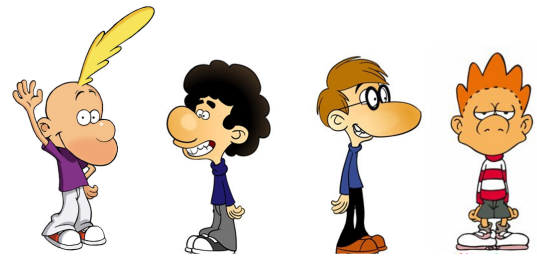


3

A



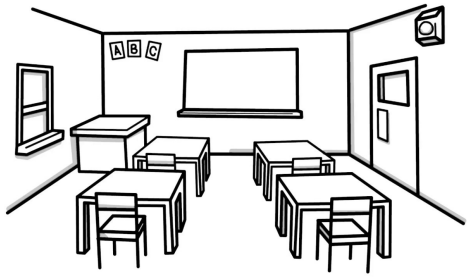
B

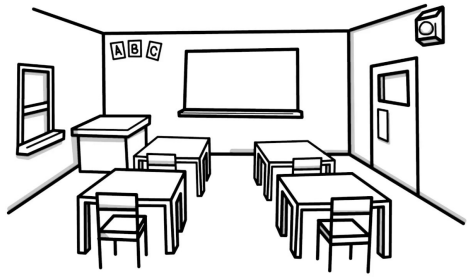


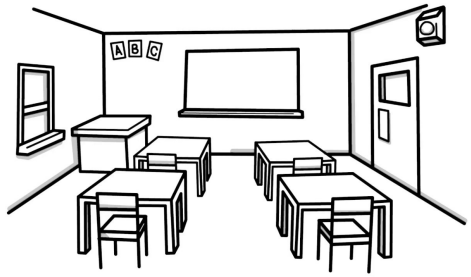
C

Is it possible to attribute each group to a classroom?

SAT an example



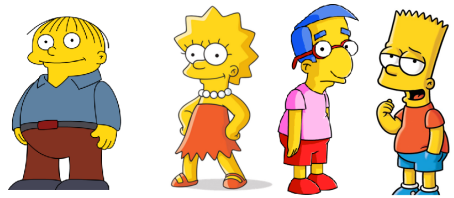


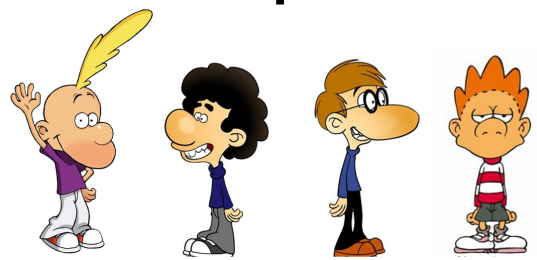


1

2

3





A

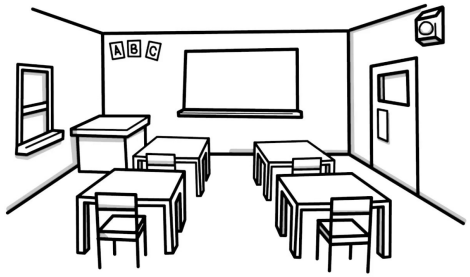
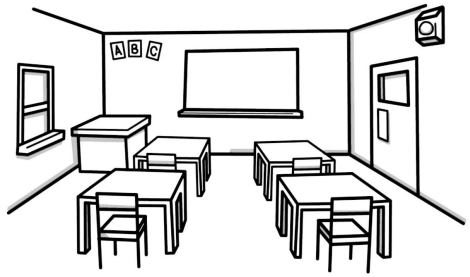
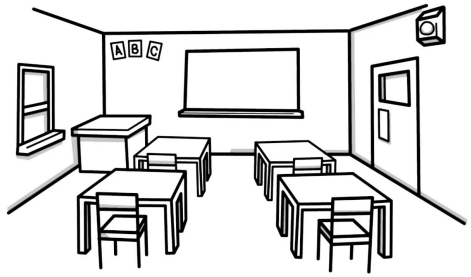
B

C

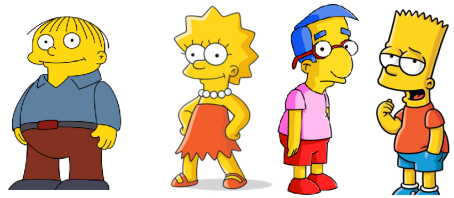
Is it possible to attribute each group to a classroom?

YES!

SAT an example

1 2 3

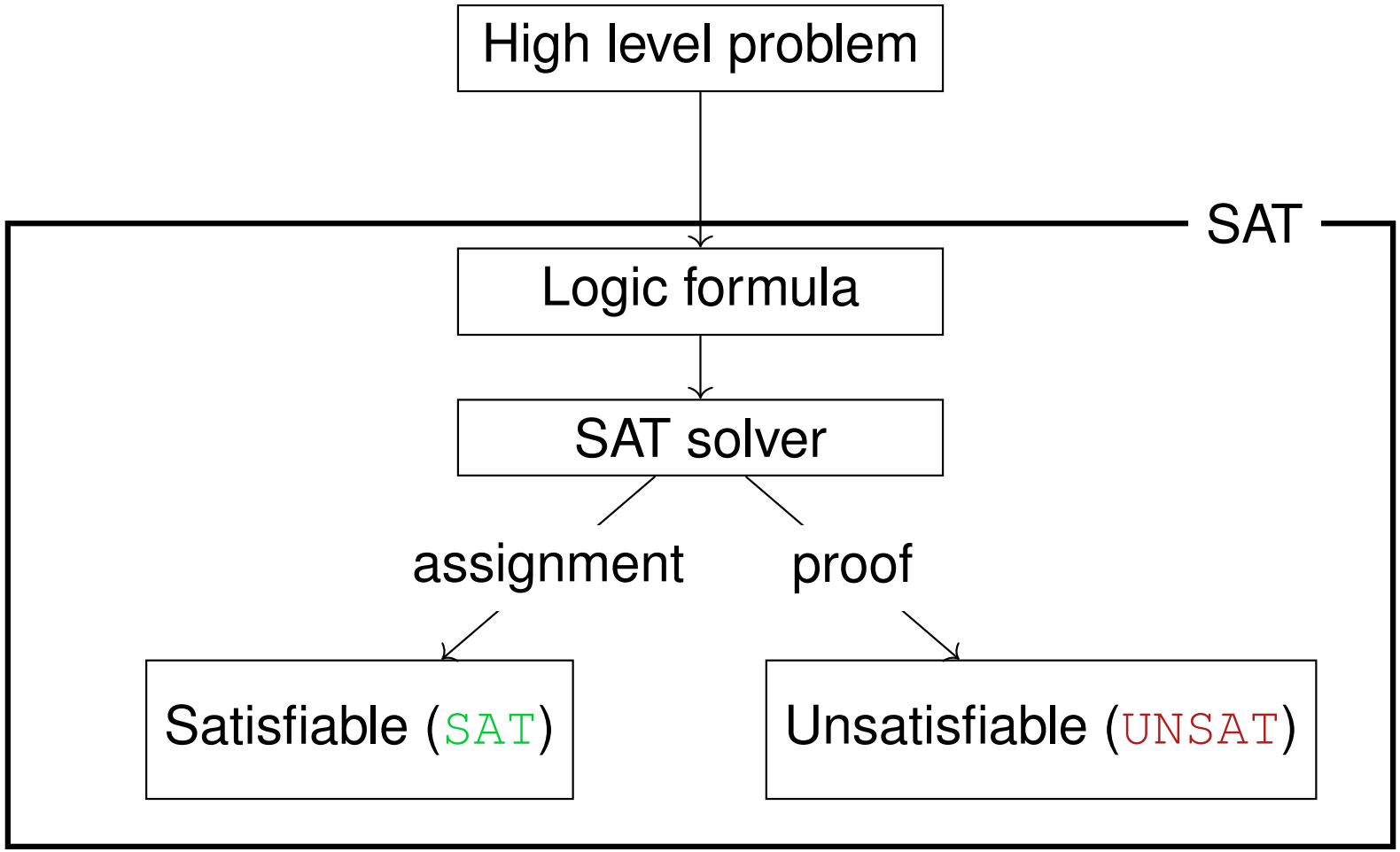




A B C

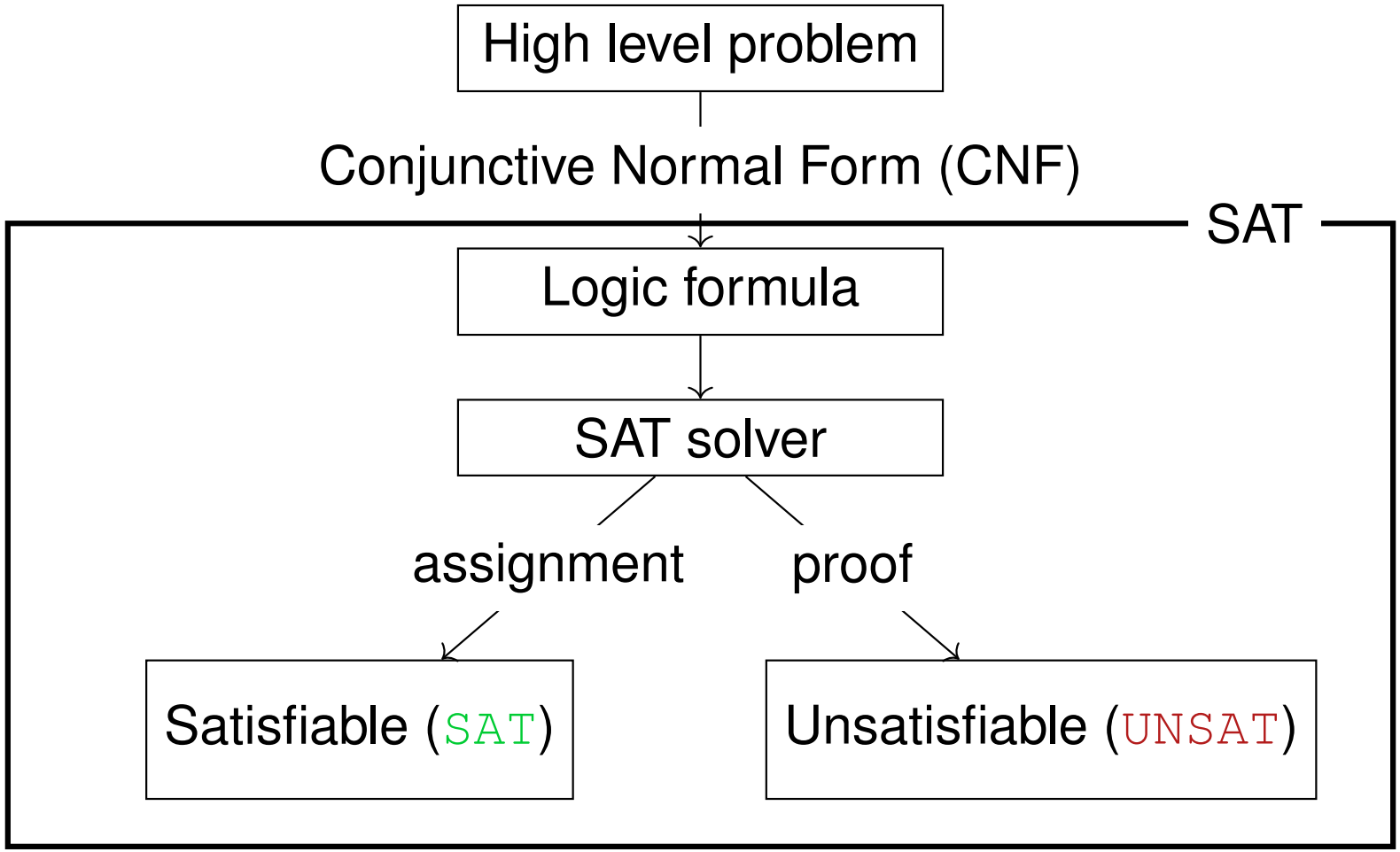
Is it possible to attribute each group to a classroom?

YES! Many solutions

Boolean SATisfiability problem



Boolean SATisfiability problem

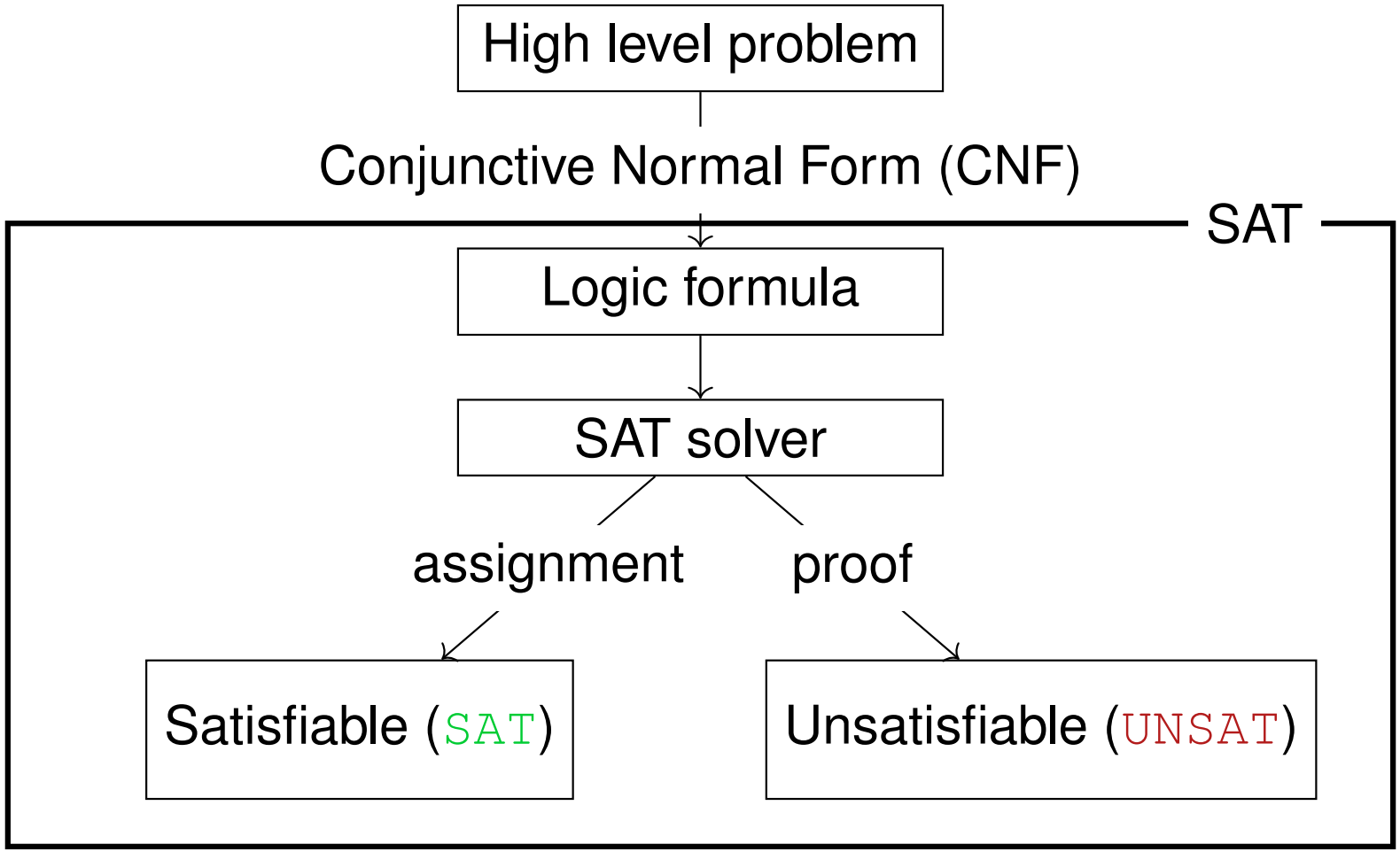


CNF representation:

