SAT solvers: why are they working so well?

Laurent Simon Labri, Bordeaux, France

some common work with
Gilles Audemard CRIL, Lens, France
George Katsirelos INRA, Toulouse, France

Today's Itinerary

Introduction

What we know

Community Structure and LBD

Conclusion

Today's Itinerary

Introduction

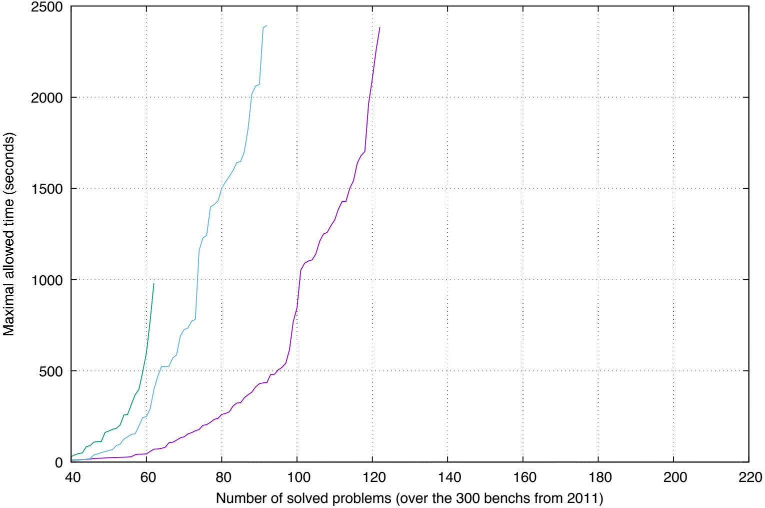
- Preliminaries
- DP-60
- DPLL-62
- SAT ingredients
- Literal Block Distance and glucose