$$\underbrace{(x_1 \vee x_2 \vee \neg x_3)}$$

Clause with literals $x_1, x_2, \neg x_3$

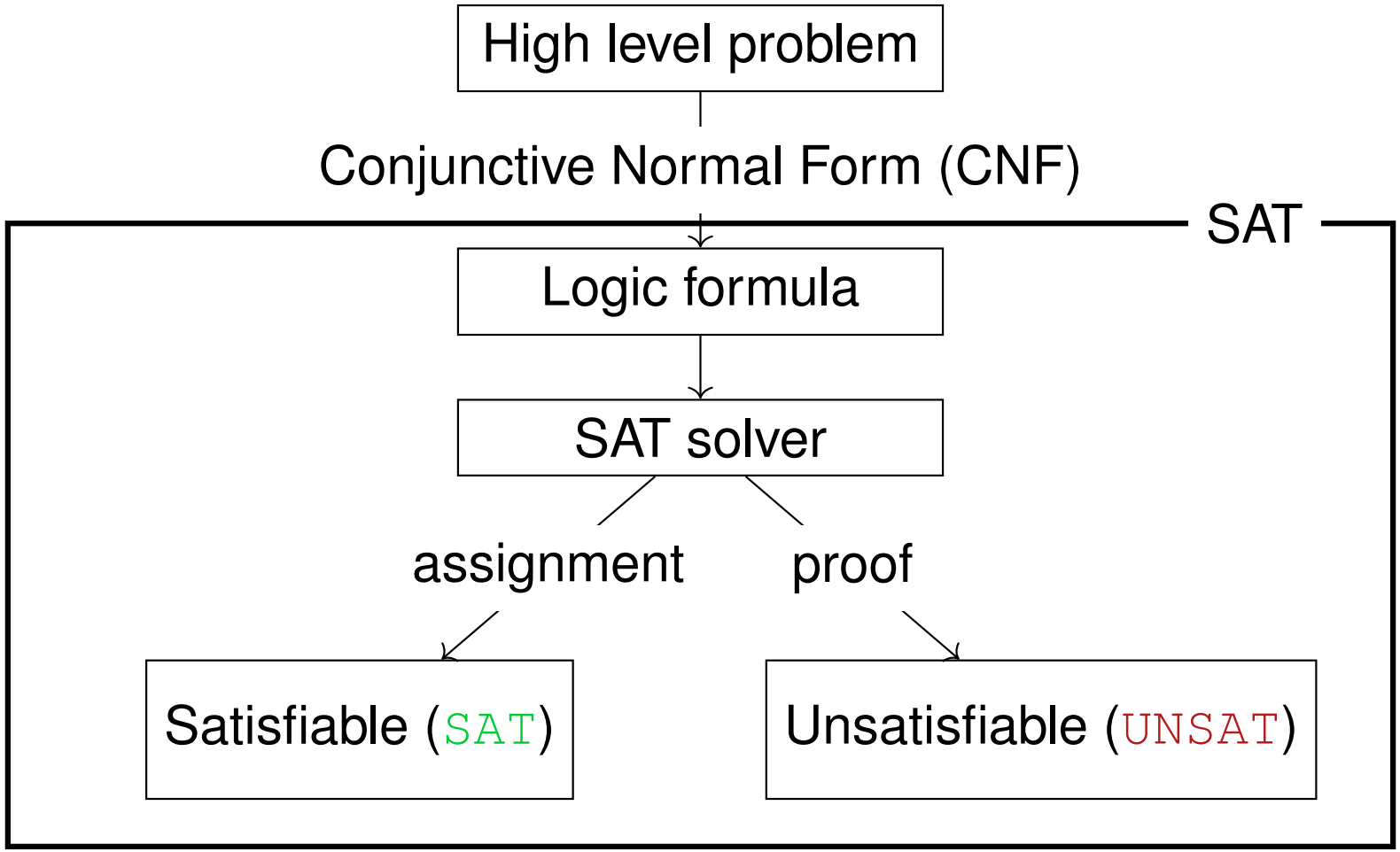
Boolean SATisfiability problem



CNF representation:

$$\overbrace{\underbrace{(x_1 \vee x_2 \vee \neg x_3)}_{\textit{Clause}} \wedge (\neg x_1 \vee \neg x_2) \wedge (x_2 \vee \neg x_4)}^{\text{Formula (CNF)}}$$

Boolean SATisfiability problem



Clause representation as a set:

$$(x_1 \vee x_2 \vee \neg x_3) \rightarrow \{x_1, x_2, \neg x_3\}$$

SAT Solving

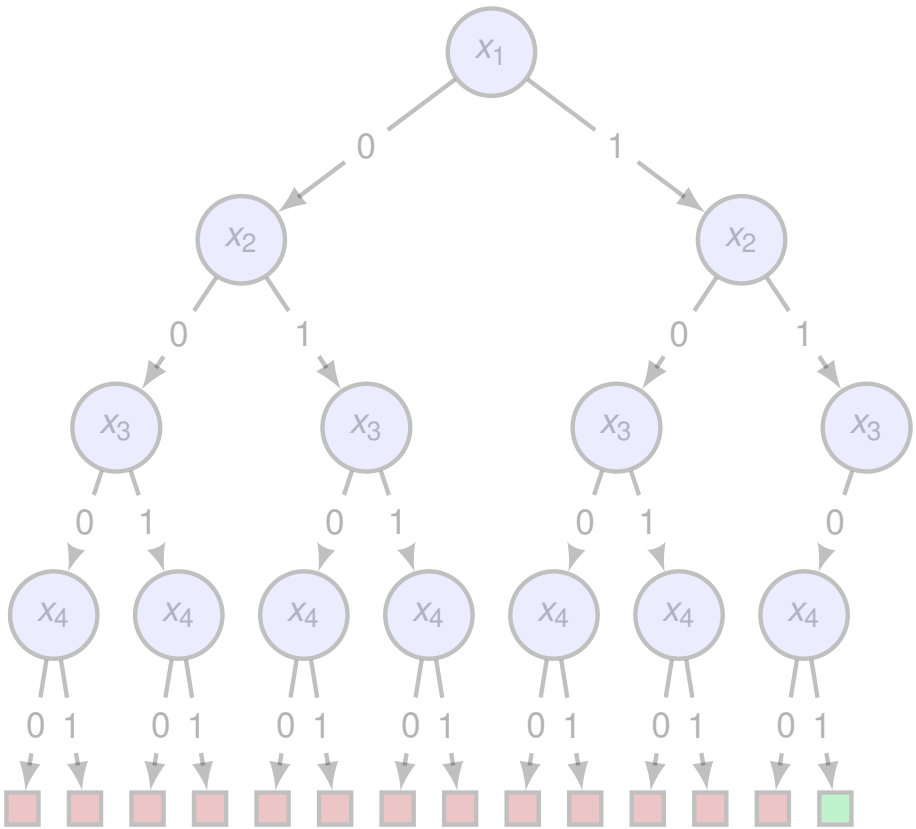
Solving SAT formula is known to be **NP-complete** [Coo71]

Enumerative algorithms:

- Davis, Putnam, Logemann, and Loveland (DPLL) [DLL62]
  - Boolean Constraint Propagation (BCP)

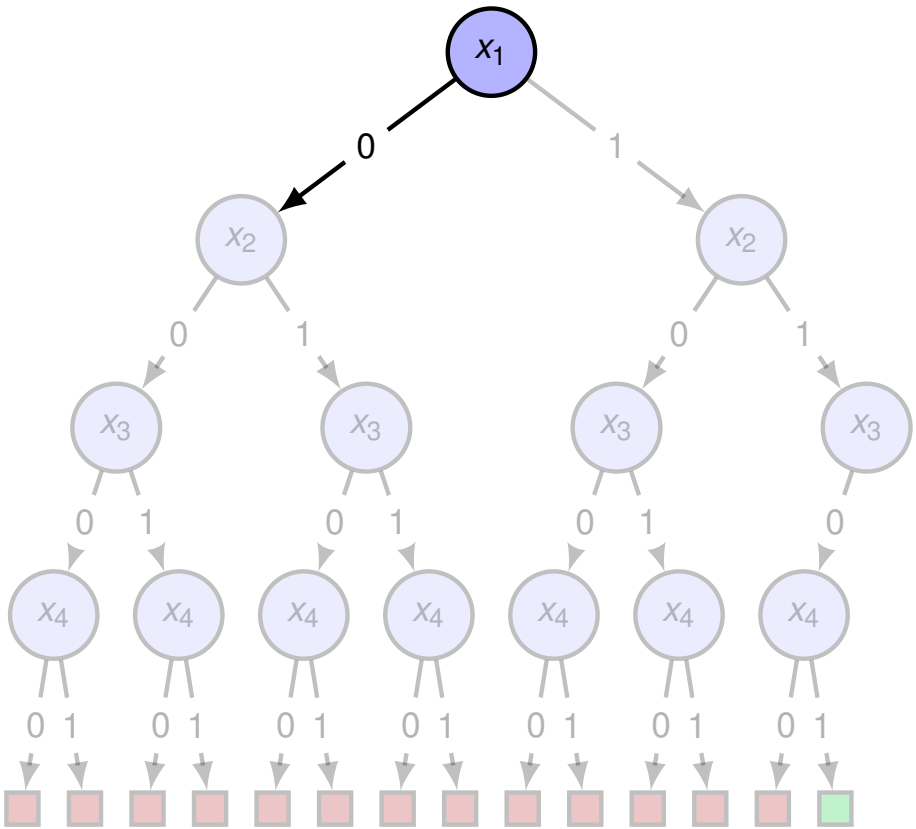
- Conflict Driven Clause Learning (CDCL) [MSS99]
  - Derived from DPLL
  - Clause learning

CDCL in action



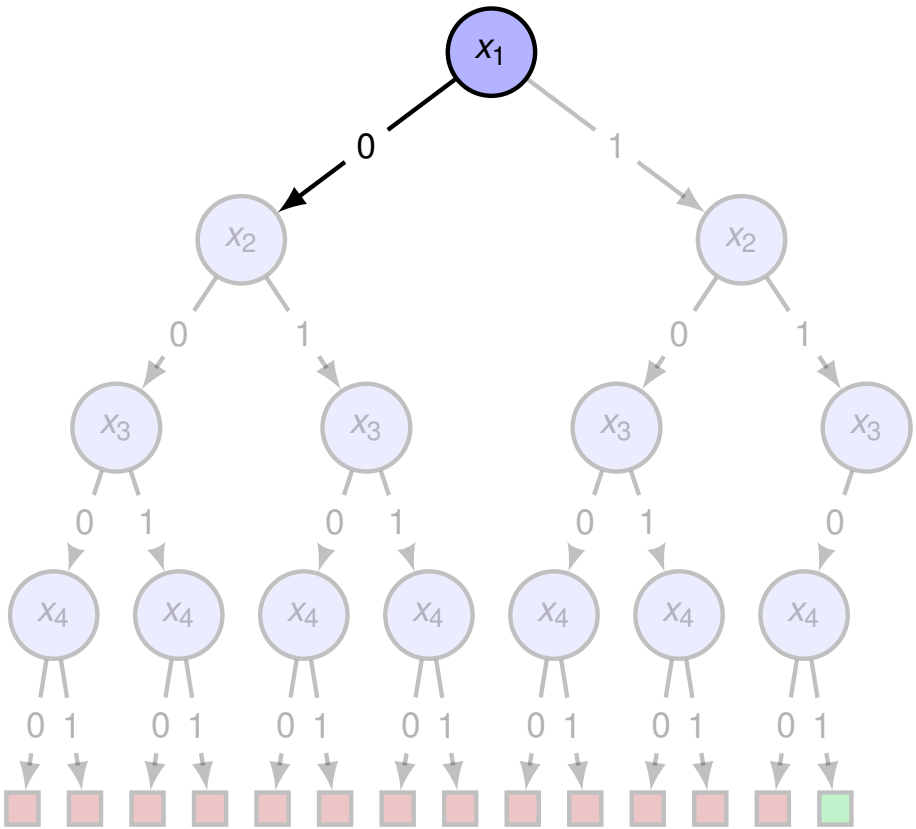
$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$

CDCL in action



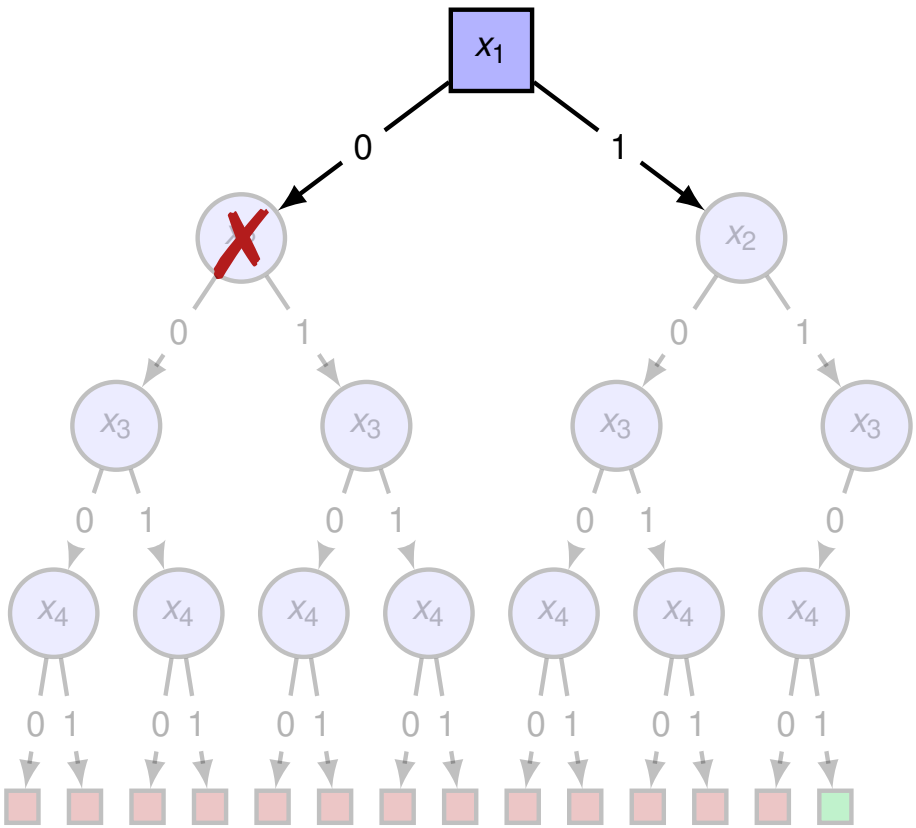
$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$

CDCL in action



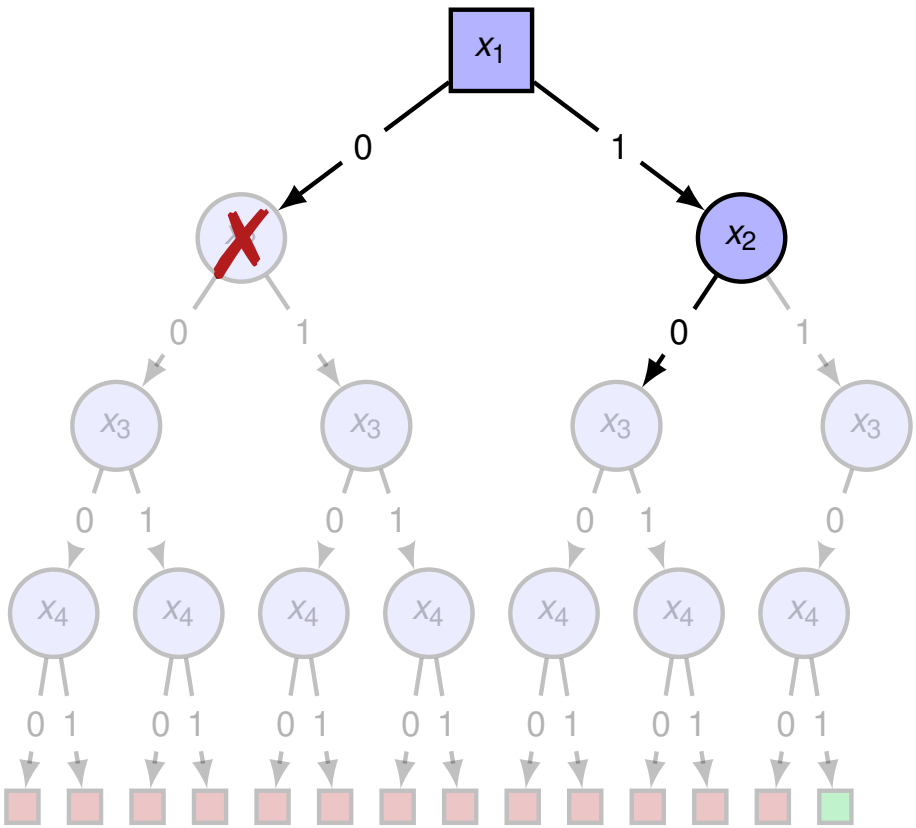
$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$

CDCL in action



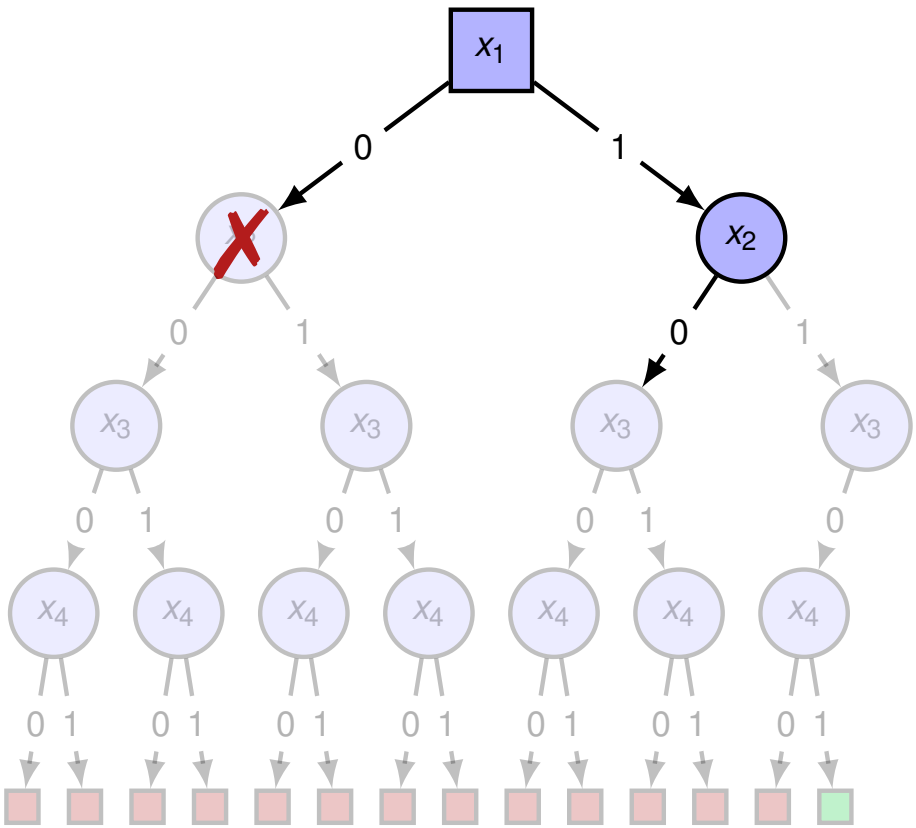
$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$
$$\omega_7 = \{x_1\}$$