What we know

Community Structure and LBD

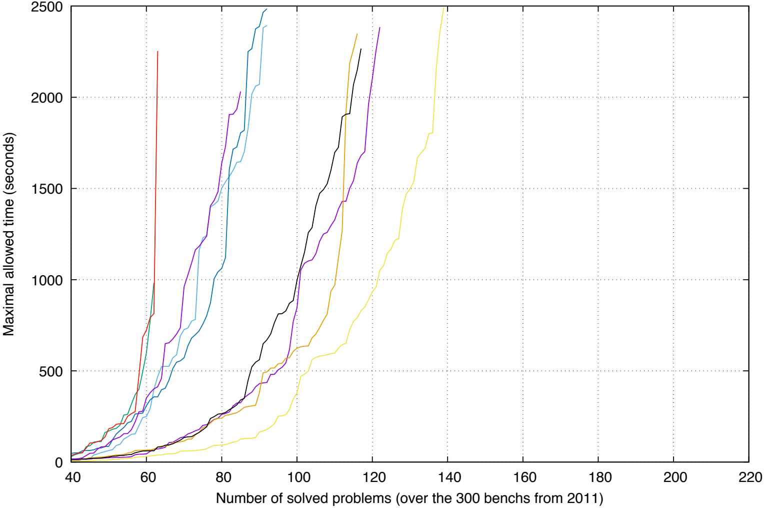
Conclusion

Performances of SAT Solvers, after 2001



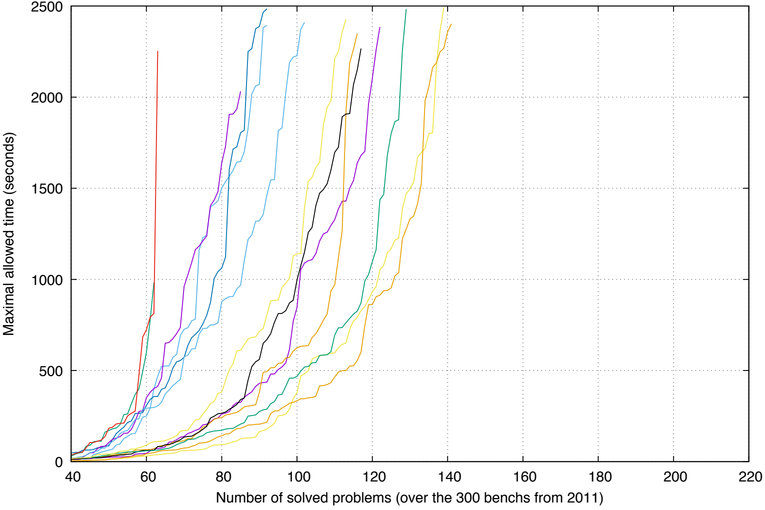
2002

Performances of SAT Solvers, after 2001



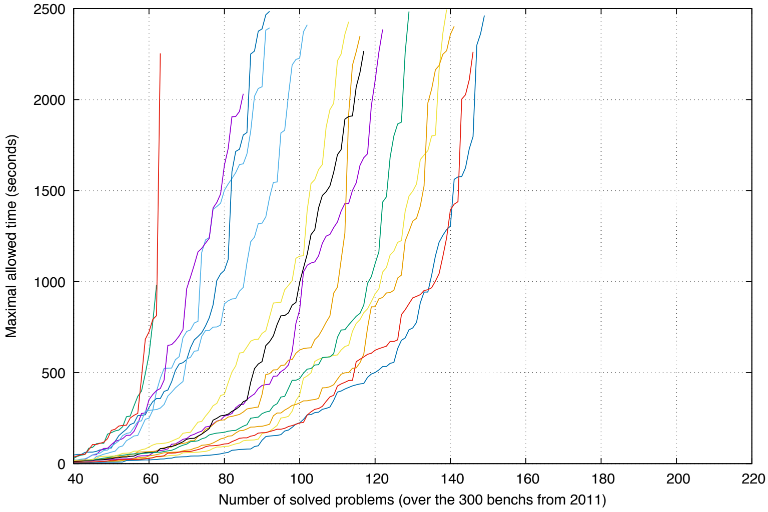
2003

Performances of SAT Solvers, after 2001



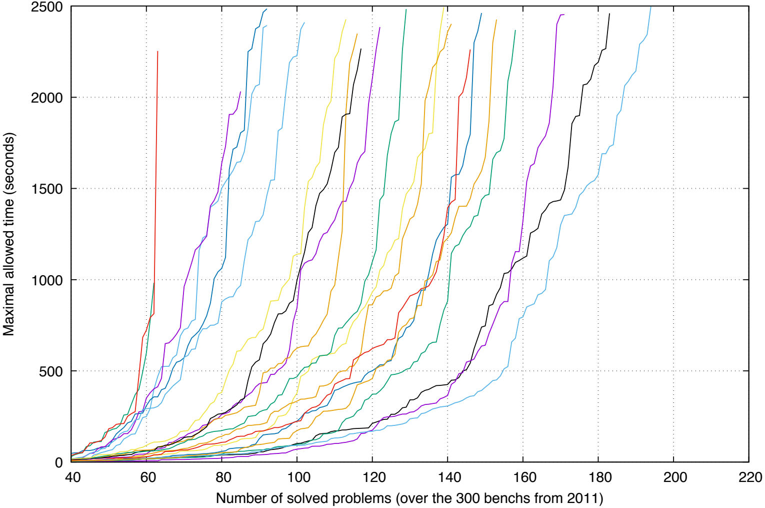
2005

Performances of SAT Solvers, after 2001



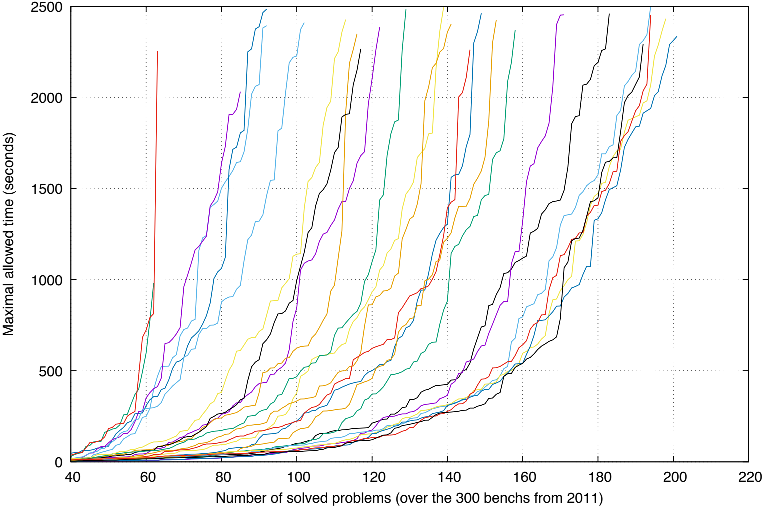
2007

Performances of SAT Solvers, after 2001



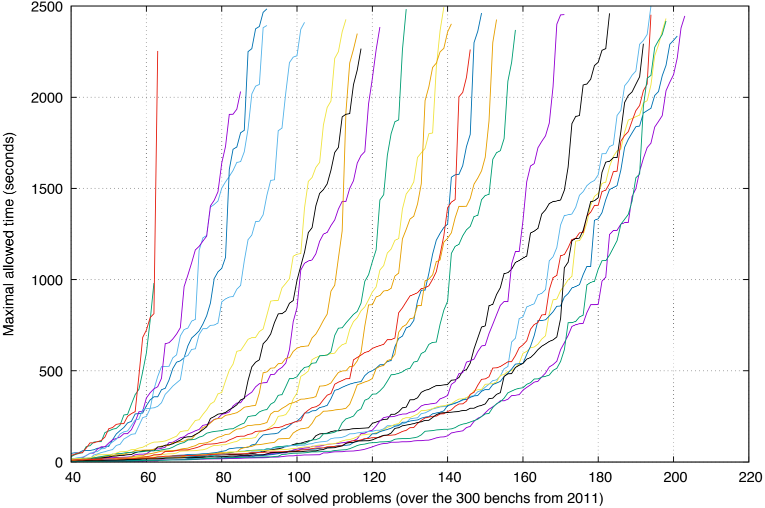
2009

Performances of SAT Solvers, after 2001



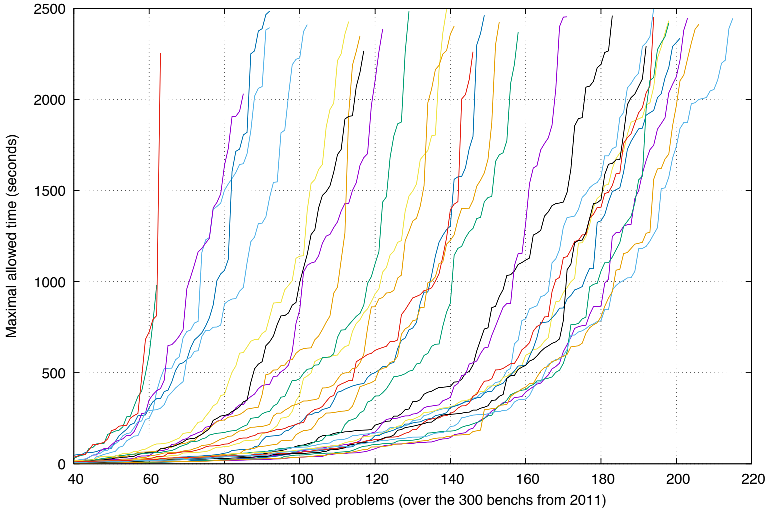
2011

Performances of SAT Solvers, after 2001



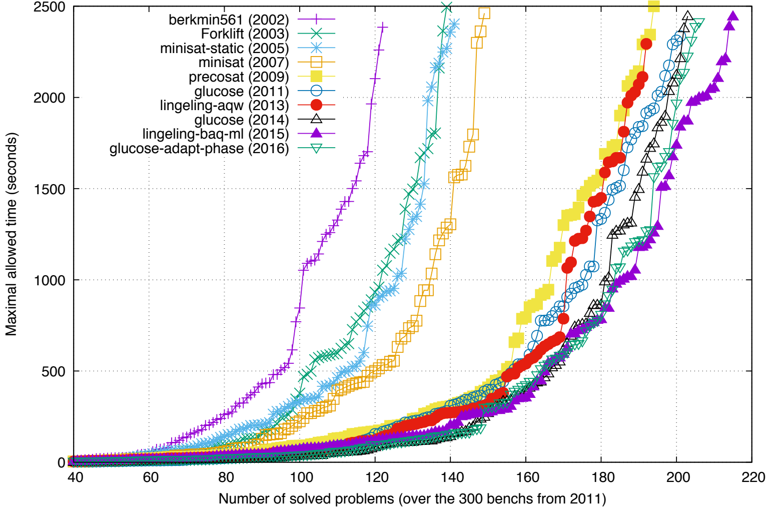
2014

Performances of SAT Solvers, after 2001



2016

Performances of SAT Solvers, after 2001



the winners

The firsts SAT steps

1958: Hilary Putnam and Martin Davis look for funding their research around propositional logic

« What we're interested in is good algorithms for propositional calculus » (NSA)

Before that, only inefficient methods (truth tables, . . .)

First papers

- *Computational Methods in The Propositional calculus* [Davis Putnam 1958][1]
- *A Computing Procedure for Quantification Theory* [Davis Putnam 1960]

[1]Rapport interne NSA

1960, already a first (kind of) competition!

« *The superiority of the present procedure (i.e. DP) over those previously available is indicated in part by the fact that a formula on which Gilmores routine for the IBM 704 causes* **the machine to compute for 21 minutes** *without obtaining a result was worked successfully by* **hand computation using the present method in 30 minutes** »

[Davis et Putnam 1960], page 202.

One of the reasons of the success of SAT is its competitions

Principles of DP-60

DP-60: forgets variables one after the other

Example : forgets x_1.

$x_1 \vee x_4$
$\overline{x_1} \vee x_4 \vee x_{14}$
$\overline{x_1} \vee \overline{x_3} \vee \overline{x_8}$
$x_1 \vee x_8 \vee x_{12}$
$x_1 \vee x_5 \vee \overline{x_9}$
$x_2 \vee x_{11}$
$\overline{x_3} \vee \overline{x_7} \vee x_{13}$
$\overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$
$x_8 \vee \overline{x_7} \vee \overline{x_9}$

Principles of DP-60

DP-60: forgets variables one after the other

Example : forgets x_1.

$x_1 \vee x_4$
$x_1 \vee x_8 \vee x_{12}$
$x_1 \vee x_5 \vee \overline{x_9}$

$\overline{x_1} \vee x_4 \vee x_{14}$
$\overline{x_1} \vee \overline{x_3} \vee \overline{x_8}$

$x_2 \vee x_{11}$
$\overline{x_3} \vee \overline{x_7} \vee x_{13}$
$\overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$
$x_8 \vee \overline{x_7} \vee \overline{x_9}$

Principles of DP-60

DP-60: forgets variables one after the other

Example : forgets x_1.

$$x_1 \vee \begin{pmatrix} x_4 \\ x_8 \vee x_{12} \\ x_5 \vee \overline{x_9} \end{pmatrix}$$

$$\overline{x_1} \vee \begin{pmatrix} x_4 \vee x_{14} \\ \overline{x_3} \vee \overline{x_8} \end{pmatrix}$$

$x_2 \vee x_{11}$

$\overline{x_3} \vee \overline{x_7} \vee x_{13}$

$\overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$

$x_8 \vee \overline{x_7} \vee \overline{x_9}$

Principles of DP-60

DP-60: forgets variables one after the other

Example : forgets x_1.

$$\begin{pmatrix} x_4 \\ x_8 \lor x_{12} \\ x_5 \lor \overline{x_9} \end{pmatrix} \lor \begin{pmatrix} x_4 \lor x_{14} \\ \overline{x_3} \lor \overline{x_8} \end{pmatrix}$$

$$x_2 \lor x_{11}$$
$$\overline{x_3} \lor \overline{x_7} \lor x_{13}$$
$$\overline{x_3} \lor \overline{x_7} \lor \overline{x_{13}} \lor x_9$$
$$x_8 \lor \overline{x_7} \lor \overline{x_9}$$

Principles of DP-60

DP-60: forgets variables one after the other

Example : forgets x_1.

$x_4 \vee x_{14}$
$x_4 \vee \overline{x_3} \vee \overline{x_8}$
$x_8 \vee x_{12} \vee x_4 \vee x_{14}$
$x_5 \vee \overline{x_9} \vee x_4 \vee x_{14}$
$x_5 \vee \overline{x_9} \vee \overline{x_3} \vee \overline{x_8}$

$x_2 \vee x_{11}$
$\overline{x_3} \vee \overline{x_7} \vee x_{13}$
$\overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$
$x_8 \vee \overline{x_7} \vee \overline{x_9}$

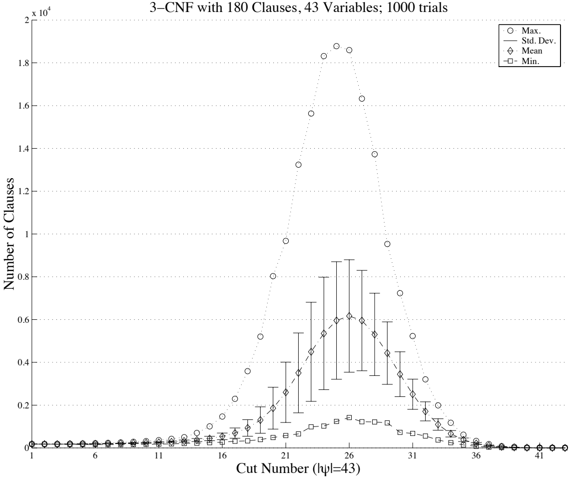
Principles of DP-60

DP-60: forgets variables one after the other

Example : forgets x_1.

$$x_4 \vee x_{14}$$
$$x_4 \vee \overline{x_3} \vee \overline{x_8}$$
$$x_8 \vee x_{12} \vee x_4 \vee x_{14}$$
$$x_5 \vee \overline{x_9} \vee x_4 \vee x_{14}$$
$$x_5 \vee \overline{x_9} \vee \overline{x_3} \vee \overline{x_8}$$
$$x_2 \vee x_{11}$$
$$\overline{x_3} \vee \overline{x_7} \vee x_{13}$$
$$\overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$$
$$x_8 \vee \overline{x_7} \vee \overline{x_9}$$

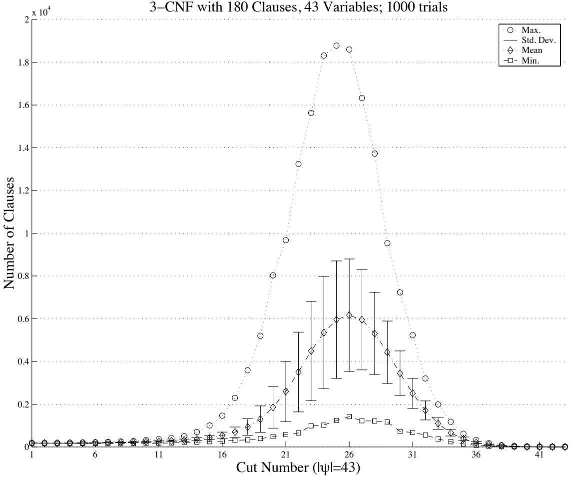
Untractable Space Problems



Combinatorial explosion, even on very small problems!

But possible on some very special cases (SAT pre-processing)

Untractable Space Problems



Combinatorial explosion, even on very small problems!

But possible on some very special cases (SAT pre-processing)

1962-2001: DPLL rules the world

Systematically explore the space of partial models (backtrack)

- Choose a literal
- Try to find a solution with this literal set to True
- If it is not possible:
  Finds a solution with this literal set to False

Backtrack search on partial models
Systematic (ordered) exploration ensures completeness

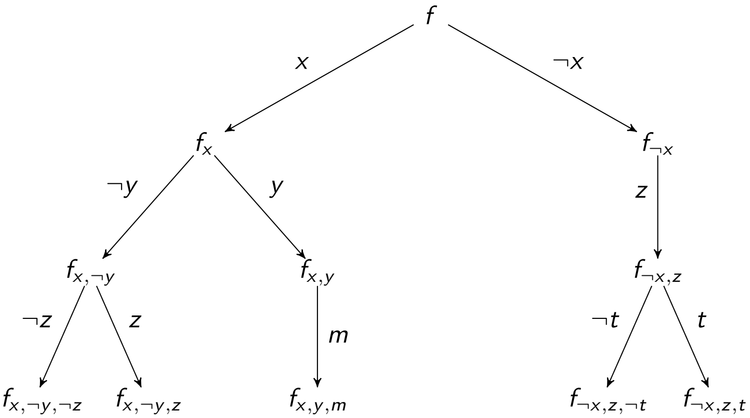
1962-2001: DPLL rules the world

Systematically explore the space of partial models (backtrack)

- Choose a literal
- Try to find a solution with this literal set to True
- If it is not possible:
  Finds a solution with this literal set to False

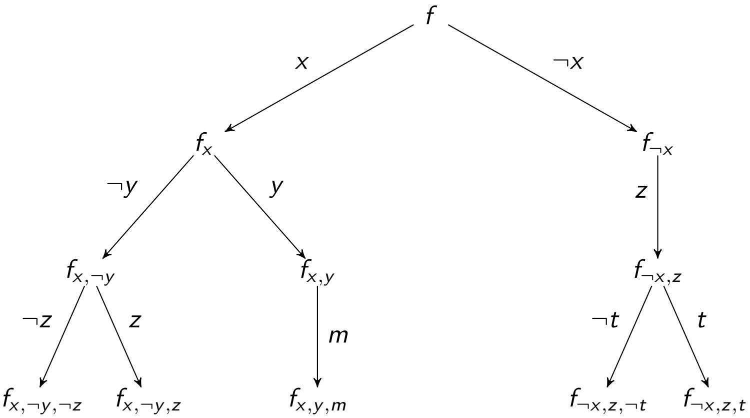
Backtrack search on partial models
Systematic (ordered) exploration ensures completeness

Backtrack search



- How to choose the right literal to branch on?
- First search for a model or a contradiction?

Backtrack search



- How to choose the right literal to branch on?
- First search for a model or a contradiction?

An example of DPLL

Formule	Simplified Formula	Partial Model
$x_1 \vee x_4$	$x_1 \vee x_4$	Lev. Lit. Back?
$\overline{x_1} \vee x_4 \vee x_{14}$	$\overline{x_1} \vee x_4 \vee x_{14}$	
$x_1 \vee \overline{x_3} \vee \overline{x_8}$	$x_1 \vee \overline{x_3} \vee \overline{x_8}$	
$x_1 \vee x_8 \vee x_{12}$	$x_1 \vee x_8 \vee x_{12}$	
$x_2 \vee x_{12}$	$x_2 \vee x_{12}$	
$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$	$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$	
$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$	$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$	
$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$	$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$	

$x1$ appears in 4 clauses and 1 binary clause

An example of DPLL

Formule

$x_1 \vee x_4$
$\overline{x_1} \vee x_4 \vee x_{14}$
$x_1 \vee \overline{x_3} \vee \overline{x_8}$
$x_1 \vee x_8 \vee x_{12}$
$x_2 \vee x_{12}$
$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$
$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$
$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Simplified Formula

$x_1 \vee x_4$
$\overline{x_1} \vee x_4 \vee x_{14}$
$x_1 \vee \overline{x_3} \vee \overline{x_8}$
$x_1 \vee x_8 \vee x_{12}$
$x_2 \vee x_{12}$
$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$
$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$
$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Partial Model

Lev.	Lit.	Back?
1	$\overline{x_1}$	(d)

x_4 appears in 1 unary clause

An example of DPLL

Formule

$x_1 \vee x_4$
$\overline{x_1} \vee x_4 \vee x_{14}$
$x_1 \vee \overline{x_3} \vee \overline{x_8}$
$x_1 \vee x_8 \vee x_{12}$
$x_2 \vee x_{12}$
$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$
$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$
$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Simplified Formula

$x_1 \vee x_4$
$\overline{x_1} \vee x_4 \vee x_{14}$
$x_1 \vee \overline{x_3} \vee \overline{x_8}$
$x_1 \vee x_8 \vee x_{12}$
$x_2 \vee x_{12}$
$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$
$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$
$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Partial Model

Lev.	Lit.	Back?
1	$\overline{x_1}$	(d)
+	x_4	

x_3 appears in 3 clauses incl. 1 (new) binary clause

An example of DPLL

Formule

$x_1 \vee x_4$

$\overline{x_1} \vee x_4 \vee x_{14}$

$x_1 \vee \overline{x_3} \vee \overline{x_8}$

$x_1 \vee x_8 \vee x_{12}$

$x_2 \vee x_{12}$

$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$

$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$

$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Simplified Formula

$x_1 \vee x_4$

$\overline{x_1} \vee x_4 \vee x_{14}$

$x_1 \vee \overline{x_3} \vee \overline{x_8}$

$x_1 \vee x_8 \vee x_{12}$

$x_2 \vee x_{12}$

$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$

$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$

$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Partial Model

Lev.	Lit.	Back?
1	$\overline{x_1}$	(d)
+	x_4	
2	x_3	(d)

$\overline{x_8}$ appears in one unary clause

An example of DPLL

Formule	Simplified Formula
$x_1 \vee x_4$	$x_1 \vee x_4$
$\overline{x_1} \vee x_4 \vee x_{14}$	$\overline{x_1} \vee x_4 \vee x_{14}$
$x_1 \vee \overline{x_3} \vee \overline{x_8}$	$x_1 \vee \overline{x_3} \vee \overline{x_8}$
$x_1 \vee x_8 \vee x_{12}$	$x_1 \vee x_8 \vee x_{12}$
$x_2 \vee x_{12}$	$x_2 \vee x_{12}$
$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$	$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$
$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$	$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$
$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$	$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Partial Model

Lev.	Lit.	Back?
1	$\overline{x_1}$	(d)
+	x_4	
2	x_3	(d)
+	$\overline{x_8}$	

x_{12} appears in 1 unary clause

An example of DPLL

Formule	Simplified Formula
$x_1 \vee x_4$	$x_1 \vee x_4$
$\overline{x_1} \vee x_4 \vee x_{14}$	$\overline{x_1} \vee x_4 \vee x_{14}$
$x_1 \vee \overline{x_3} \vee \overline{x_8}$	$x_1 \vee \overline{x_3} \vee \overline{x_8}$
$x_1 \vee x_8 \vee x_{12}$	$x_1 \vee x_8 \vee x_{12}$
$x_2 \vee x_{12}$	$x_2 \vee x_{12}$
$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$	$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$
$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$	$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$
$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$	$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Partial Model

Lev.	Lit.	Back?
1	$\overline{x_1}$	(d)
+	x_4	
2	x_3	(d)
+	$\overline{x_8}$	
+	x_{12}	

x_{13}, $\overline{x_7}$ appear in unary clauses

An example of DPLL

Formule	Simplified Formula
$x_1 \lor x_4$	$x_1 \lor x_4$
$\overline{x_1} \lor x_4 \lor x_{14}$	$\overline{x_1} \lor x_4 \lor x_{14}$
$x_1 \lor \overline{x_3} \lor \overline{x_8}$	$x_1 \lor \overline{x_3} \lor \overline{x_8}$
$x_1 \lor x_8 \lor x_{12}$	$x_1 \lor x_8 \lor x_{12}$
$x_2 \lor x_{12}$	$x_2 \lor x_{12}$
$\overline{x_3} \lor \overline{x_{12}} \lor x_{13}$	$\overline{x_3} \lor \overline{x_{12}} \lor x_{13}$
$\overline{x_3} \lor x_7 \lor \overline{x_{13}}$	$\overline{x_3} \lor x_7 \lor \overline{x_{13}}$
$x_8 \lor \overline{x_7} \lor \overline{x_{12}}$	$x_8 \lor \overline{x_7} \lor \overline{x_{12}}$

Partial Model

Lev.	Lit.	Back?
1	$\overline{x_1}$	(d)
+	x_4	
2	x_3	(d)
+	$\overline{x_8}$	
+	x_{12}	
+	x_{13}	

x_7, $\overline{x_7}$ appear in unary clauses

An example of DPLL

Formule

$x_1 \vee x_4$

$\overline{x_1} \vee x_4 \vee x_{14}$

$x_1 \vee \overline{x_3} \vee \overline{x_8}$

$x_1 \vee x_8 \vee x_{12}$

$x_2 \vee x_{12}$

$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$

$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$

$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Simplified Formula

$x_1 \vee x_4$

$\overline{x_1} \vee x_4 \vee x_{14}$

$x_1 \vee \overline{x_3} \vee \overline{x_8}$

$x_1 \vee x_8 \vee x_{12}$

$x_2 \vee x_{12}$

$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$

$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$

$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Partial Model

Lev.	Lit.	Back?
1	$\overline{x_1}$	(d)
+	x_4	
2	x_3	(d)
+	$\overline{x_8}$	
+	x_{12}	
+	x_{13}	
+	$\overline{x_7}$	

Conflict! Undo everything until last decision

An example of DPLL

Formule

$x_1 \vee x_4$

$\overline{x_1} \vee x_4 \vee x_{14}$

$x_1 \vee \overline{x_3} \vee \overline{x_8}$

$x_1 \vee x_8 \vee x_{12}$

$x_2 \vee x_{12}$

$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$

$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$

$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Simplified Formula

$x_1 \vee x_4$

$\overline{x_1} \vee x_4 \vee x_{14}$

$x_1 \vee \overline{x_3} \vee \overline{x_8}$

$x_1 \vee x_8 \vee x_{12}$

$x_2 \vee x_{12}$

$\overline{x_3} \vee \overline{x_{12}} \vee x_{13}$

$\overline{x_3} \vee x_7 \vee \overline{x_{13}}$

$x_8 \vee \overline{x_7} \vee \overline{x_{12}}$

Partial Model

Lev.	Lit.	Back?
1	$\overline{x_1}$	(d)
+	x_4	
*	$\overline{x_3}$	

Now, $\overline{x_3}$ is not a decision

From LookAhead to Lookback

All solvers are now turned to lazily detect Unit Propagation

No way to maintain counters for “smart” branching

Look ahead heuristics were “easy” to understand

Look back heuristics are very hard to study

Ingredients of an efficient SAT solver

Preprocessing
(and inprocessing)