CDCL in action



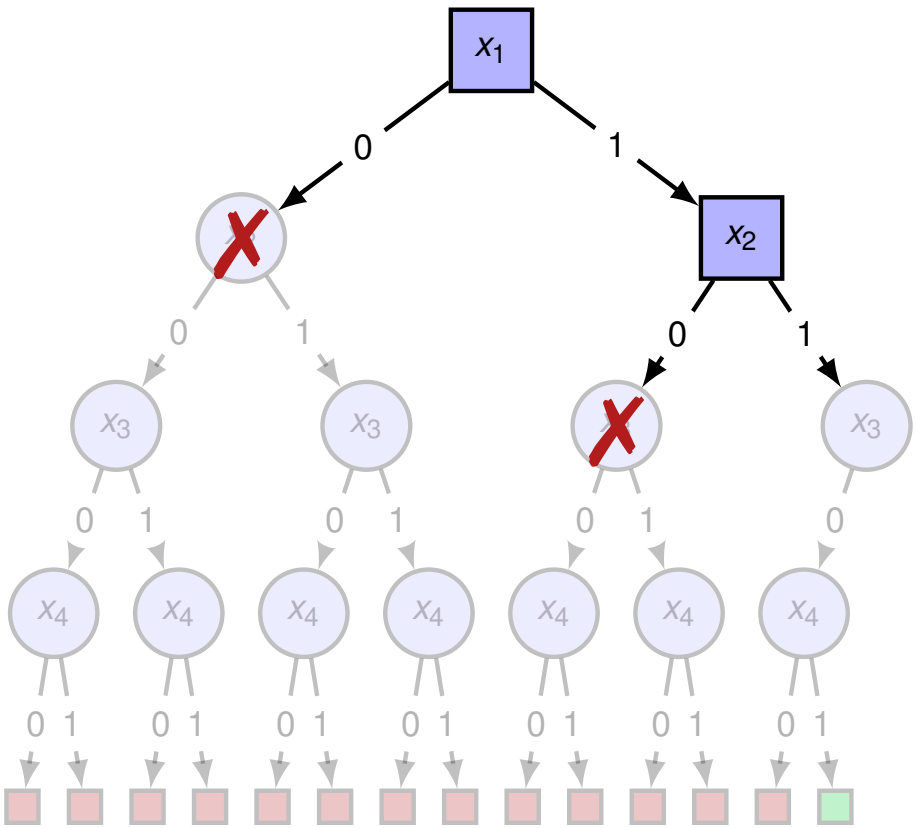
$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$
$$\omega_7 = \{x_1\}$$

CDCL in action



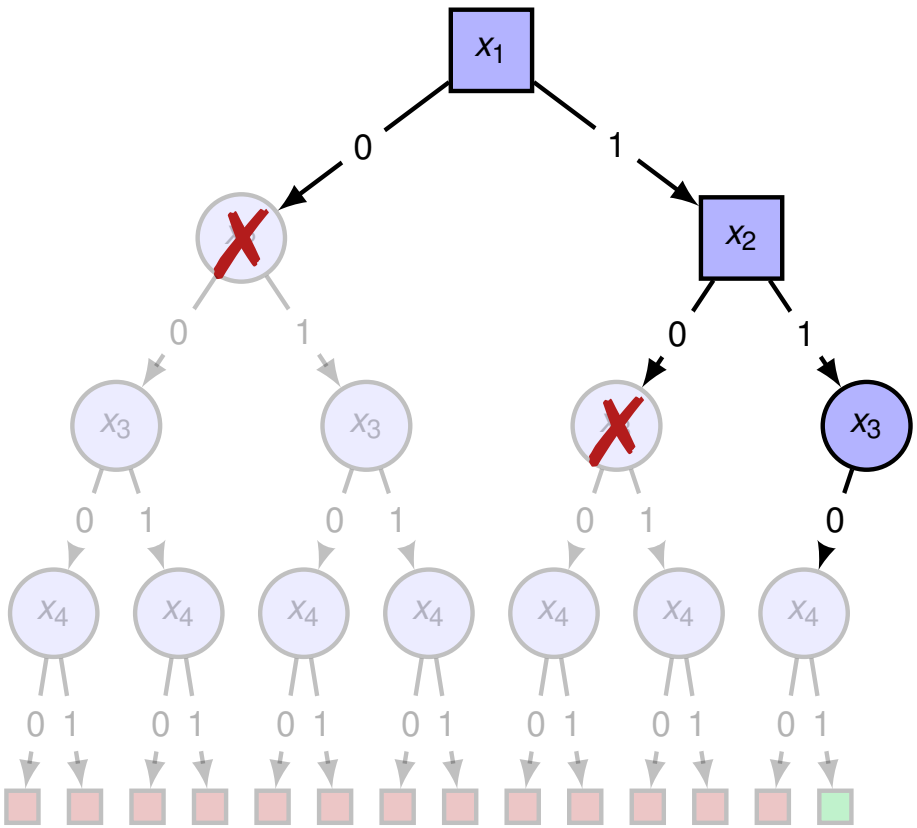
$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$
$$\omega_7 = \{x_1\}$$

CDCL in action



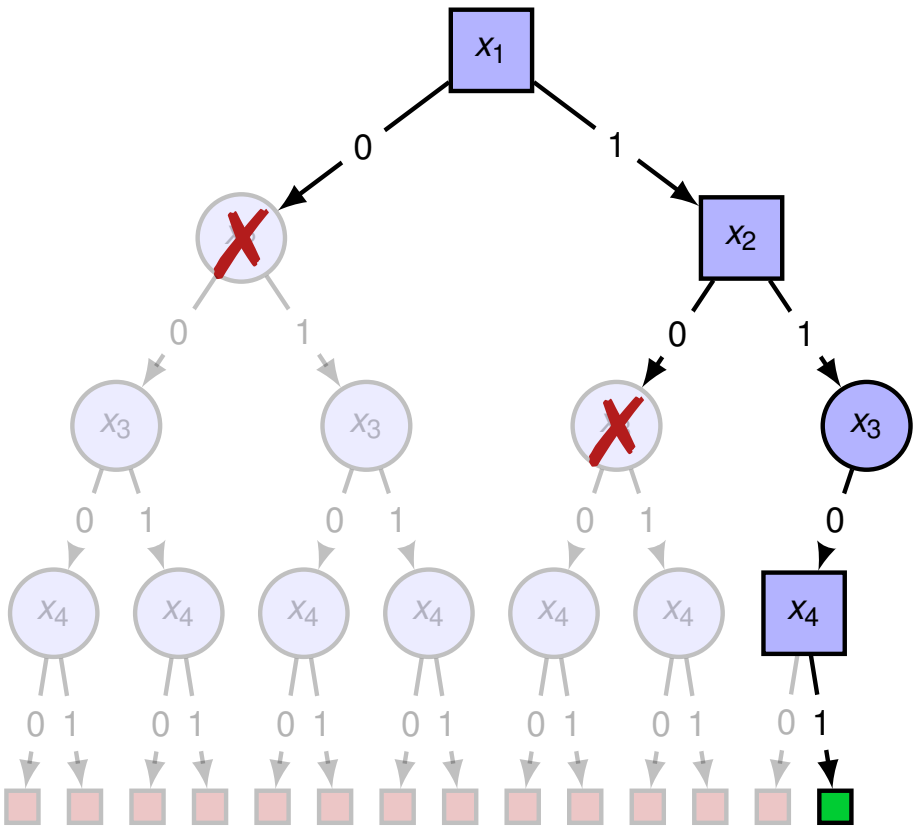
$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$
$$\omega_7 = \{x_1\}$$
$$\omega_8 = \{x_2\}$$

CDCL in action



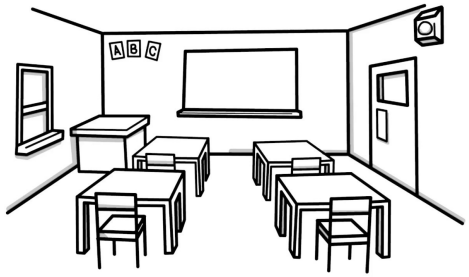
$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$
$$\omega_7 = \{x_1\}$$
$$\omega_8 = \{x_2\}$$

CDCL in action

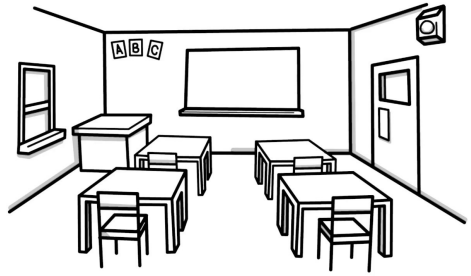


$$\omega_1 = \{x_1, x_2, x_3, x_4\}$$
$$\omega_2 = \{x_1, \neg x_4\}$$
$$\omega_3 = \{x_1, x_4\}$$
$$\omega_4 = \{x_2, \neg x_4\}$$
$$\omega_5 = \{x_2, x_4\}$$
$$\omega_6 = \{x_3, x_4\}$$
$$\omega_7 = \{x_1\}$$
$$\omega_8 = \{x_2\}$$

An UNSAT example

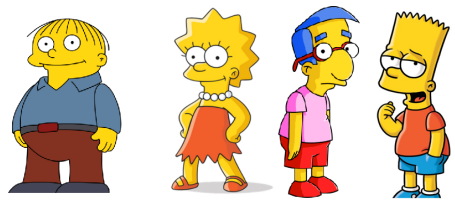


1

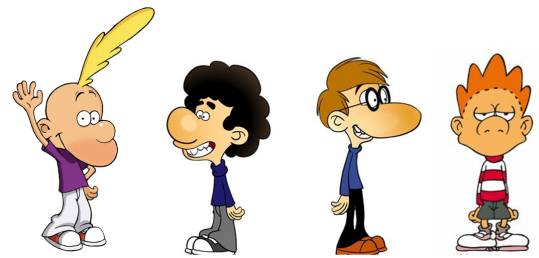


2

A



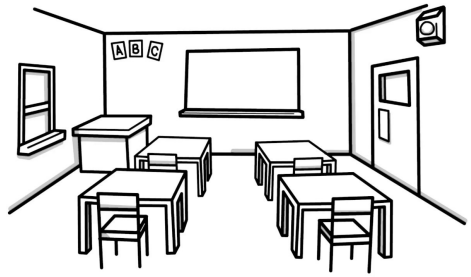
B

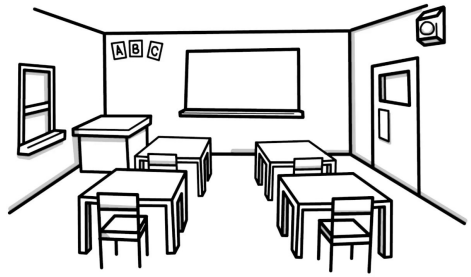


C

Is it possible to attribute each group to a classroom?

An UNSAT example

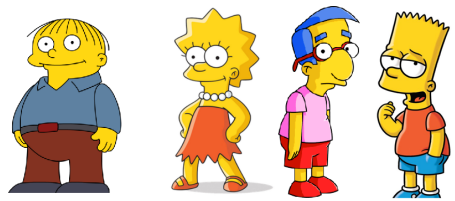


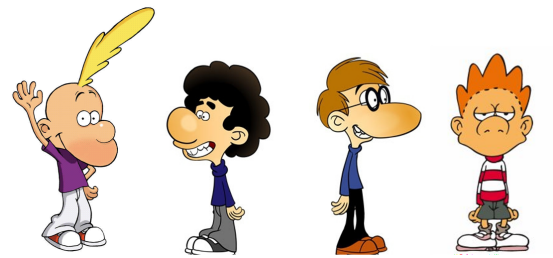


1

2

?





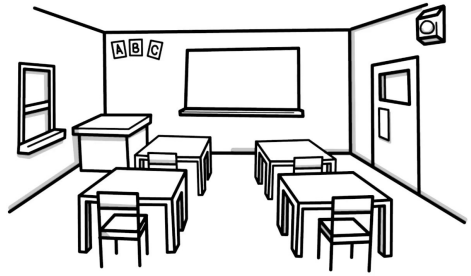
A

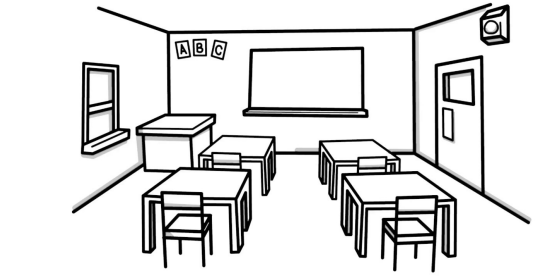
B

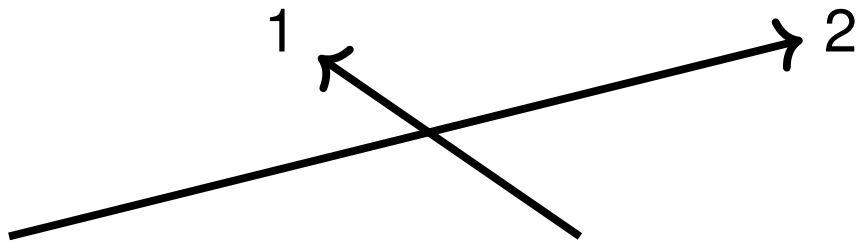
C

Is it possible to attribute each group to a classroom?

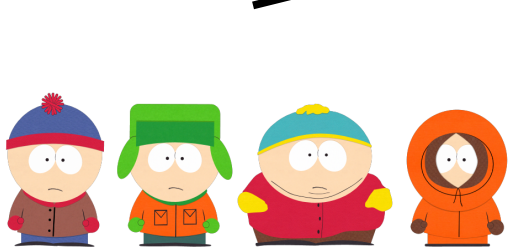
An UNSAT example

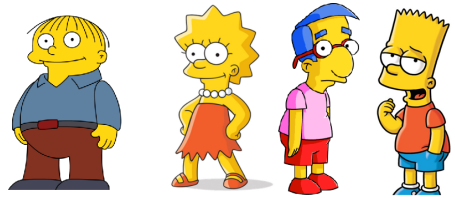


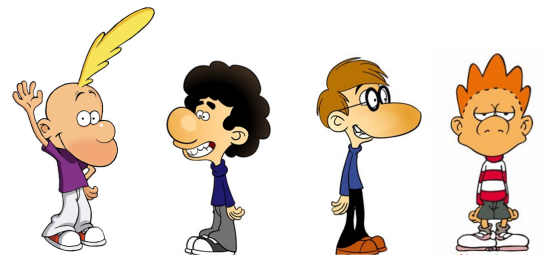




?



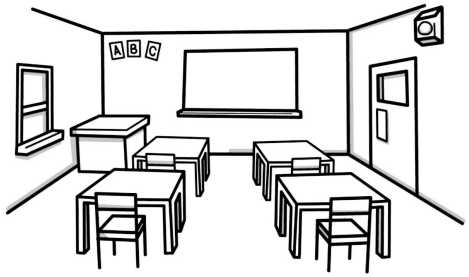




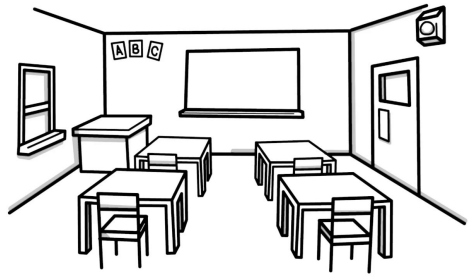
A B C

Is it possible to attribute each group to a classroom?

An UNSAT example



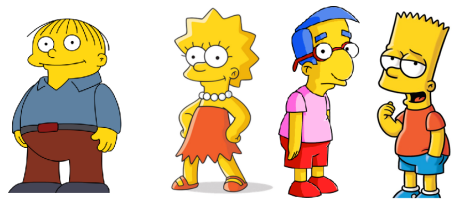
1



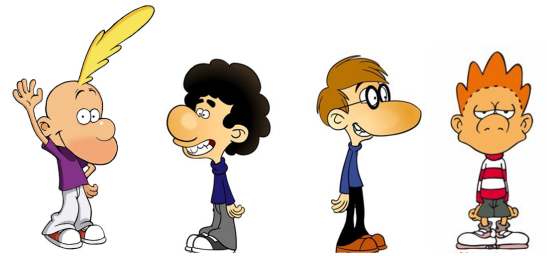
2

...