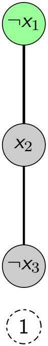
Restarting

Branching

Conflict Analysis

Clause Database Cleaning

CDCL Principles



$x_1 \vee x_2$

$\neg x_2 \vee \neg x_3$

$\neg x_2 \vee \neg x_4 \vee \neg x_5$

$x_3 \vee x_5 \vee x_6$

$x_7 \vee \neg x_6 \vee \neg x_8$

$\neg x_4 \vee x_8 \vee x_9$

$x_{10} \vee \neg x_9 \vee x_{11}$

$\neg x_{11} \vee x_8 \vee \neg x_{12}$

$x_{12} \vee \neg x_{13}$

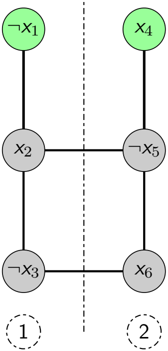
$\neg x_6 \vee x_{12} \vee x_{15}$

$x_{13} \vee \neg x_{14} \vee \neg x_{16}$

$\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



$x_1 \vee x_2$ $\quad \neg x_2 \vee \neg x_3$

$\neg x_2 \vee \neg x_4 \vee \neg x_5$ $\quad x_3 \vee x_5 \vee x_6$

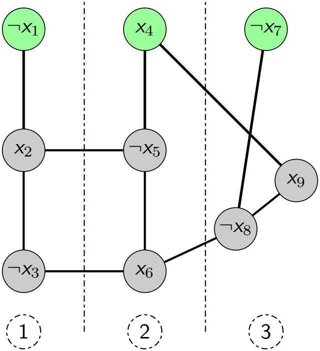
$x_7 \vee \neg x_6 \vee \neg x_8$ $\quad \neg x_4 \vee x_8 \vee x_9$

$x_{10} \vee \neg x_9 \vee x_{11}$ $\quad \neg x_{11} \vee x_8 \vee \neg x_{12}$ $\quad x_{12} \vee \neg x_{13}$

$\neg x_6 \vee x_{12} \vee x_{15}$ $\quad x_{13} \vee \neg x_{14} \vee \neg x_{16}$ $\quad \neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

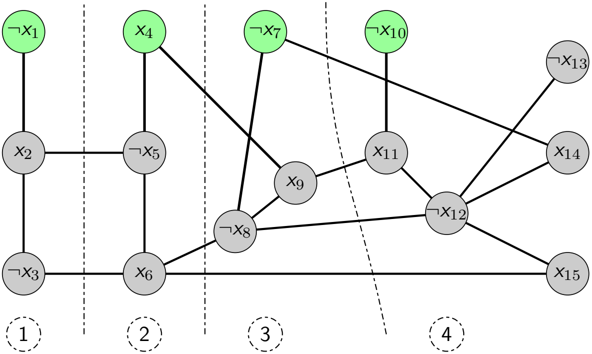
CDCL Principles



$x_1 \vee x_2$	$\neg x_2 \vee \neg x_4 \vee \neg x_5$	$x_7 \vee \neg x_6 \vee \neg x_8$	$x_{10} \vee \neg x_9 \vee x_{11}$	$\neg x_6 \vee x_{12} \vee x_{15}$
$\neg x_2 \vee \neg x_3$	$x_3 \vee x_5 \vee x_6$	$\neg x_4 \vee x_8 \vee x_9$	$\neg x_{11} \vee x_8 \vee \neg x_{12}$	$x_{13} \vee \neg x_{14} \vee \neg x_{16}$
			$x_{12} \vee \neg x_{13}$	$\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



$x_1 \vee x_2$
$\neg x_2 \vee \neg x_3$

$\neg x_2 \vee \neg x_4 \vee \neg x_5$
$x_3 \vee x_5 \vee x_6$

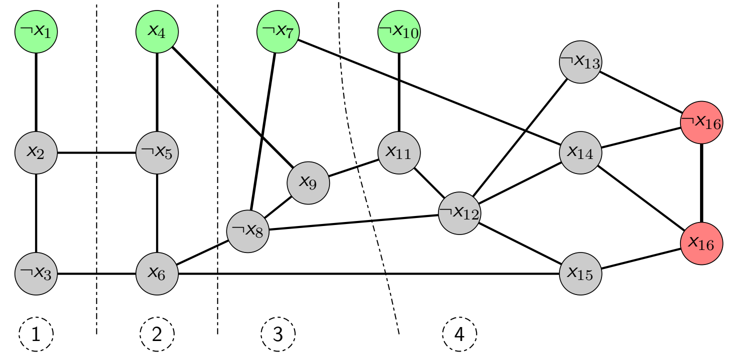
$x_7 \vee \neg x_6 \vee \neg x_8$
$\neg x_4 \vee x_8 \vee x_9$

$x_{10} \vee \neg x_9 \vee x_{11}$
$\neg x_{11} \vee x_8 \vee \neg x_{12}$
$x_{12} \vee \neg x_{13}$

$\neg x_6 \vee x_{12} \vee x_{15}$
$x_{13} \vee \neg x_{14} \vee \neg x_{16}$
$\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



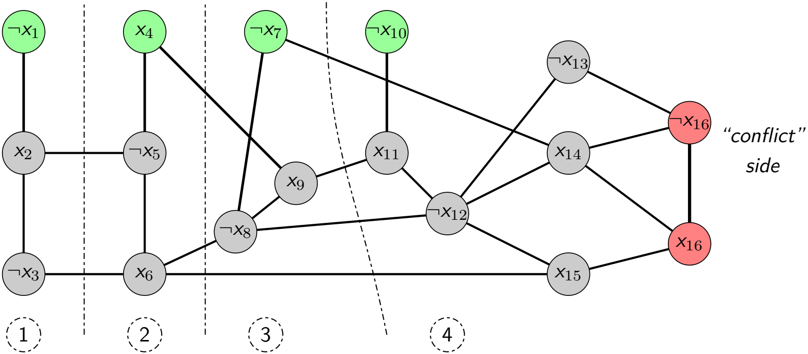
$x_1 \vee x_2$ $\quad \neg x_2 \vee \neg x_4 \vee \neg x_5$ $\quad x_7 \vee \neg x_6 \vee \neg x_8$ $\quad x_{10} \vee \neg x_9 \vee x_{11}$ $\quad \neg x_6 \vee x_{12} \vee x_{15}$

$\neg x_2 \vee \neg x_3$ $\quad x_3 \vee x_5 \vee x_6$ $\quad \neg x_4 \vee x_8 \vee x_9$ $\quad \neg x_{11} \vee x_8 \vee \neg x_{12}$ $\quad x_{13} \vee \neg x_{14} \vee \neg x_{16}$

$x_{12} \vee \neg x_{13}$ $\quad \neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



$x_1 \vee x_2$
$\neg x_2 \vee \neg x_3$

$\neg x_2 \vee \neg x_4 \vee \neg x_5$
$x_3 \vee x_5 \vee x_6$

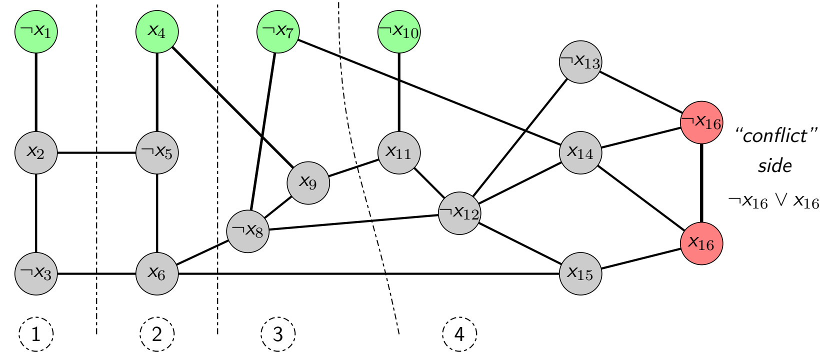
$x_7 \vee \neg x_6 \vee \neg x_8$
$\neg x_4 \vee x_8 \vee x_9$

$x_{10} \vee \neg x_9 \vee x_{11}$
$\neg x_{11} \vee x_8 \vee \neg x_{12}$
$x_{12} \vee \neg x_{13}$

$\neg x_6 \vee x_{12} \vee x_{15}$
$x_{13} \vee \neg x_{14} \vee \neg x_{16}$
$\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



$x_1 \vee x_2$

$\neg x_2 \vee \neg x_3$

$\neg x_2 \vee \neg x_4 \vee \neg x_5$

$x_3 \vee x_5 \vee x_6$

$x_7 \vee \neg x_6 \vee \neg x_8$

$\neg x_4 \vee x_8 \vee x_9$

$x_{10} \vee \neg x_9 \vee x_{11}$

$\neg x_{11} \vee x_8 \vee \neg x_{12}$

$x_{12} \vee \neg x_{13}$

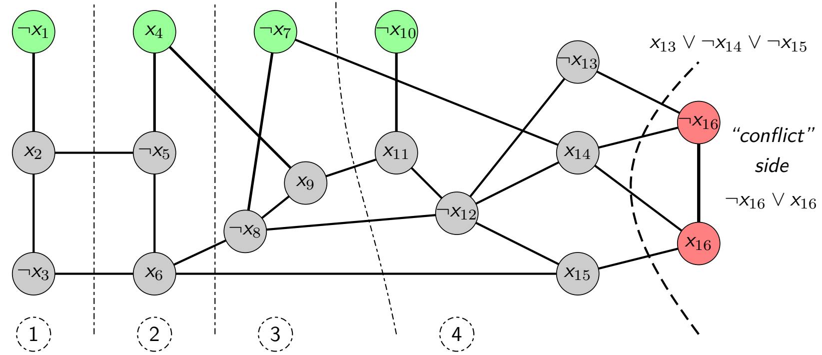
$\neg x_6 \vee x_{12} \vee x_{15}$

$x_{13} \vee \neg x_{14} \vee \neg x_{16}$

$\neg x_{15} \vee \neg x_{14} \vee x_{16}$

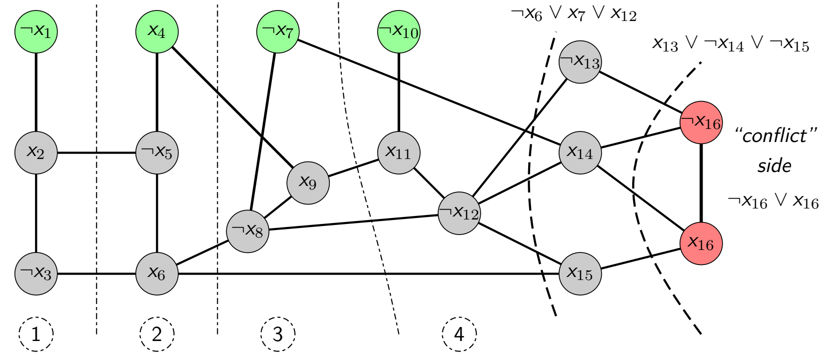
$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