A



B

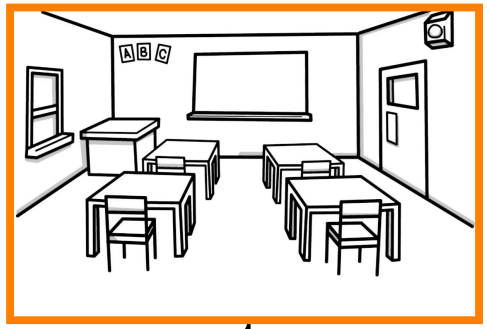


C

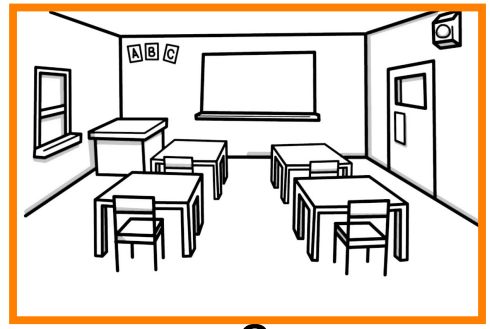
Is it possible to attribute each group to a classroom?

No!

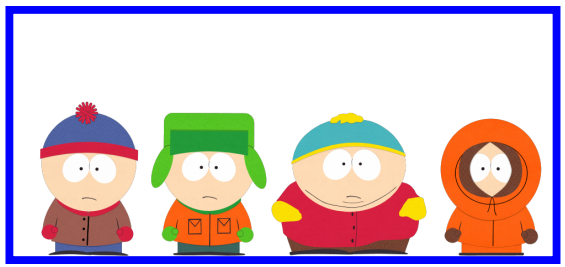
An UNSAT example



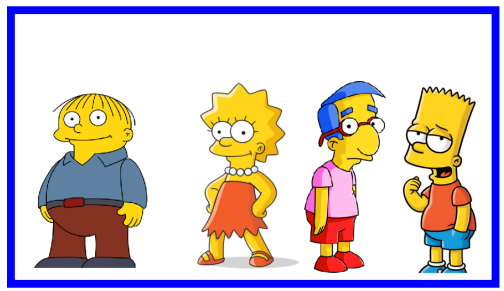
1



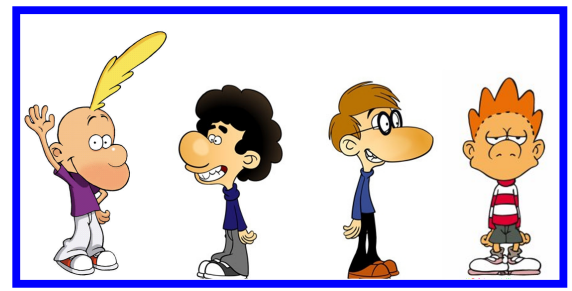
2



A



B



C

Is it possible to attribute each group to a classroom?

No!

Presence of symmetries hinders the performance of the solver

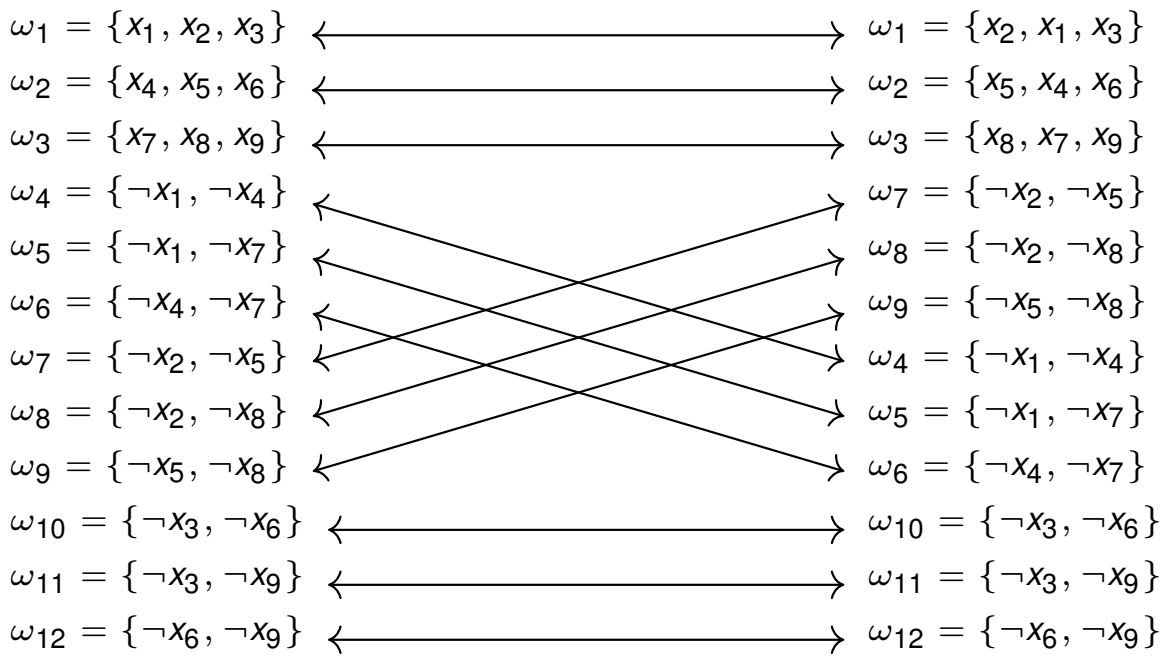
Outline

1. SAT overview
   - SAT basics
   - SAT and symmetries
2. Existing approaches
   - Static symmetry breaking
   - Dynamic symmetry breaking
3. Contribution and results

Symmetry

A symmetry (permuation) g is a bijective function (on variables) that leaves the formula invariant

$$g = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 \\ x_2 & x_1 & x_3 & x_5 & x_4 & x_6 & x_8 & x_7 & x_9 \end{pmatrix} \rightarrow (x_1\ x_2)(x_4\ x_5)(x_7\ x_8)$$



The set of symmetries of a formula is a group noted G

Computing symmetries of a SAT problem

CNF formula

$$(x_1 \vee x_2 \vee x_3) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_7 \vee x_8 \vee x_9)$$
$$\wedge(\neg x_1 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_7) \wedge (\neg x_4 \vee \neg x_7)$$
$$\wedge(\neg x_2 \vee \neg x_5) \wedge (\neg x_2 \vee \neg x_8) \wedge (\neg x_5 \vee \neg x_8)$$
$$\wedge(\neg x_3 \vee \neg x_6) \wedge (\neg x_3 \vee \neg x_9) \wedge (\neg x_6 \vee \neg x_9)$$

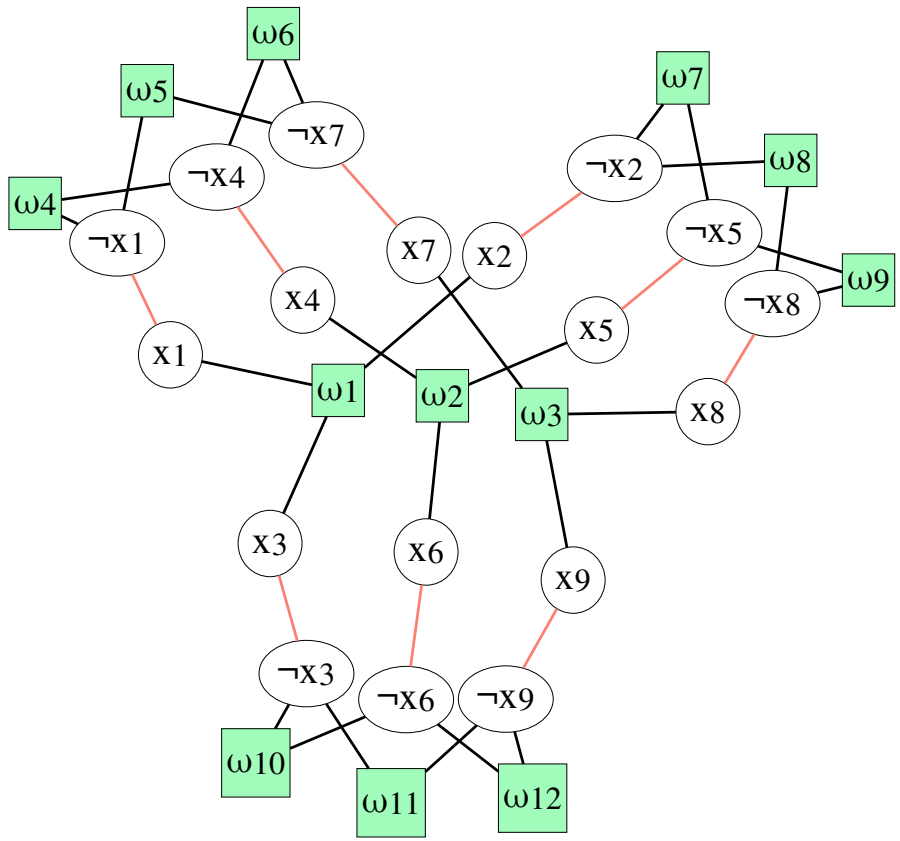
Computing symmetries of a SAT problem

CNF formula

$$(x_1 \vee x_2 \vee x_3) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_7 \vee x_8 \vee x_9)$$
$$\wedge(\neg x_1 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_7) \wedge (\neg x_4 \vee \neg x_7)$$
$$\wedge(\neg x_2 \vee \neg x_5) \wedge (\neg x_2 \vee \neg x_8) \wedge (\neg x_5 \vee \neg x_8)$$
$$\wedge(\neg x_3 \vee \neg x_6) \wedge (\neg x_3 \vee \neg x_9) \wedge (\neg x_6 \vee \neg x_9)$$

$\Downarrow$ $\Downarrow$

colored graph



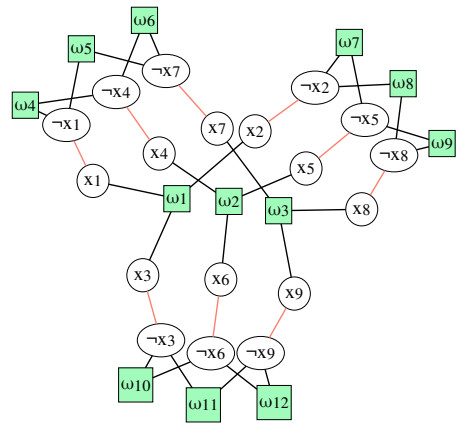
Computing symmetries of a SAT problem

CNF formula

$$(x_1 \vee x_2 \vee x_3) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_7 \vee x_8 \vee x_9)$$
$$\wedge(\neg x_1 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_7) \wedge (\neg x_4 \vee \neg x_7)$$
$$\wedge(\neg x_2 \vee \neg x_5) \wedge (\neg x_2 \vee \neg x_8) \wedge (\neg x_5 \vee \neg x_8)$$
$$\wedge(\neg x_3 \vee \neg x_6) \wedge (\neg x_3 \vee \neg x_9) \wedge (\neg x_6 \vee \neg x_9)$$

$\Downarrow$ $\qquad$ $\Downarrow$

colored graph



$\|$ $\qquad$ $\|$

graph automorphism $\qquad$ (`bliss`, `saucy`, $\cdots$)

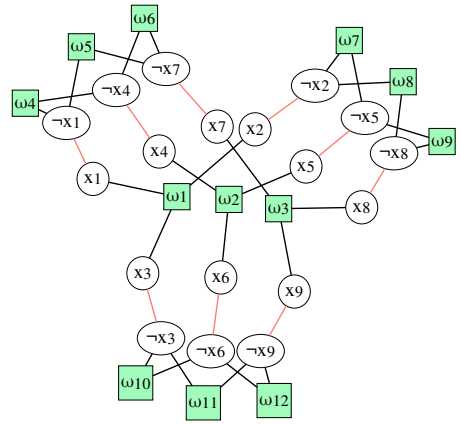
Computing symmetries of a SAT problem

CNF formula

$$(x_1 \vee x_2 \vee x_3) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_7 \vee x_8 \vee x_9)$$
$$\wedge(\neg x_1 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_7) \wedge (\neg x_4 \vee \neg x_7)$$
$$\wedge(\neg x_2 \vee \neg x_5) \wedge (\neg x_2 \vee \neg x_8) \wedge (\neg x_5 \vee \neg x_8)$$
$$\wedge(\neg x_3 \vee \neg x_6) \wedge (\neg x_3 \vee \neg x_9) \wedge (\neg x_6 \vee \neg x_9)$$

$\Downarrow$ $\qquad$ $\Downarrow$

colored graph



$\Vert$ $\qquad$ $\Vert$

graph automorphism $\qquad$ (`bliss`, `saucy`, $\cdots$)

$\Downarrow$ $\qquad$ $\Downarrow$

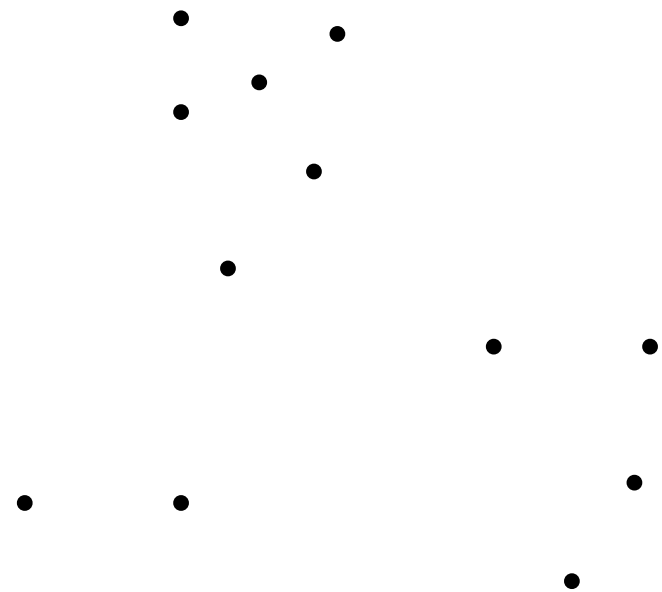
set of symmetries

$$g_1 = (x_2 \;\; x_3)(x_5 \;\; x_6)(x_8 \;\; x_9)$$
$$g_2 = (x_4 \;\; x_7)(x_5 \;\; x_8)(x_6 \;\; x_9)$$
$$g_3 = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$$
$$g_4 = (x_1 \;\; x_4)(x_2 \;\; x_5)(x_3 \;\; x_6)$$