$x_1 \vee x_2$	$\neg x_2 \vee \neg x_4 \vee \neg x_5$	$x_7 \vee \neg x_6 \vee \neg x_8$	$x_{10} \vee \neg x_9 \vee x_{11}$	$\neg x_6 \vee x_{12} \vee x_{15}$
$\neg x_2 \vee \neg x_3$	$x_3 \vee x_5 \vee x_6$	$\neg x_4 \vee x_8 \vee x_9$	$\neg x_{11} \vee x_8 \vee \neg x_{12}$	$x_{13} \vee \neg x_{14} \vee \neg x_{16}$
			$x_{12} \vee \neg x_{13}$	$\neg x_{15} \vee \neg x_{14} \vee x_{16}$
			$x_7 \vee x_{12} \vee x_{14}$	

CDCL Principles



$x_1 \vee x_2$
$\neg x_2 \vee \neg x_3$

$\neg x_2 \vee \neg x_4 \vee \neg x_5$
$x_3 \vee x_5 \vee x_6$

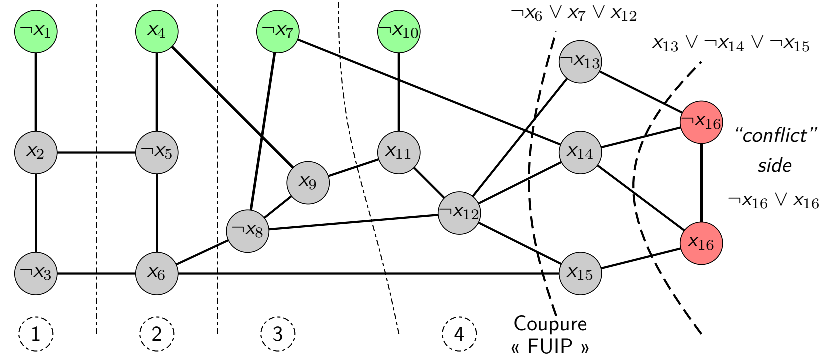
$x_7 \vee \neg x_6 \vee \neg x_8$
$\neg x_4 \vee x_8 \vee x_9$

$x_{10} \vee \neg x_9 \vee x_{11}$
$\neg x_{11} \vee x_8 \vee \neg x_{12}$
$x_{12} \vee \neg x_{13}$

$\neg x_6 \vee x_{12} \vee x_{15}$
$x_{13} \vee \neg x_{14} \vee \neg x_{16}$
$\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



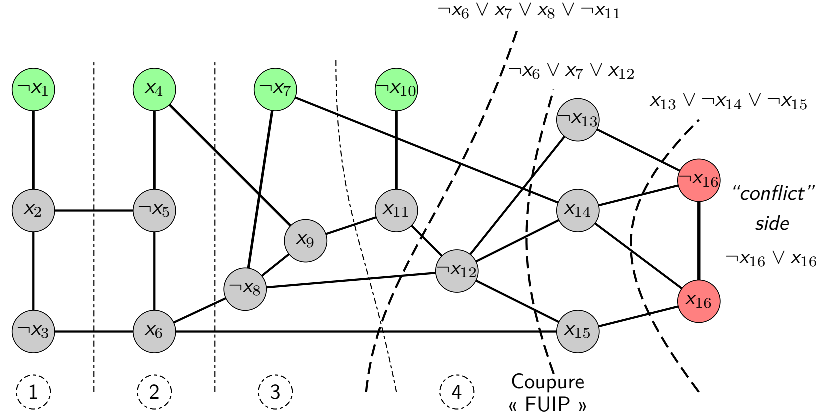
$x_1 \vee x_2$ $\neg x_2 \vee \neg x_4 \vee \neg x_5$ $x_7 \vee \neg x_6 \vee \neg x_8$ $x_{10} \vee \neg x_9 \vee x_{11}$ $\neg x_6 \vee x_{12} \vee x_{15}$

$\neg x_2 \vee \neg x_3$ $x_3 \vee x_5 \vee x_6$ $\neg x_4 \vee x_8 \vee x_9$ $\neg x_{11} \vee x_8 \vee \neg x_{12}$ $x_{13} \vee \neg x_{14} \vee \neg x_{16}$

$x_{12} \vee \neg x_{13}$ $\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



$x_1 \vee x_2$
$\neg x_2 \vee \neg x_3$

$\neg x_2 \vee \neg x_4 \vee \neg x_5$
$x_3 \vee x_5 \vee x_6$

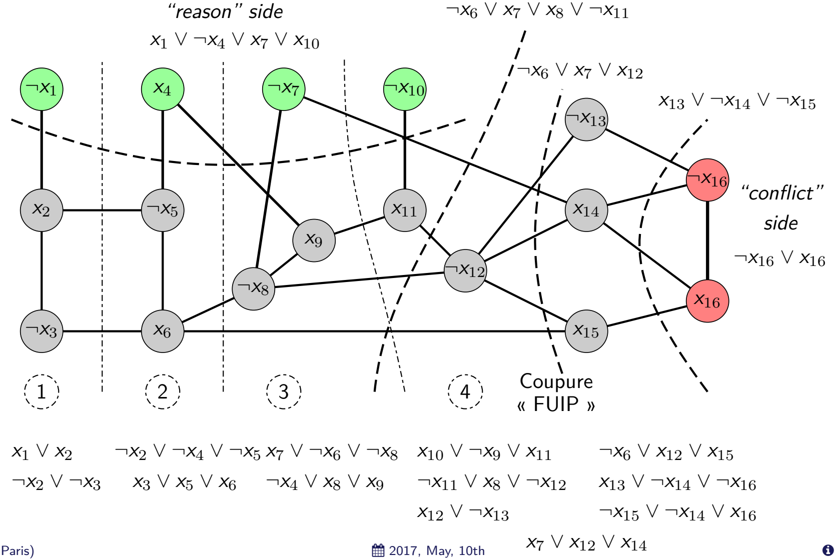
$x_7 \vee \neg x_6 \vee \neg x_8$
$\neg x_4 \vee x_8 \vee x_9$

$x_{10} \vee \neg x_9 \vee x_{11}$
$\neg x_{11} \vee x_8 \vee \neg x_{12}$
$x_{12} \vee \neg x_{13}$

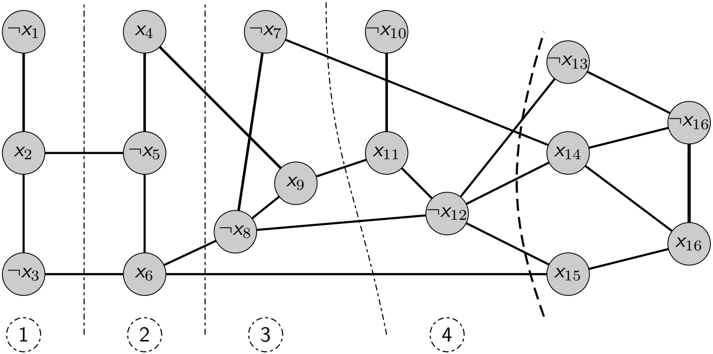
$\neg x_6 \vee x_{12} \vee x_{15}$
$x_{13} \vee \neg x_{14} \vee \neg x_{16}$
$\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



CDCL Principles



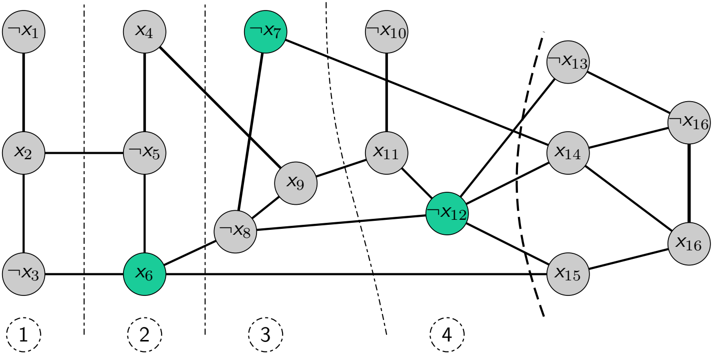
$x_1 \vee x_2$ $\quad$ $\neg x_2 \vee \neg x_4 \vee \neg x_5$ $\quad$ $x_7 \vee \neg x_6 \vee \neg x_8$ $\quad$ $x_{10} \vee \neg x_9 \vee x_{11}$ $\quad$ $\neg x_6 \vee x_{12} \vee x_{15}$

$\neg x_2 \vee \neg x_3$ $\quad$ $x_3 \vee x_5 \vee x_6$ $\quad$ $\neg x_4 \vee x_8 \vee x_9$ $\quad$ $\neg x_{11} \vee x_8 \vee \neg x_{12}$ $\quad$ $x_{13} \vee \neg x_{14} \vee \neg x_{16}$

$x_{12} \vee \neg x_{13}$ $\quad$ $\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



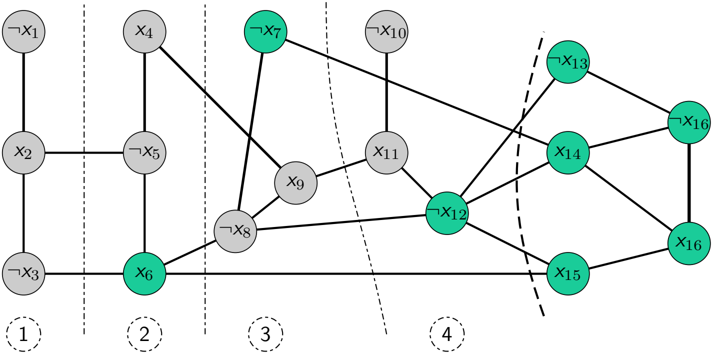
$x_1 \vee x_2$ $\quad$ $\neg x_2 \vee \neg x_4 \vee \neg x_5$ $\quad$ $x_7 \vee \neg x_6 \vee \neg x_8$ $\quad$ $x_{10} \vee \neg x_9 \vee x_{11}$ $\quad$ $\neg x_6 \vee x_{12} \vee x_{15}$