Orbit

Orbit of an assignment $\alpha = G.\alpha = \{g.\alpha \mid g \in G\}$

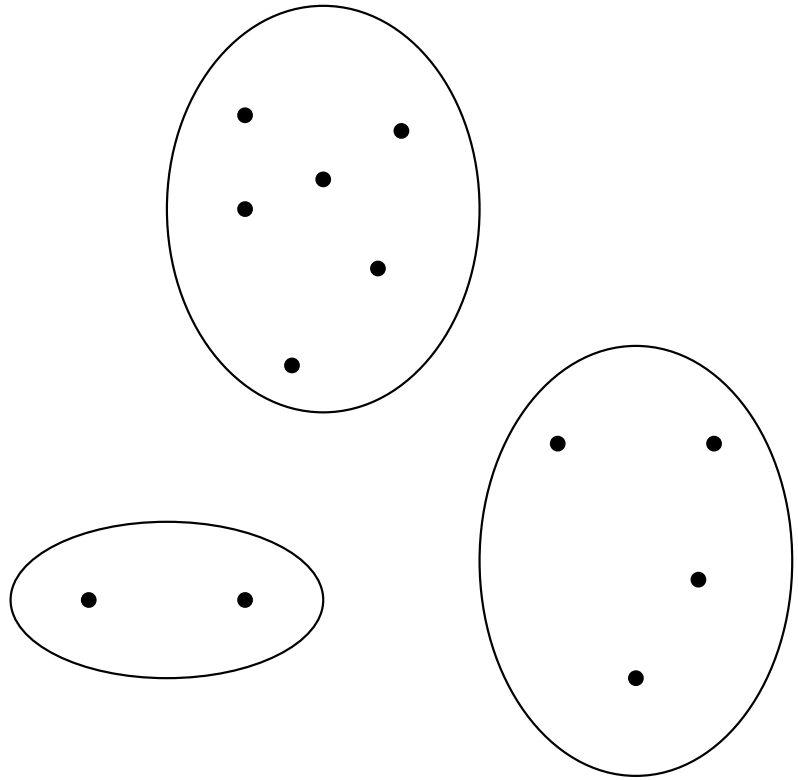
- full assignment



Orbit

Orbit of an assignment $\alpha = G.\alpha = \{g.\alpha \mid g \in G\}$

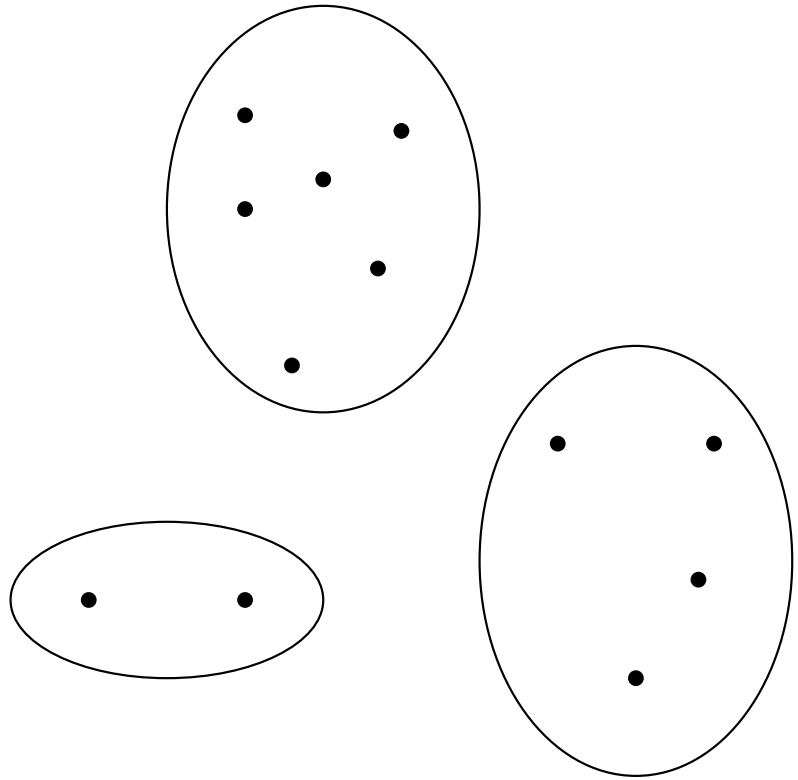
- full assignment
- orbit



Orbit

Orbit of an assignment $\alpha = G.\alpha = \{g.\alpha \mid g \in G\}$

- full assignment
- orbit

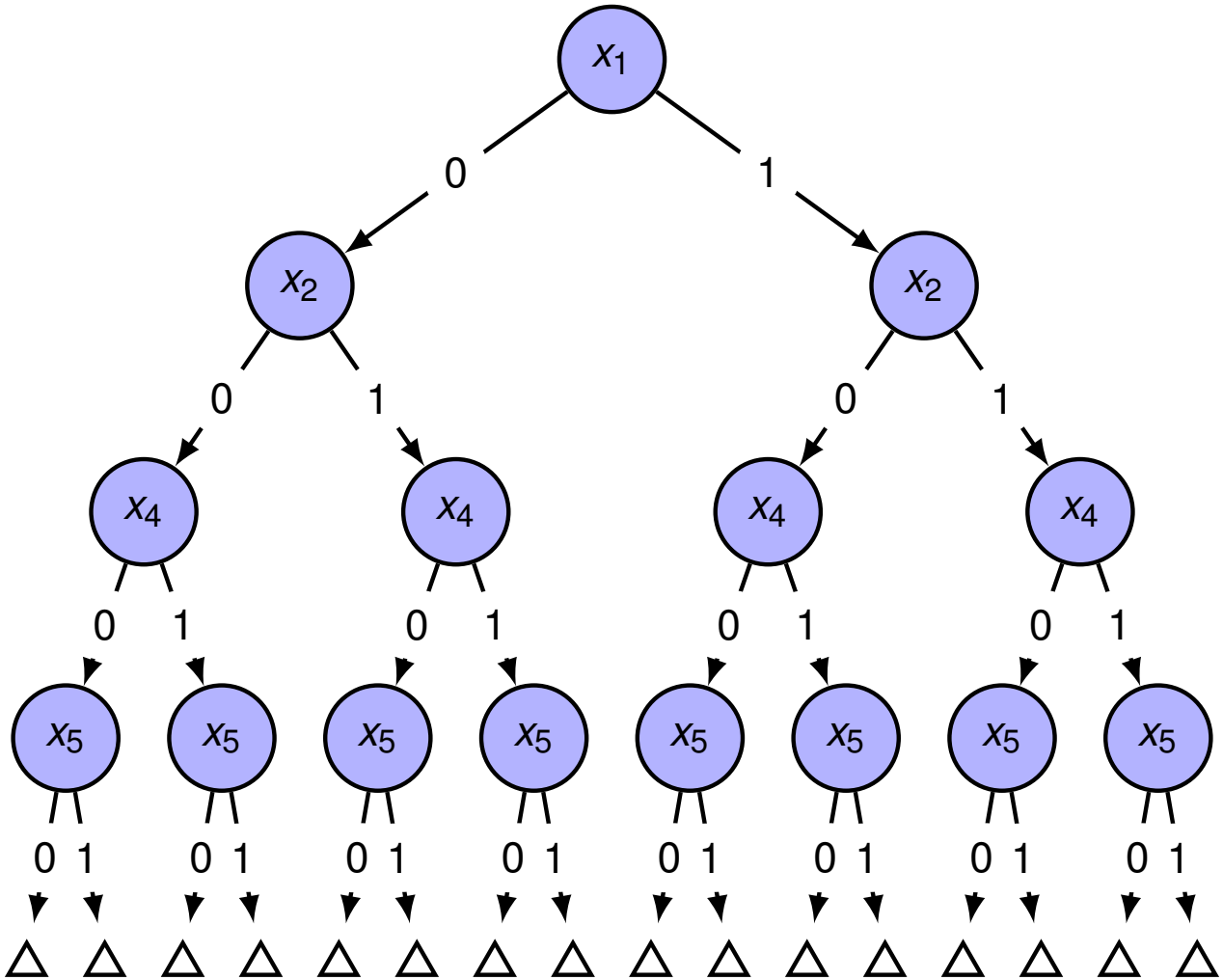


All or nothing property:

- Either $G.\alpha$ contains no solution
- Or all elements of $G.\alpha$ are solutions

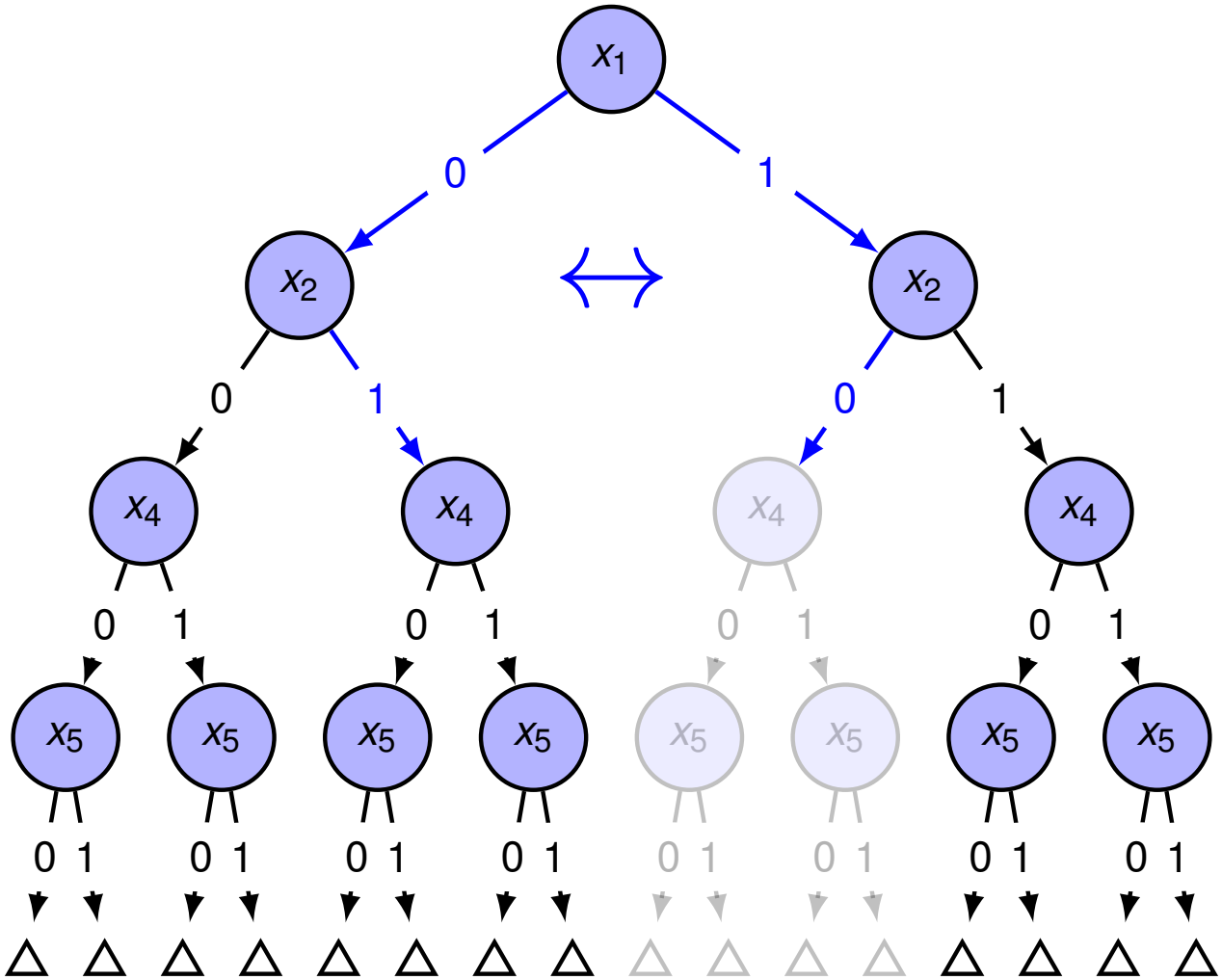
Using symmetries to prune the search space

$g = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$



Using symmetries to prune the search space

$g = (x_1\ \ x_2)(x_4\ \ x_5)(x_7\ \ x_8)$



Add symmetry breaking predicates to prune the search space.

Generates symmetry breaking predicates (SBP)

- Define lexicographic order
  - Define total order on variables
  - Define minimal value
- Forbid non minimal assignment for each orbit

Example:

$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8; \mathtt{F} < \mathtt{T}$

$g = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$

	x_1	x_2	x_3	x_4	x_5	$\cdots$	lex-leader	SBP

Generates symmetry breaking predicates (SBP)

- Define lexicographic order
  - Define total order on variables
  - Define minimal value
- Forbid non minimal assignment for each orbit

Example:

$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8; \mathtt{F} < \mathtt{T}$

$g = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$

	x_1	x_2	x_3	x_4	x_5	$\cdots$	lex-leader	SBP
O_1	F	T	–	–	–	$\cdots$	✓	

Generates symmetry breaking predicates (SBP)

- Define lexicographic order
  - Define total order on variables
  - Define minimal value
- Forbid non minimal assignment for each orbit

Example:

$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8; \mathtt{F} < \mathtt{T}$

$g = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$

	x_1	x_2	x_3	x_4	x_5	$\cdots$	lex-leader	SBP
O_1	F	T	–	–	–	$\cdots$	✓	
	T	F	–	–	–	$\cdots$	✗	$\rightarrow \neg x_1 \vee x_2$

Generates symmetry breaking predicates (SBP)

- Define lexicographic order
  - Define total order on variables
  - Define minimal value
- Forbid non minimal assignment for each orbit

Example:

$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8; \mathtt{F} < \mathtt{T}$

$g = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$

	x_1	x_2	x_3	x_4	x_5	$\cdots$	lex-leader	SBP
O_1	F	T	–	–	–	$\cdots$	✓	
	T	F	–	–	–	$\cdots$	✗	$\rightarrow \neg x_1 \vee x_2$
O_2	F	F	–	F	T	$\cdots$	✓	

Generates symmetry breaking predicates (SBP)

- Define lexicographic order
  - Define total order on variables
  - Define minimal value
- Forbid non minimal assignment for each orbit

Example:

$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8; \mathtt{F} < \mathtt{T}$

$g = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$

	x_1	x_2	x_3	x_4	x_5	$\cdots$	lex-leader	SBP
O_1	F	T	–	–	–	$\cdots$	✓	
	T	F	–	–	–	$\cdots$	✗	$\rightarrow \neg x_1 \vee x_2$
O_2	F	F	–	F	T	$\cdots$	✓	
	F	F	–	T	F	$\cdots$	✗	$\rightarrow x_1 \vee x_2 \vee \neg x_4 \vee x_5$

Generates symmetry breaking predicates (SBP)

- Define lexicographic order
  - Define total order on variables
  - Define minimal value
- Forbid non minimal assignment for each orbit

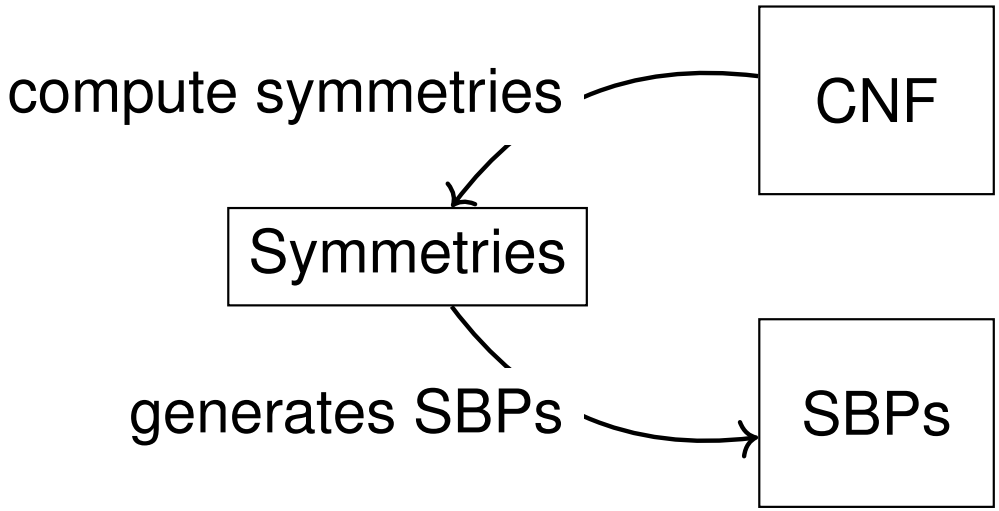
Example:

$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8; \mathtt{F} < \mathtt{T}$

$g = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$

	x_1	x_2	x_3	x_4	x_5	$\cdots$	lex-leader	SBP
O_1	F	T	–	–	–	$\cdots$	✓	
	T	F	–	–	–	$\cdots$	✗	$\rightarrow \neg x_1 \vee x_2$
O_2	F	F	–	F	T	$\cdots$	✓	
	F	F	–	T	F	$\cdots$	✗	$\rightarrow x_1 \vee x_2 \vee \neg x_4 \vee x_5$
						$\cdots$		

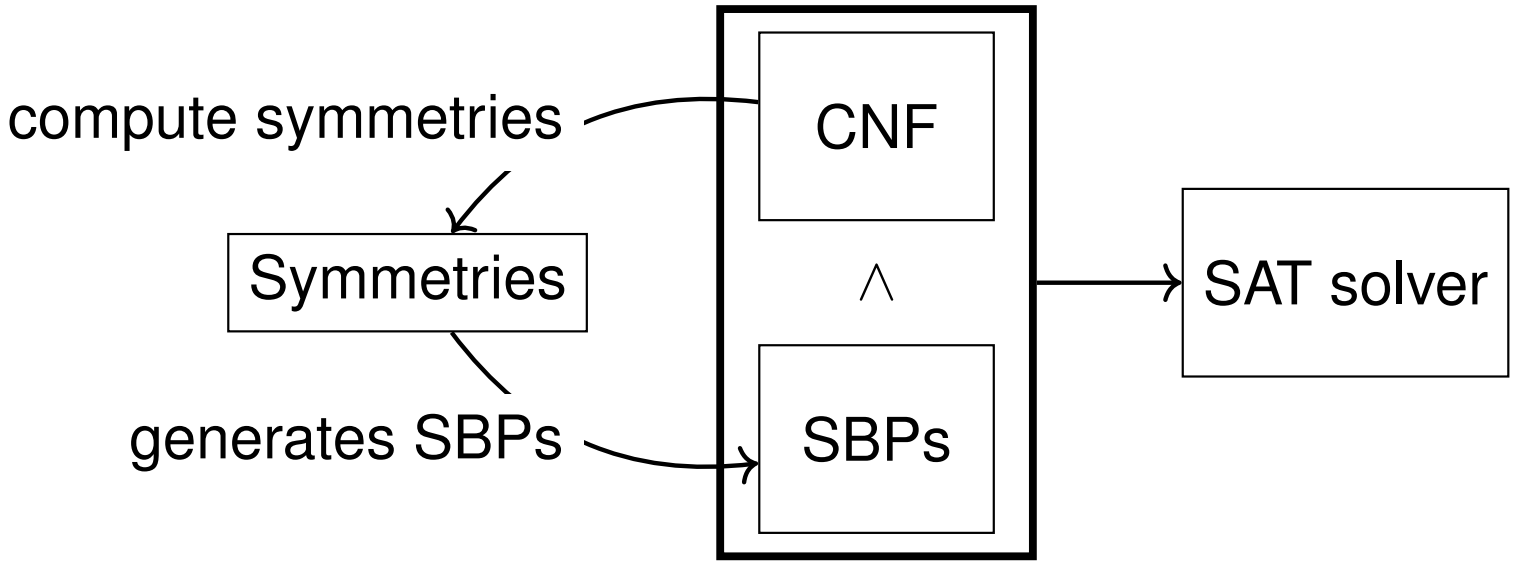
Static symmetry breaking



Different approaches:

- Shatter [ASM06]
- BreakID [DBBD16]
- ...