$\neg x_2 \vee \neg x_3$ $\quad$ $x_3 \vee x_5 \vee x_6$ $\quad$ $\neg x_4 \vee x_8 \vee x_9$ $\quad$ $\neg x_{11} \vee x_8 \vee \neg x_{12}$ $\quad$ $x_{13} \vee \neg x_{14} \vee \neg x_{16}$

$x_{12} \vee \neg x_{13}$ $\quad$ $\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL Principles



$x_1 \vee x_2$
$\neg x_2 \vee \neg x_3$

$\neg x_2 \vee \neg x_4 \vee \neg x_5$
$x_3 \vee x_5 \vee x_6$

$x_7 \vee \neg x_6 \vee \neg x_8$
$\neg x_4 \vee x_8 \vee x_9$

$x_{10} \vee \neg x_9 \vee x_{11}$
$\neg x_{11} \vee x_8 \vee \neg x_{12}$
$x_{12} \vee \neg x_{13}$

$\neg x_6 \vee x_{12} \vee x_{15}$
$x_{13} \vee \neg x_{14} \vee \neg x_{16}$
$\neg x_{15} \vee \neg x_{14} \vee x_{16}$

$x_7 \vee x_{12} \vee x_{14}$

CDCL solvers are complex systems

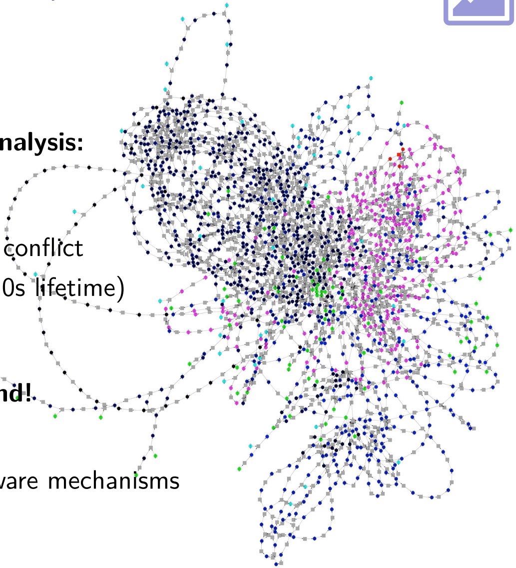
We have a lot of open problems around these questions:
“ Understand what we have implemented ”

It's ok if we don't fully "understand" our code

- Very fast and unpredictable
- Work well on real-world instances, but how to define such a structure?
- All components are tightly connected, side effects are everywhere
- There is no "one-simple reason" explaining their performance (supposition)
- At least we know that we don't know

Idea behind glucose
A real experimental study of CDCL solvers

CDCL solvers are complex systems – Illustration



Example of a real conflict analysis:

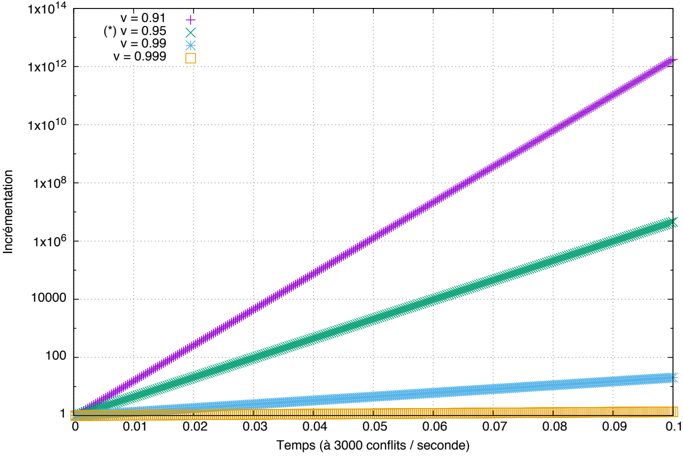
- Many resolutions at each conflict
- Very reactive VSIDS (1/10s lifetime)

But: A clear structure behind!

Let's search for structure aware mechanisms in CDCL solvers!

High dynamicity of the heuristics

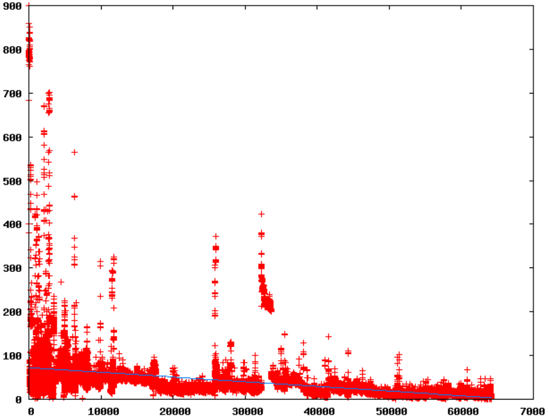
Exponential Increasing of Heuristics Bumps



After each conflict, the increment is multipled by $1/v$ ($v = 0.95$)

Number of decisions before reaching a conflict

een-pico-prop05-50 – UNSAT – 13,000 vars and 65,000 clauses

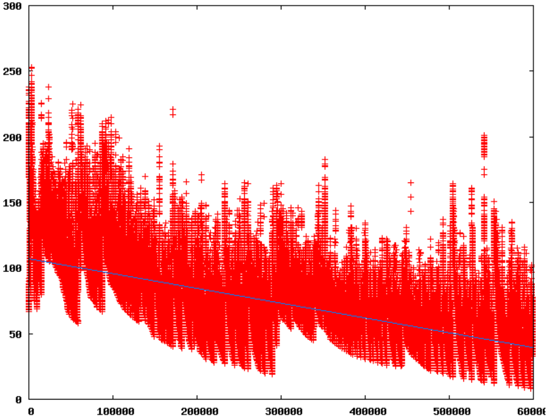


een-pico-prop05-50 – UNSAT – 13,000 vars and 65,000 clauses

Experimental observation behind glucose (2009):
A good CDCL learns clauses that reduces the number of decisions to make

Number of decisions before reaching a conflict

grieu-vmpc-s05-25 – sat – 625 vars and 76,000 clauses

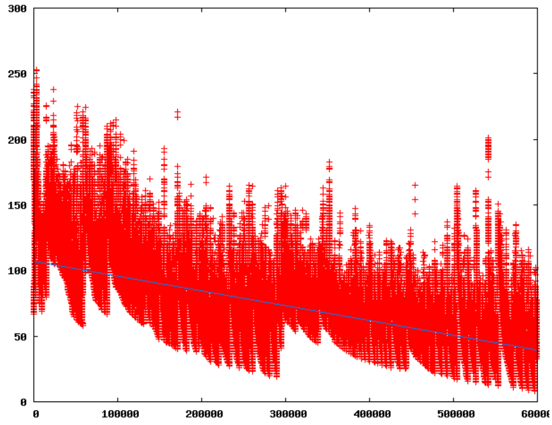


grieu-vmpc-s05-25 – SAT – 625 vars and 76,000 clauses

Experimental observation behind glucose (2009):
A good CDCL learns clauses that reduces the number of decisions to make

Number of decisions before reaching a conflict

grieu-vmpc-s05-25 – sat – 625 vars and 76,000 clauses



grieu-vmpc-s05-25 – SAT – 625 vars and 76,000 clauses

Experimental observation behind glucose (2009):
A good CDCL learns clauses that reduces the number of decisions to make

Literal Block Distance (LBD) – initial idea (2009)

- One decision often creates a lot of propagated literals ("blocks")
  ➔ Those variables will probably be propagated together again and again
- Reducing decisions? Adds dependencies between independent blocks
- How? Add the strongest possible constraints between them

LBD of a learnt clause: number of prop. blocks of literals

- Small LBD scores are better
- The importance of "Glue Clauses" (LBD=2)
  - Only one literal from the last decision level (the assertive one)
  - This literal will be **glued** to the other block
  - Kept forever in glucose

The restart policy is based on LBD too

Why are they working so well?

We know how to build an efficient (single engine) SAT solver, but:

- CDCL is not DPLL
  ➔ *because of ultra-rapid restarts and agressive clause DB cleaning*
- Learning can be bad
  ➔ *we'll see that keeping all clauses is not a winning strategy*
- Restarting is not restarting
  ➔ *directly go to the same search space, by an another path*
- Luby-based restarts are dangerous
  ➔ *rare but very large windows are following a fixed restart strategy*
- What is the phase?
  ➔ *good to reach a solution or a contradiction?*
- "good" variables: top or bottom of the tree?
  ➔ *splitting on top, resolving on bottom variables*
- The completeness of glucose is really theoretical! ➔ *Too many restarts and forgetting*

Today's Itinerary

Introduction

What we know
- Community Structure of Industrial Problems
- Centrality of Industrial Instances
- Experimental Evidences

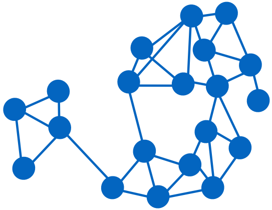
Community Structure and LBD

Conclusion

Community Structure

Central idea:

- Dense internal connections inside a group
- Sparser connections between groups



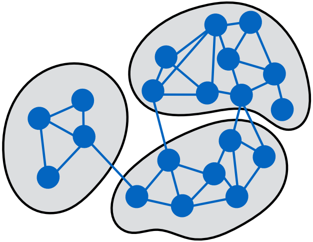
Work of [Ansótegui, Girádez-Cru, Levy'12]:

- **Industrial instances do have strong communities** (with high confidence)
- Learning does preserve them in many cases

Community Structure

Central idea:

- Dense internal connections inside a group
- Sparser connections between groups



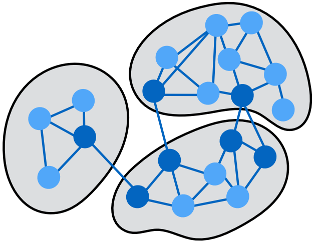
Work of [Ansótegui, Girádez-Cru, Levy'12]:

- **Industrial instances do have strong communities** (with high confidence)
- Learning does preserve them in many cases

Community Structure

Central idea:

- Dense internal connections inside a group
- Sparser connections between groups



Work of [Ansótegui, Girádez-Cru, Levy'12]:

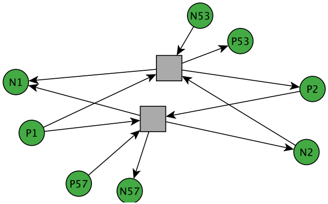
- **Industrial instances do have strong communities** (with high confidence)
- Learning does preserve them in many cases

The Direct Graphical Model

As done in [Katsirelos, Simon '12]

A bipartite, directed graph

- Nodes are literals and clauses
- Outgoing edge from Clause c to Literal l iff $l \in c$
- Outgoing edge from Literal l to Clause c iff $\neg l \in c$



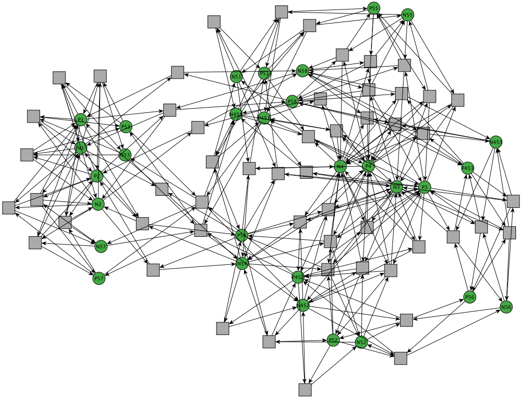
$$(\neg 1 \vee 53 \vee 2) \wedge (\neg 1 \vee \neg 2 \vee \neg 57)$$

- A conflict graph is a subgraph of the DGM
- A path in this graph has a also meaning: it tries to assign literals that satisfy clauses.