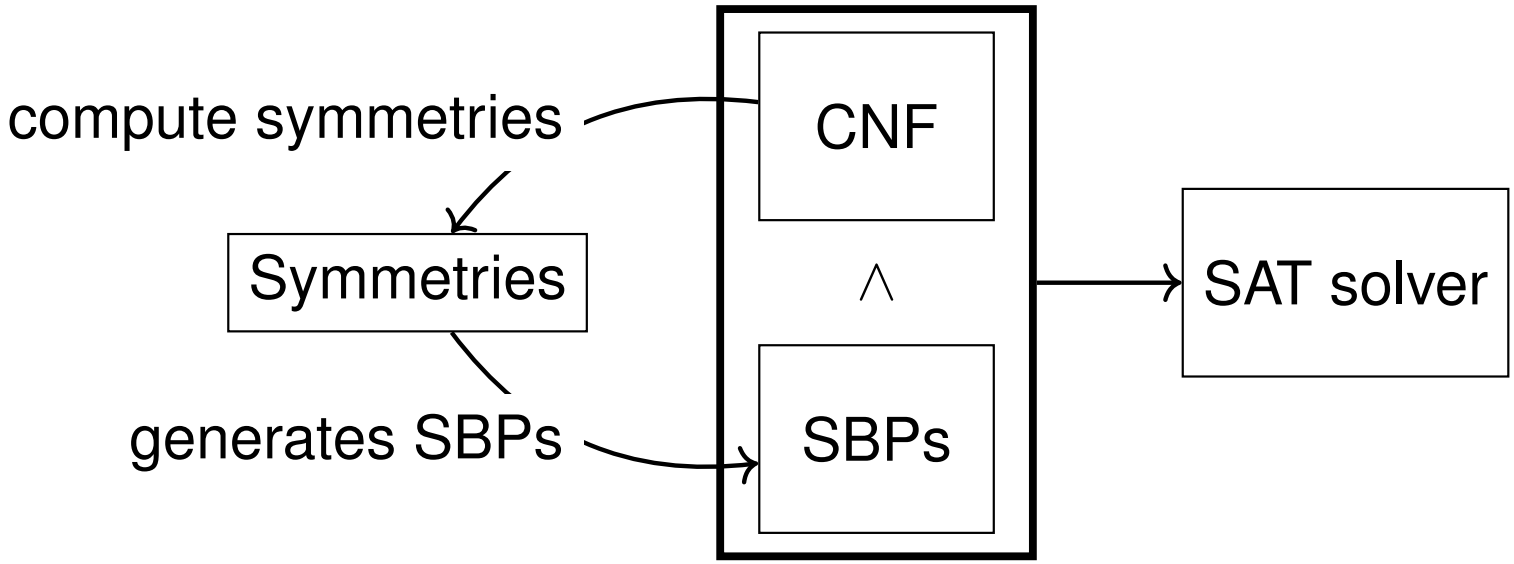
Static symmetry breaking



Different approaches:

- Shatter [ASM06]
- BreakID [DBBD16]
- ...

Static symmetry breaking



Different approaches:

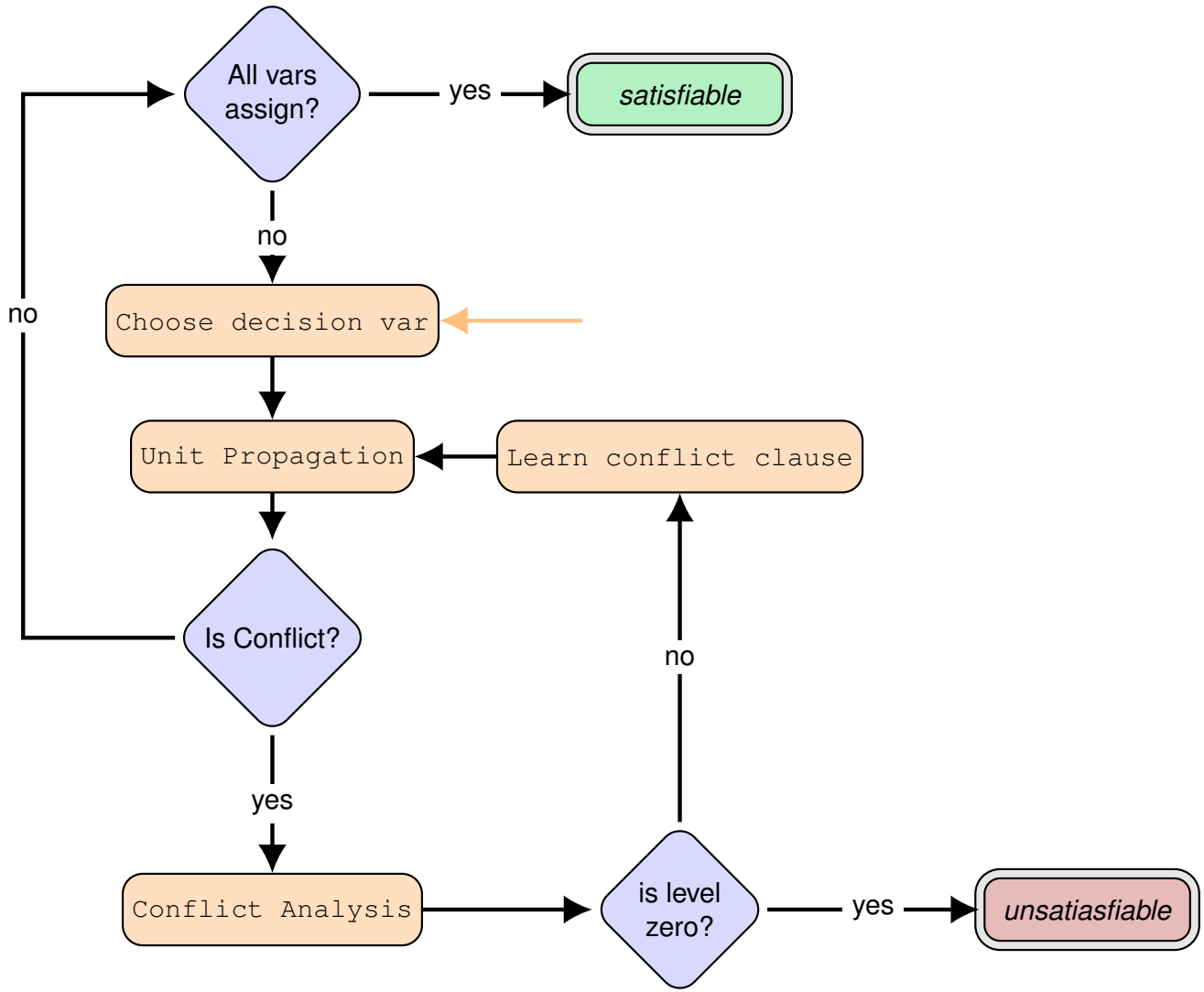
- Shatter [ASM06]
- BreakID [DBBD16]
- ...

Pros/Cons:

- Works well on many symmetric instances
- The solver can ”explode” instead of being helped

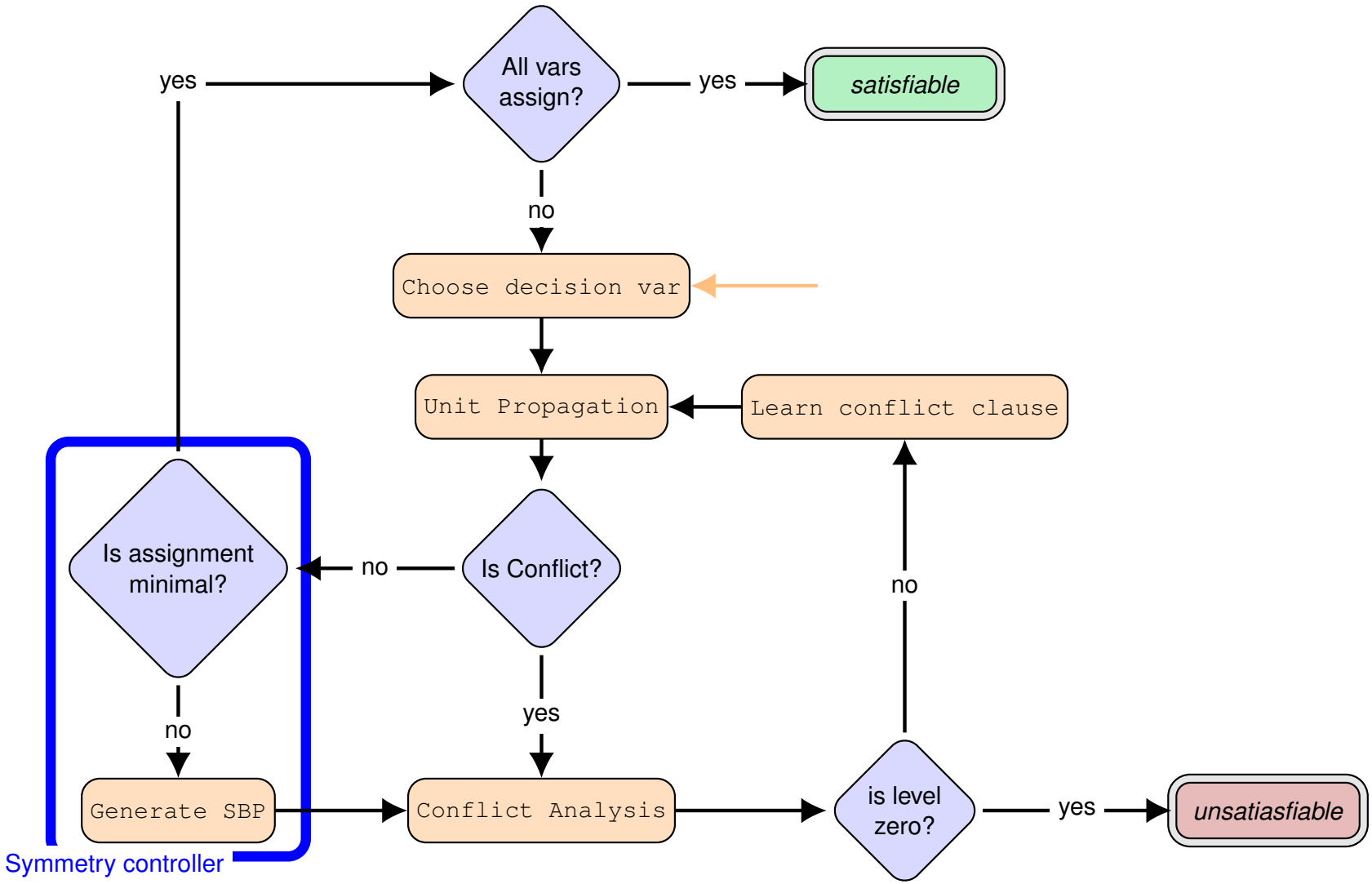
Our contribution CDCL[Sym]

Compute and inject SBP opportunistically, during the solving



Our contribution CDCL[Sym]

Compute and inject SBP opportunistically, during the solving



Symmetry status

- `reducer`: $g.\alpha \prec \alpha$
- `inactive` : $\alpha \prec g.\alpha$
- `active` : *not enough information*

Efficient implementation of symmetry status

Keep track the smallest unassigned variable x :

1. $\alpha(g.x) \leq \alpha(x)$, then g is `reducer` $\Rightarrow$ Effective SBP (ESBP)
2. $\alpha(x) \leq \alpha(g.x)$, then g is `inactive` $\Rightarrow$ g cannot reduce α
3. $\alpha(g.x)$ or $\alpha(x)$ is unassigned then g is `active`

Update whenever variables are assigned / unassigned

Example

1. `reducer`: $\alpha(g.x) \leq \alpha(x)$
2. `inactive`: $\alpha(x) \leq \alpha(g.x)$
3. `active`: $\alpha(g.x)$ or $\alpha(x)$ is unassigned

$$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8 \;;\; \texttt{F} < \texttt{T}$$

$$g_1 = \begin{array}{cccccc} (x_2 & x_3) & (x_5 & x_6) & (x_8 & x_9) \\ \uparrow & & & & & \end{array} \Bigg| \begin{array}{ll} x = x_2 & g.x = x_3 \\ & \texttt{active} \end{array}$$

$$g_2 = \begin{array}{cccccc} (x_1 & x_2) & (x_4 & x_5) & (x_7 & x_8) \\ \uparrow & & & & & \end{array} \Bigg| \begin{array}{ll} x = x_1 & g.x = x_2 \\ & \texttt{active} \end{array}$$

$$\ldots$$

$$\alpha = \{ \qquad \}$$

Example

1. `reducer`: $\alpha(g.x) \leq \alpha(x)$
2. `inactive`: $\alpha(x) \leq \alpha(g.x)$
3. `active`: $\alpha(g.x)$ or $\alpha(x)$ is unassigned

$$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8 \;\; ; \;\; \mathtt{F} < \mathtt{T}$$

$$g_1 = \underset{\uparrow}{(x_2 \quad x_3)} \quad (x_5 \quad x_6) \quad (x_8 \quad x_9) \;\Big|\; x = x_2 \quad g.x = x_3 \;\; \mathtt{active}$$

$$g_2 = \underset{\uparrow}{(x_1 \quad x_2)} \quad (x_4 \quad x_5) \quad (x_7 \quad x_8) \;\Big|\; x = x_1 \quad g.x = x_2 \;\; \mathtt{active}$$

$$\ldots$$

$$\alpha = \{\neg x_2 \qquad\qquad \}$$

Example

1. `reducer`: $\alpha(g.x) \leq \alpha(x)$
2. `inactive`: $\alpha(x) \leq \alpha(g.x)$
3. `active`: $\alpha(g.x)$ or $\alpha(x)$ is unassigned

$$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8 \;;\; \mathtt{F} < \mathtt{T}$$

$$g_1 = \begin{array}{cccccccc} & (x_2 & x_3) & & (x_5 & x_6) & (x_8 & x_9) \\ & & & & \uparrow & & & \end{array} \quad \Big| \quad x = x_5 \quad g.x = x_6 \;\; \mathtt{active}$$

$$g_2 = \begin{array}{cccccccc} & (x_1 & x_2) & & (x_4 & x_5) & (x_7 & x_8) \\ & \uparrow & & & & & & \\ & & & \ldots & & & & \end{array} \quad \Big| \quad x = x_1 \quad g.x = x_2 \;\; \mathtt{reducer}$$

$$\alpha = \{\neg x_2, \neg x_3, x_1\}$$

Example

1. `reducer`: $\alpha(g.x) \leq \alpha(x)$
2. `inactive`: $\alpha(x) \leq \alpha(g.x)$
3. `active`: $\alpha(g.x)$ or $\alpha(x)$ is unassigned

$$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq x_6 \leq x_7 \leq x_8 \;;\; \mathtt{F} < \mathtt{T}$$

$$g_1 = \begin{array}{cccccccc} (x_2 & x_3) & (x_5 & x_6) & (x_8 & x_9) \\ & & \uparrow & & & \end{array} \Bigg| \begin{array}{ll} x = x_5 & g.x = x_6 \\ & \mathtt{active} \end{array}$$

$$g_2 = \begin{array}{cccccccc} (x_1 & x_2) & (x_4 & x_5) & (x_7 & x_8) \\ \uparrow & & \ldots & & & \end{array} \Bigg| \begin{array}{ll} x = x_1 & g.x = x_2 \\ & \mathtt{reducer} \end{array}$$

$$\alpha = \{\neg x_2, \neg x_3, x_1\}$$

g_2 generates $\omega = \{\neg x_1, x_2\}$

CDCL[Sym] Implementation

- Packaged as a library **cosy**[1], to be combined with your solver

  → e.g. +3% LOC on `MiniSAT`.
- Follows symmetry status
- Should work with any enumerative SAT solver

[1] `https://github.com/lip6/cosy`

Experiments

Benchmark:

- from SAT contests 2012 – 2017
- retain only instances for which `bliss` finds significant symmetries in 1000s
- 1350 symmetric instances (out of 3700)

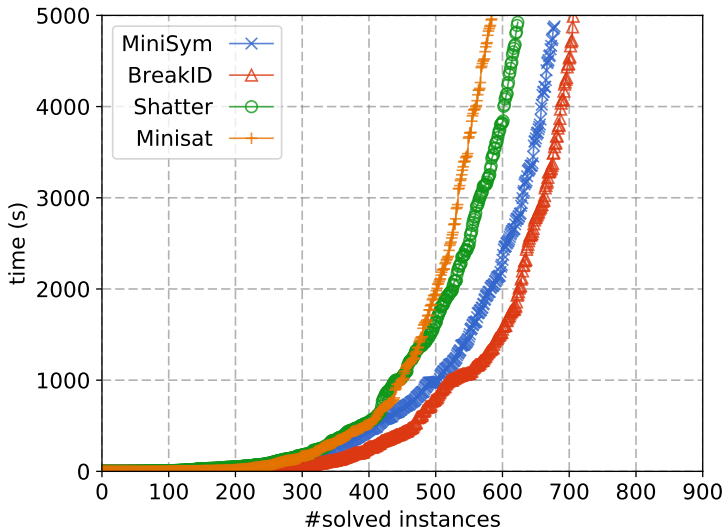
Setup:

- four tools
  - MiniSat (no symmetry, baseline)
  - MiniSat + BreakID (SOTA SAT solver using symmetries)
  - MiniSat + Shatter (SOTA SAT solver using symmetries)
  - **MiniSym** = MiniSat + CDCL[Sym] (our approach)
- 5000s timeout, 8GB memory
- includes time to compute symmetries (except for MiniSat)

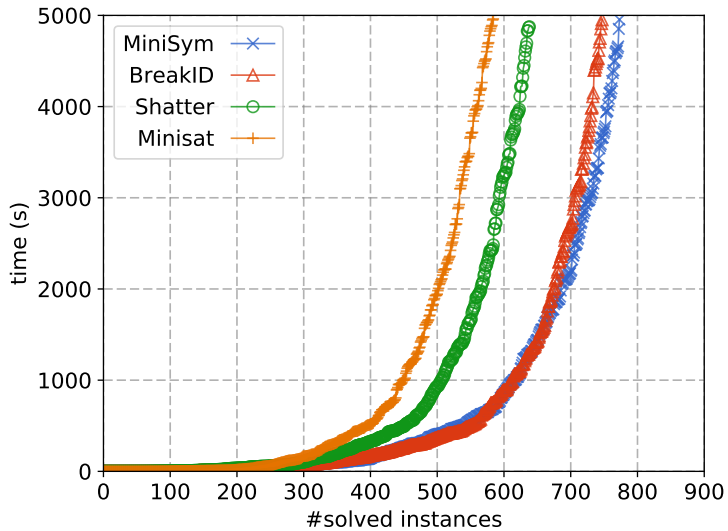
Experimental results

`bliss` gives more generators than `saucy3`

Figure: Cactus plot total number of instances



(a) with `saucy3`



(b) with `bliss`

	MiniSAT	Shatter	BreakID	MiniSym
PAR-2 sum	8 074 348	7 770 434	**6 909 999**	7 229 700
PAR-2 avg	5 981	5 756	**5 119**	5 355

	MiniSAT	Shatter	BreakID	MiniSym
PAR-2 sum	8 074 348	7 517 556	6 444 954	**6 245 448**
PAR-2 avg	5 981	5 569	4 774	**4 626**

Table: Time comparison

Experimental results (UNSAT versus SAT)

	MiniSAT	Shatter	BreakID	MiniSym
TOTAL	261	302	**371**	345

(a) With `saucy3`

	MiniSAT	Shatter	BreakID	MiniSym
TOTAL	261	324	415	**439**

(b) With `bliss`

Table: Comparison on UNSAT instances

	MiniSAT	Shatter	BreakID	MiniSym
TOTAL	325	323	**337**	335

(a) With `saucy3`

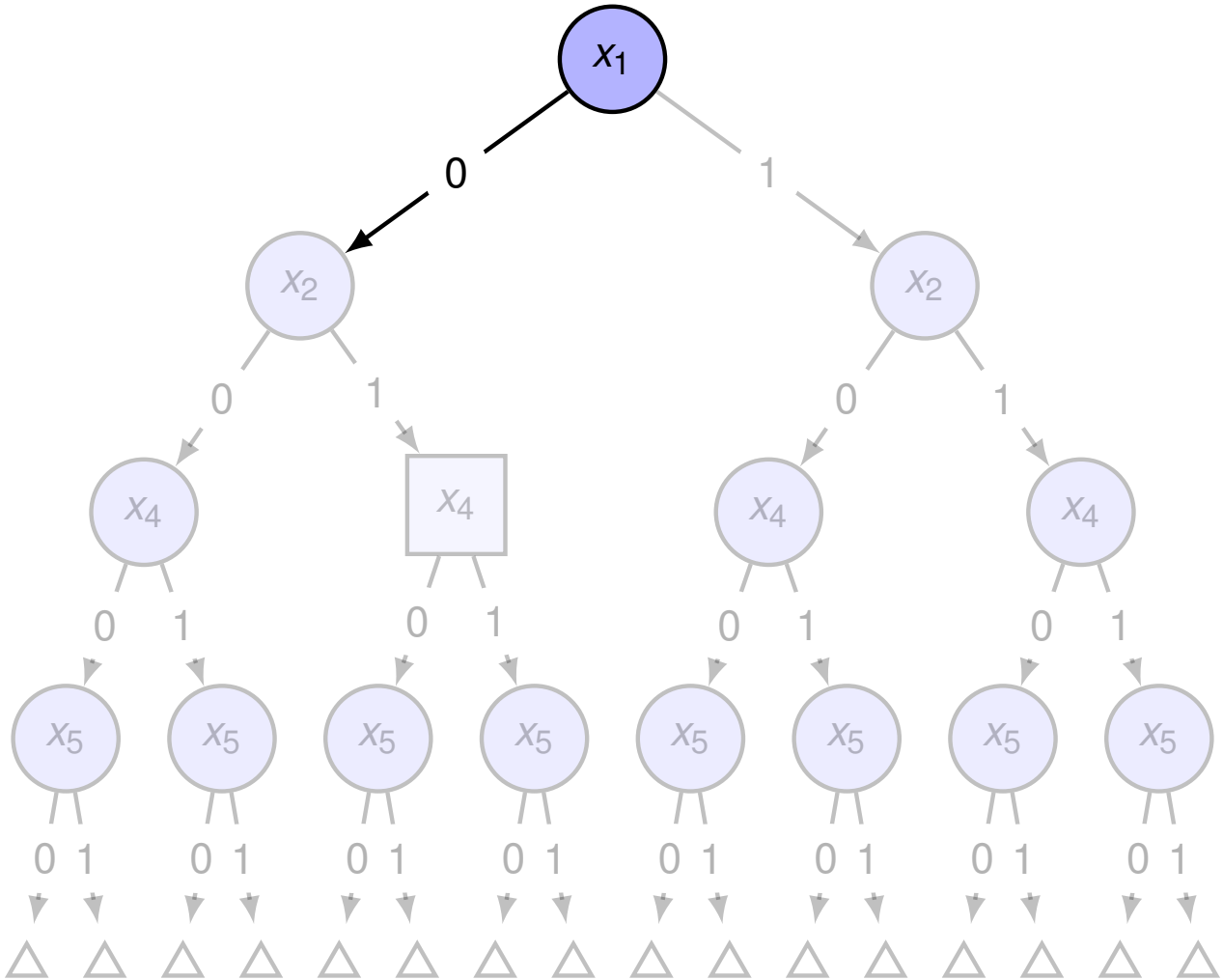
	MiniSAT	Shatter	BreakID	MiniSym
TOTAL	325	316	334	**336**

(b) With `bliss`

Table: Comparison on SAT instances

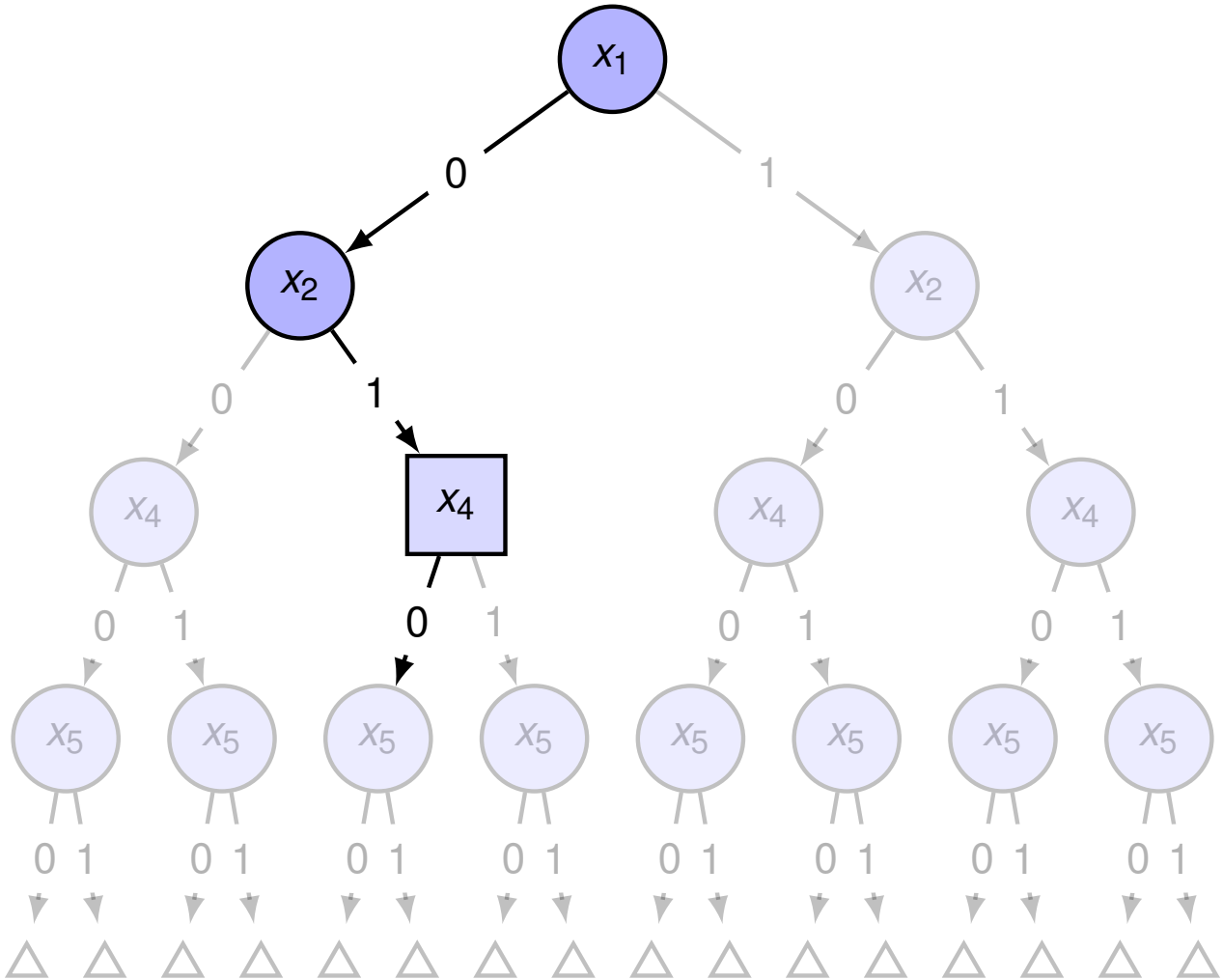
Using symmetries to accelerate the tree traversal

$g = (x_1\ \ x_2)(x_4\ \ x_5)(x_7\ \ x_8)$



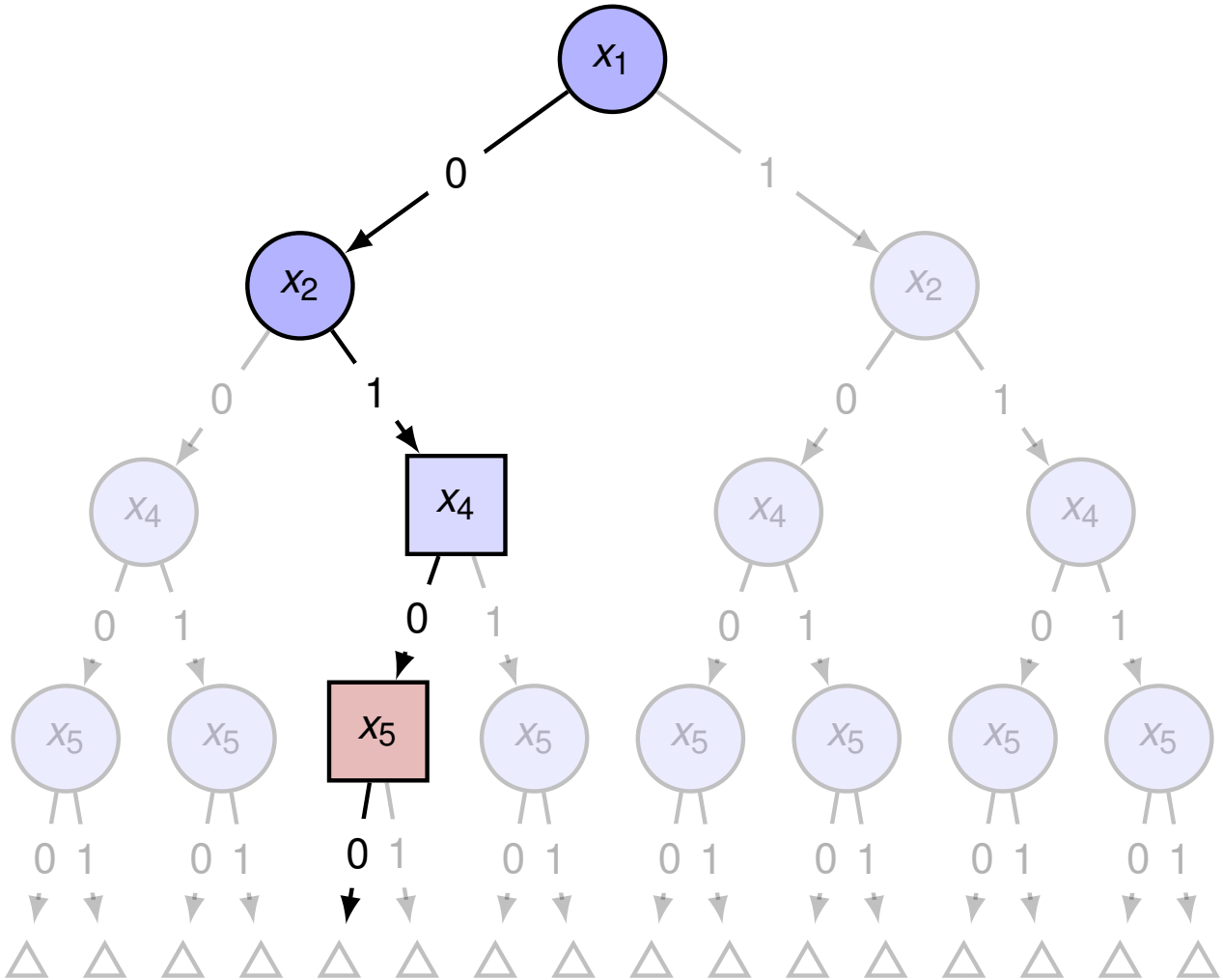
Using symmetries to accelerate the tree traversal

$g = (x_1\ \ x_2)(x_4\ \ x_5)(x_7\ \ x_8)$



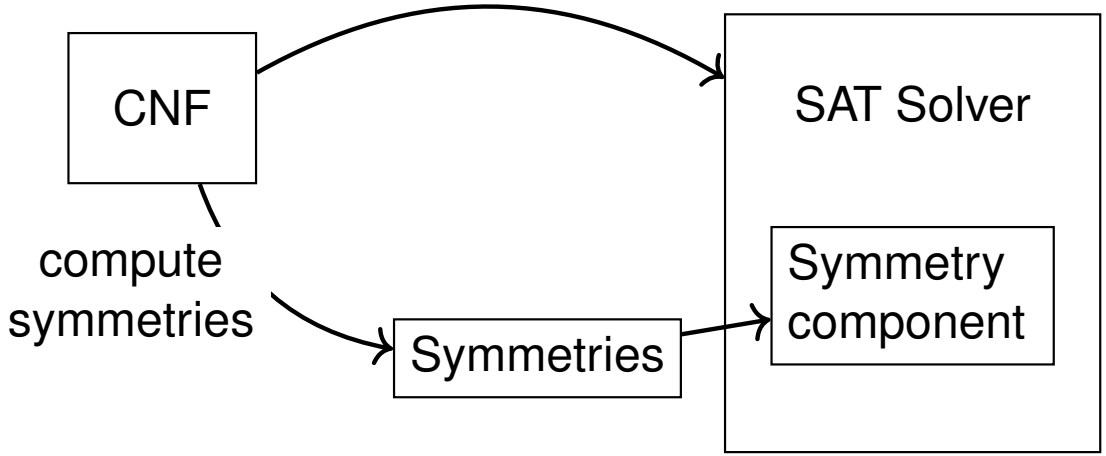
Using symmetries to accelerate the tree traversal

$g = (x_1 \;\; x_2)(x_4 \;\; x_5)(x_7 \;\; x_8)$



Use symmetries to deduce symmetrical facts.