Eigenvector Centrality and CDCL

[Katsirelos, Simon '12] also studied the graphical representation of CNFs.

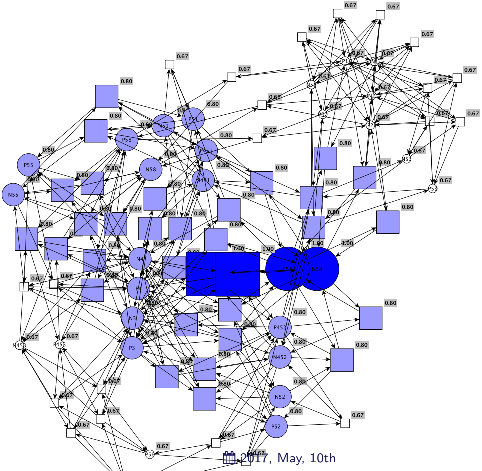
The (Eigenvector) centrality: "importance" of a node (see pagerank algorithm)



Eigenvector Centrality and CDCL

[Katsirelos, Simon '12] also studied the graphical representation of CNFs.

The (Eigenvector) centrality: "importance" of a node (see pagerank algorithm)



Centrality and communities

Work of [Katsirelos, Simon'12]:

- Measure the (initial) centrality of each variable
- Observe (during run time) solvers choices w.r.t. centrality
- Try to see if general tendencies can be observed

Relationship between centrality and communities

- Central nodes: the ones that you want to remove to partition the graph
- Central nodes: likely to be on the frontiere of clusters (communities)
- Non-Central nodes: inside clusters (communities)

Computing the centrality of huge graphs

The PageRank algorithm (Google) *An efficient iterative algorithm approximating the stationary distribution of a random walk on a graph*

Centrality of literals is computed once, in an off-line run.

Each analysis can take up to 20-30 min

On beeing central, or not

- On the web, central pages are the most important ones
- In a CNF, central nodes are likely to be on a fringe between clusters
- Decomposition can be performed by removing central nodes first

First Experimental Protocol

We want to detect some correlation between:

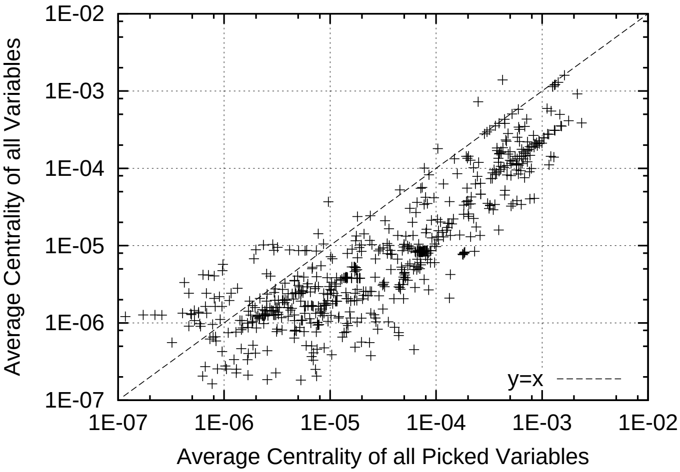
- The centrality (computed only once on the initial, preprocessed, formula)
- And observations/measures made during the CDCL run

The 2012 Protocol, at a glance

- We used glucose as an archetype CDCL solver
- We tested all 658 benchmarks from SatRace 2008, SatCompetion 2009 and 2011, in the Application category
- We fixed a cutoff of 5 Million conflicts, but placed no bound on CPU time

Picked Variables are central

Picture from [Katsirelos, Simon '12]



Learning unit clauses is essential

A typical run of a CDCL

- will "produce" top-level implied literals during the run
- but the frequency of produced literals will decrease

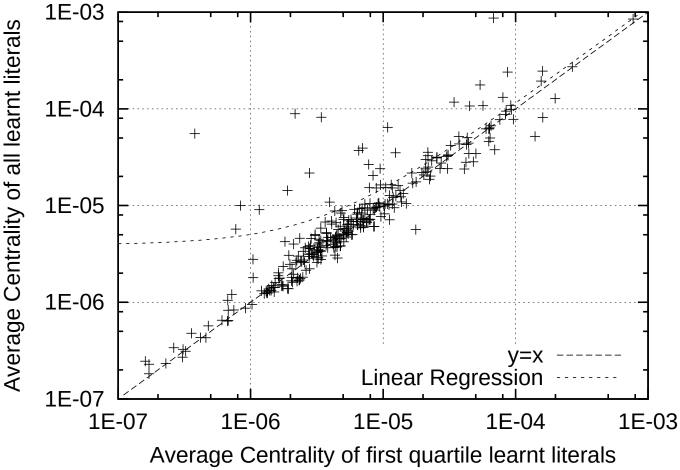
In practice

- A unit clause is learnt, and the literal is added at the top level
- This assignment will often produce immediatly other top-level propagated literals

Of course, on some problems, no unit-clause are learnt.

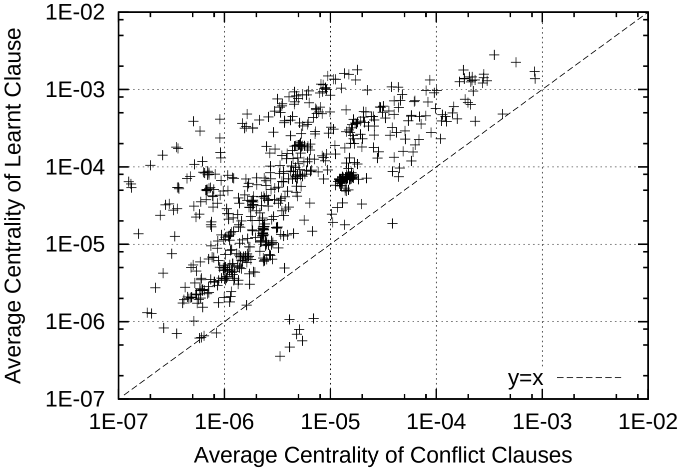
Measuring how the solver progresses

Picture from [Katsirelos, Simon '12]



Centrality of Conflict Clauses vs Learnt Clauses

Picture from [Katsirelos, Simon '12]



Today's Itinerary

Introduction

What we know

Community Structure and LBD

- Experiment parameters and Goals
- Experiments with Communities
- It's hard to study so many different problems

Conclusion

Making connections between LBD and communities

Initial Goal of the experiment:

“ Do we observe a relationship between LBD and communities? ”

How we built the experiment:

- We partitioned the variables into communities on the initial (preprocessed) formulas.
- We were able to compute the communities for 189 benchmarks of the SAT'13 competition
  (over 300 benchmarks, with 5000s time out)
- At each conflict, we observe the differences between the
  - The number of communities in the clause
  - The number of decision levels (LBD) in the clause

And. . . We cross our fingers!

A good example

Benchmarks
dated-5-13-u

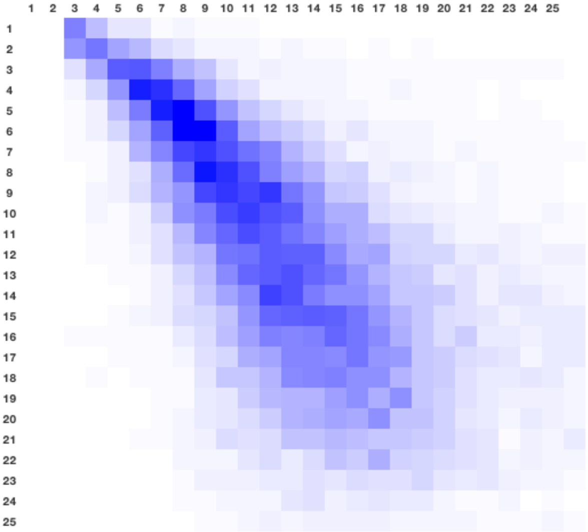
Nb Variables
138808

Nb Clauses
626501

Max conflicts
20000

Q
0.907721

Nb Partitions
97775



Another good example (there are many)

Benchmarks
gss-17-s100

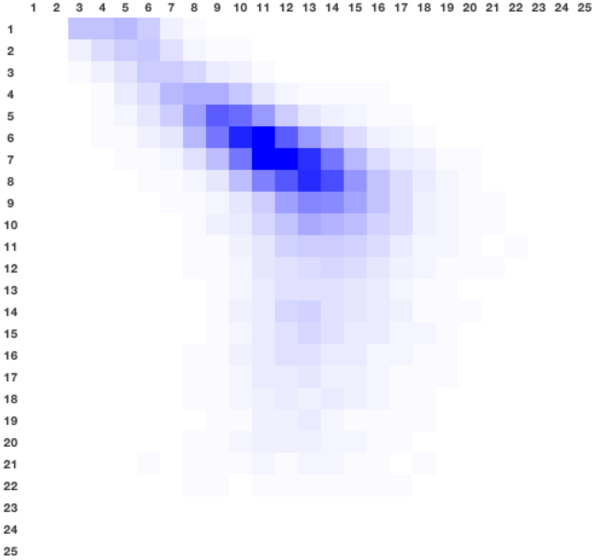
Nb Variables
31318

Nb Clauses
94116

Max conflicts
20000

Q
0.923977

Nb Partitions
17801



A bad example (there are some)