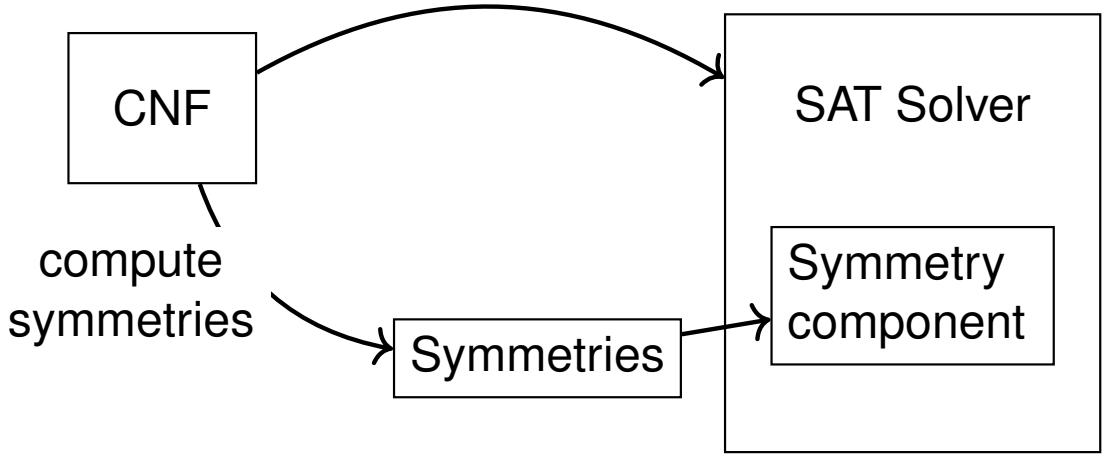
Dynamic Symmetry Breaking



Different approaches:

- Symmchaff [Sab05]
- Symmetry Propagation (SP) [DBdC+12]
- Symmetry Learning Scheme (SLS) [BNOS10]
- Symmetry Explanation Leaning (SEL) [DBB17]
- ...

Dynamic Symmetry Breaking



Different approaches:

- Symmchaff [Sab05]
- Symmetry Propagation (SP) [DBdC+12]
- Symmetry Learning Scheme (SLS) [BNOS10]
- Symmetry Explanation Leaning (SEL) [DBB17]
- ...

Pros/Cons:

- Works well on many symmetric instances
- Cannot handle some instances solved by static approach

ESBP + SP

Compose the symmetry propagation and the ESBP

prune the decision tree while accelerating its traversal

Problems:

- ESBP breaks symmetries (incrementally)
- SP considers the manipulated symmetries valid all time

In a hybrid approach, SP must be able to identify
valid symmetries

Local symmetries

Formula $\leftarrow$ (Symmetries)

$\omega_1 \leftarrow$ (Local symmetries)
$\omega_2 \leftarrow$ (Local symmetries)
$\omega_3 \leftarrow$ (Local symmetries)
$\omega_4 \leftarrow$ (Local symmetries)

Macro level $\rightarrow$ Micro level

Local symmetries

Formula $\leftarrow$ (Symmetries)

$\omega_1 \leftarrow$ (Local symmetries)

$\omega_2 \leftarrow$ (Local symmetries)

$\omega_3 \leftarrow$ (Local symmetries)

$\omega_4 \leftarrow$ (Local symmetries)

ω_5

Macro level $\rightarrow$ Micro level

Local symmetries

Formula $\leftarrow$ (Symmetries)

$\omega_1 \leftarrow$ (Local symmetries)
$\omega_2 \leftarrow$ (Local symmetries)
$\omega_3 \leftarrow$ (Local symmetries)
$\omega_4 \leftarrow$ (Local symmetries)
$\omega_5 \leftarrow$ (Local symmetries)

Macro level $\rightarrow$ Micro level

Compute valid local symmetries on-the-fly at a minimal cost.

Experimental results

Benchmark:

- from SAT contests 2012 – 2018
- retain only instances for which `bliss` finds significant symmetries in 1000s
- 1400 symmetric instances (out of 4000)

Setup:

- Three tools
  - MiniSat SP (Minisat with Symmetry Propagation)
  - MiniSat ESBP (Minisat with CDCL[Sym])
  - **Minisat ESBP-SP (our approach)**
- 7200s timeout

Results:

Solver	PAR-2	ALL	SAT	UNSAT
SP	1674h00	876	406	470
ESBP	1578h30	904	416	488
ESBP-SP	**1570h15**	**911**	**420**	**491**

Conclusion

- A new dynamic symmetry breaking approach
  - Generation of SBP on the fly
  - Package as a library cosy usable with any CDCL solver
  - Overcomes drawbacks of the existing approaches

- A new hybrid approach (ESBP-SP)
  - Take advantage of static and dynamic approach
  - Introduce local symmetries

Perspectives

- Combination of CDCL[Sym] with other dynamic symmetry breaking approach

- Exploitation of partial symmetries

Perspectives

- Combination of CDCL[Sym] with other dynamic symmetry breaking approach

- Exploitation of partial symmetries

Thanks !


Fadi A. Aloul, Karem A. Sakallah, and Igor L. Markov.
Efficient symmetry breaking for boolean satisfiability.
IEEE Trans. Computers, 55(5):549–558, 2006.


Belaid Benhamou, Tarek Nabhani, Richard Ostrowski, and Mohamed Reda Saidi.

Enhancing clause learning by symmetry in sat solvers.
In *2010 22nd IEEE International Conference on Tools with Artificial Intelligence*, volume 1, pages 329–335. IEEE, 2010.


Stephen A Cook.
The complexity of theorem-proving procedures.
In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.


Jo Devriendt, Bart Bogaerts, and Maurice Bruynooghe.
Symmetric explanation learning: Effective dynamic symmetry handling for sat.
In *International Conference on Theory and Applications of Satisfiability Testing*, pages 83–100. Springer, 2017.


Jo Devriendt, Bart Bogaerts, Maurice Bruynooghe, and Marc Denecker.
Improved static symmetry breaking for sat.
In *International Conference on Theory and Applications of Satisfiability Testing*, pages 104–122. Springer, 2016.


Jo Devriendt, Bart Bogaerts, Broes de Cat, Marc Denecker, and Christopher Mears.
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, 2012.


Martin Davis, George Logemann, and Donald Loveland.
A machine program for theorem-proving.
Commun. ACM, 5(7):394–397, July 1962.

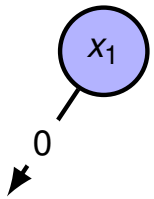

Joao P Marques-Silva and Karem A Sakallah.
Grasp: A search algorithm for propositional satisfiability.
IEEE Transactions on Computers, 48(5):506–521, 1999.


Ashish Sabharwal.
Symchaff: A structure-aware satisfiability solver.
In *AAAI*, volume 5, pages 467–474, 2005.

CDCL in action TODO

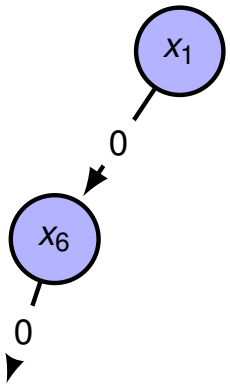
$$\omega_1 = \{x_1, x_2, x_3\}$$
$$\omega_2 = \{x_4, x_5, x_6\}$$
$$\omega_3 = \{\neg x_1, \neg x_5\}$$
$$\omega_4 = \{\neg x_2, \neg x_4\}$$
$$\omega_5 = \{\neg x_3, \neg x_4\}$$
$$\omega_6 = \{\neg x_3, \neg x_6\}$$

CDCL in action TODO



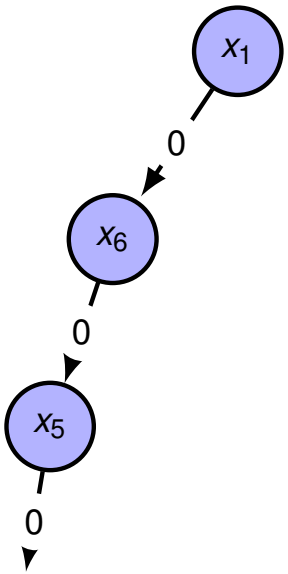
$$\omega_1 = \{x_1, x_2, x_3\}$$
$$\omega_2 = \{x_4, x_5, x_6\}$$
$$\omega_3 = \{\neg x_1, \neg x_5\}$$
$$\omega_4 = \{\neg x_2, \neg x_4\}$$
$$\omega_5 = \{\neg x_3, \neg x_4\}$$
$$\omega_6 = \{\neg x_3, \neg x_6\}$$

CDCL in action TODO



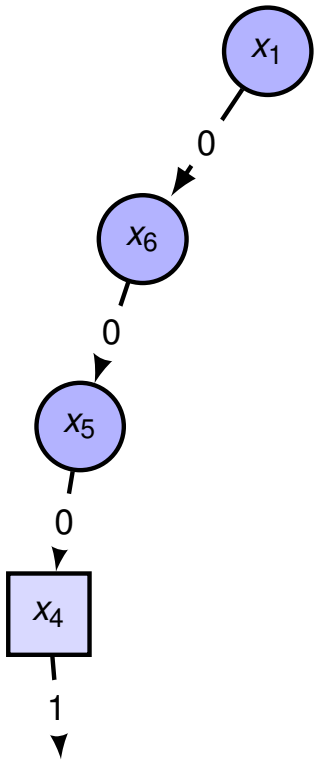
$$\omega_1 = \{x_1, x_2, x_3\}$$
$$\omega_2 = \{x_4, x_5, x_6\}$$
$$\omega_3 = \{\neg x_1, \neg x_5\}$$
$$\omega_4 = \{\neg x_2, \neg x_4\}$$
$$\omega_5 = \{\neg x_3, \neg x_4\}$$
$$\omega_6 = \{\neg x_3, \neg x_6\}$$

CDCL in action TODO



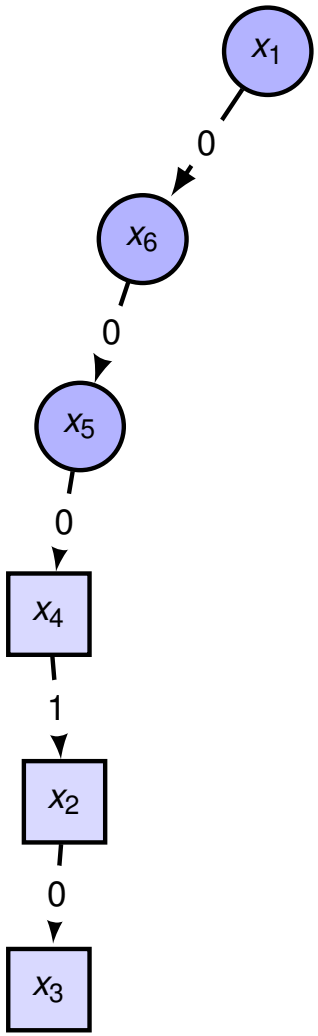
$$\omega_1 = \{x_1, x_2, x_3\}$$
$$\omega_2 = \{x_4, x_5, x_6\}$$
$$\omega_3 = \{\neg x_1, \neg x_5\}$$
$$\omega_4 = \{\neg x_2, \neg x_4\}$$
$$\omega_5 = \{\neg x_3, \neg x_4\}$$
$$\omega_6 = \{\neg x_3, \neg x_6\}$$

CDCL in action TODO



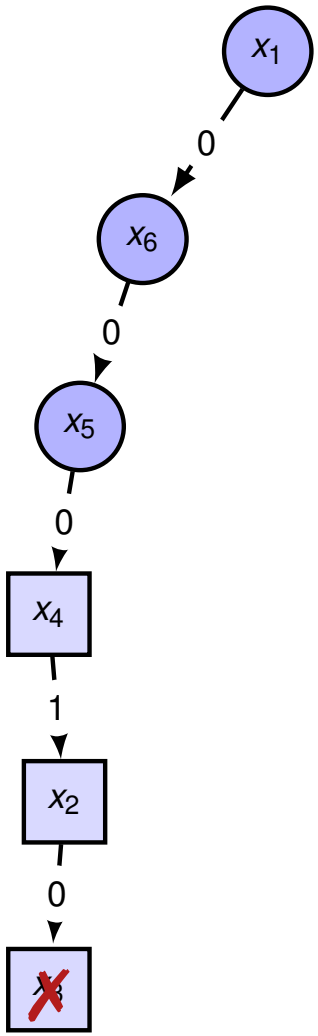
$$\omega_1 = \{x_1, x_2, x_3\}$$
$$\omega_2 = \{x_4, x_5, x_6\}$$
$$\omega_3 = \{\neg x_1, \neg x_5\}$$
$$\omega_4 = \{\neg x_2, \neg x_4\}$$
$$\omega_5 = \{\neg x_3, \neg x_4\}$$
$$\omega_6 = \{\neg x_3, \neg x_6\}$$

CDCL in action TODO



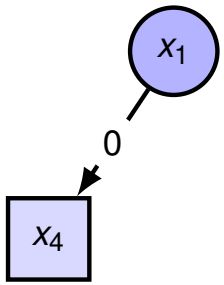
$$\omega_1 = \{x_1, x_2, x_3\}$$
$$\omega_2 = \{x_4, x_5, x_6\}$$
$$\omega_3 = \{\neg x_1, \neg x_5\}$$
$$\omega_4 = \{\neg x_2, \neg x_4\}$$
$$\omega_5 = \{\neg x_3, \neg x_4\}$$
$$\omega_6 = \{\neg x_3, \neg x_6\}$$

CDCL in action TODO



$$\omega_1 = \{x_1, x_2, x_3\}$$
$$\omega_2 = \{x_4, x_5, x_6\}$$
$$\omega_3 = \{\neg x_1, \neg x_5\}$$
$$\omega_4 = \{\neg x_2, \neg x_4\}$$
$$\omega_5 = \{\neg x_3, \neg x_4\}$$
$$\omega_6 = \{\neg x_3, \neg x_6\}$$

CDCL in action TODO



$$\omega_1 = \{x_1, x_2, x_3\}$$
$$\omega_2 = \{x_4, x_5, x_6\}$$
$$\omega_3 = \{\neg x_1, \neg x_5\}$$
$$\omega_4 = \{\neg x_2, \neg x_4\}$$
$$\omega_5 = \{\neg x_3, \neg x_4\}$$
$$\omega_6 = \{\neg x_3, \neg x_6\}$$

$$\omega_7 = \{x_1, \neg x_4\}$$

Weakly active symmetries

Logical consequence

When ω is satisfied in all satisfying assignments of φ, we say that ω is a logical consequence of φ, and we denote this by $\varphi \vdash \omega$.

Weakly active symmetries

Logical consequence

When ω is satisfied in all satisfying assignments of φ, we say that ω is a logical consequence of φ, and we denote this by $\varphi \vdash \omega$.

Weakly active symmetries

Let a subset $\delta \subseteq \alpha$, a symmetry σ of φ such that $\varphi \cup \delta \vdash \varphi \cup \alpha \wedge \sigma.\delta \subseteq \alpha$ then σ is weakly active symmetry.

Weakly active symmetries

Logical consequence

When ω is satisfied in all satisfying assignments of φ, we say that ω is a logical consequence of φ, and we denote this by $\varphi \vdash \omega$.

Weakly active symmetries

Let a subset $\delta \subseteq \alpha$, a symmetry σ of φ such that $\varphi \cup \delta \vdash \varphi \cup \alpha \wedge \sigma.\delta \subseteq \alpha$ then σ is weakly active symmetry.

Symmetry propagation

Let σ a weakly active symmetry, then

$$\varphi \cup \alpha \vdash \{l\} \Leftrightarrow \varphi \cup \alpha \vdash \sigma.\{l\}$$

Local symmetries

Logical consequence

When ω is satisfied in all satisfying assignments of φ, we say that ω is a logical consequence of φ, and we denote this by $\varphi \vdash \omega$.

Local Symmetries

Let φ be a formula. We define $L_{\omega,\varphi}$, the set of *local symmetries* for a clause ω, and with respect to a formula φ, as follows:

$$L_{\omega,\varphi} = \{\sigma \in \mathfrak{S} \mid \varphi \vdash \sigma.\omega\}$$

Local symmetries

Logical consequence

When ω is satisfied in all satisfying assignments of φ, we say that ω is a logical consequence of φ, and we denote this by $\varphi \vdash \omega$.

Local Symmetries

Let φ be a formula. We define $L_{\omega,\varphi}$, the set of *local symmetries* for a clause ω, and with respect to a formula φ, as follows:

$$L_{\omega,\varphi} = \{\sigma \in \mathfrak{S} \mid \varphi \vdash \sigma.\omega\}$$

We can state that:

$$\bigcap_{\omega \in \varphi} L_{\omega,\varphi} \subseteq G.$$

Computing local symmetries

Formula can be decomposed as : $\varphi = \varphi_o \cup \varphi_e \cup \varphi_d$ where

- φ_o is the set of the original clauses
- φ_e is the set of ESBPs
- φ_d is the set of deduced clauses.

Local symmetries

- $\omega \in \varphi_o, L_{\omega,\varphi} \supseteq G$
- $\omega \in \varphi_e, L_{\omega,\varphi} \supseteq Stab(\omega) = \{\sigma \in G \mid \omega = \sigma.\omega\}$
- $\omega \in \varphi_d, L_{\omega,\varphi} \supseteq (\bigcap_{\omega' \in \varphi_1} L_{\omega',\varphi}) \cup Stab(\omega)$
  where φ_1 is the set of clauses that derives ω.