Benchmarks
rbcl_xits_14_SAT

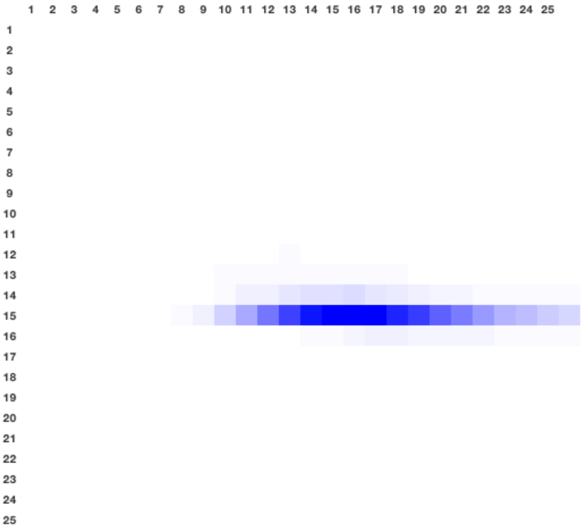
Nb Variables
2220

Nb Clauses
148488

Max conflicts
20000

Q
0.531783

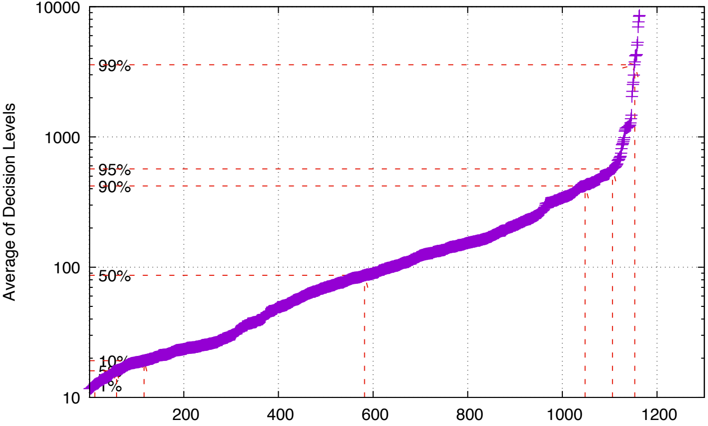
Nb Partitions
725



Outliers everywhere!

We try to understand CDCL solvers but all problems are distincts!

Number of decisions

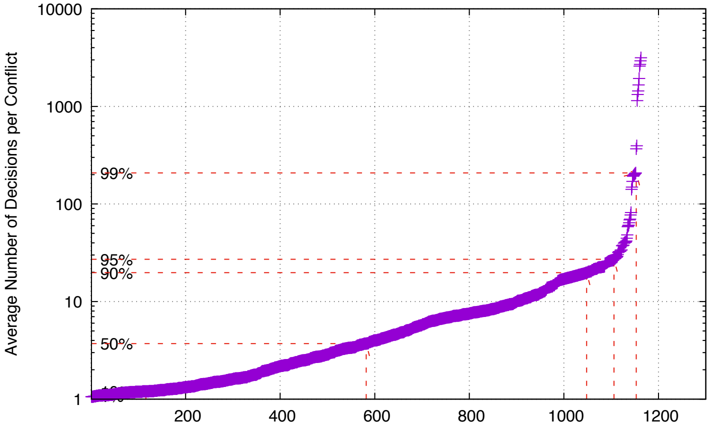


stats after 10,000 conflicts on 1164 “not easy” problems from all previous contests

Outliers everywhere!

We try to understand CDCL solvers but all problems are distincts!

Decisions per conflicts

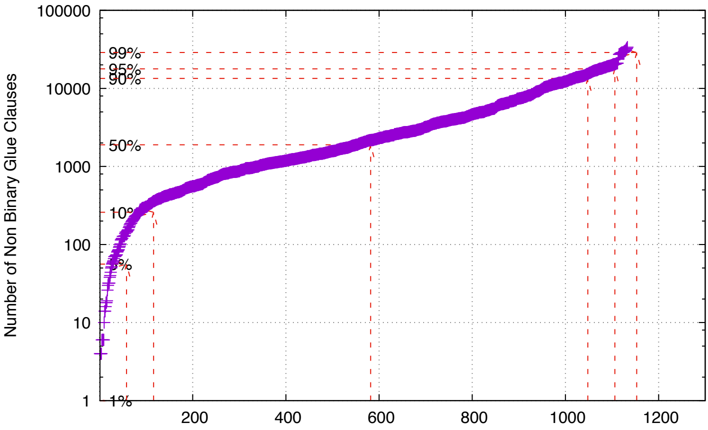


stats after 10,000 conflicts on 1164 “not easy” problems from all previous contests

Outliers everywhere!

We try to understand CDCL solvers but all problems are distincts!

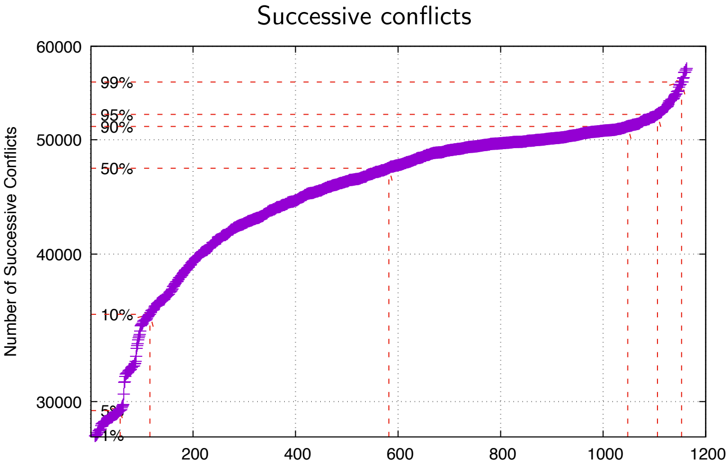
"True" (non binary) glue clauses



stats after 10,000 conflicts on 1164 "not easy" problems from all previous contests

Outliers everywhere!

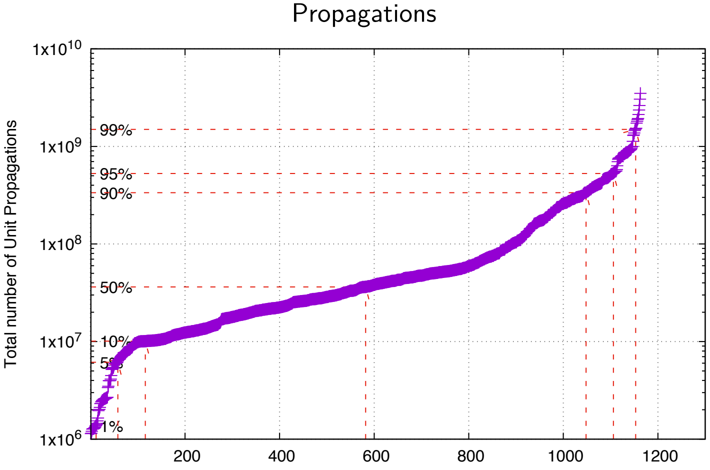
We try to understand CDCL solvers but all problems are distincts!



stats after 10,000 conflicts on 1164 "not easy" problems from all previous contests

Outliers everywhere!

We try to understand CDCL solvers but all problems are distincts!



stats after 10,000 conflicts on 1164 “not easy” problems from all previous contests

When experimenting suggests that CDCL solvers are inefficient

Most of solver's time is spent in unit propagation

But, on UNSAT instances:

- Only 50% of generated clauses are useful for deriving the final contradiction
- Only 20% of unit propagation are used during conflict analysis

Only 10% of solver's time is useful for deriving the contradiction!

Today's Itinerary

Introduction

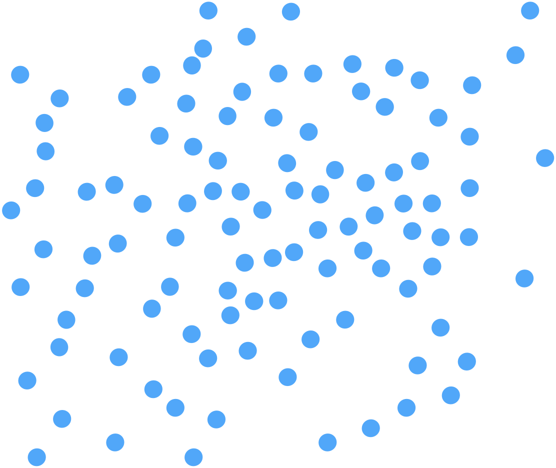
What we know

Community Structure and LBD

Conclusion

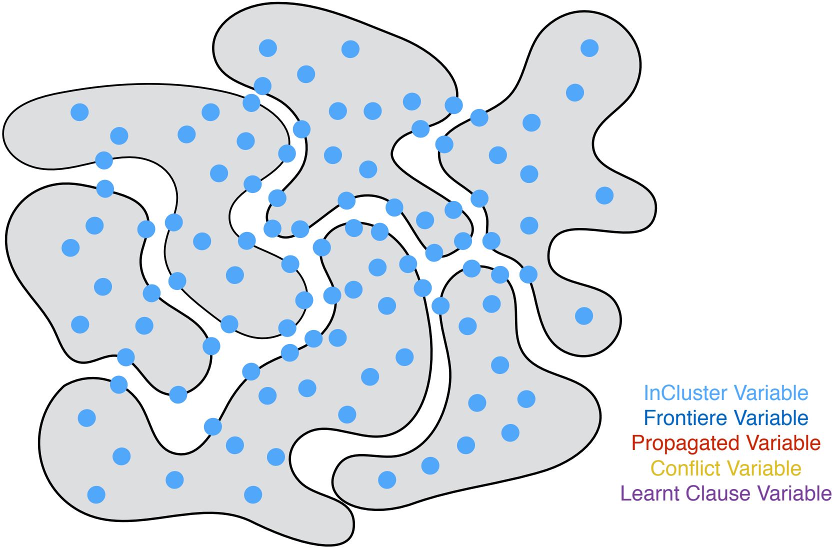
- A (Possible) Illustration of learnt clause mechanism
- Conclusion

(Possible) Illustration of learnt clause mechanism

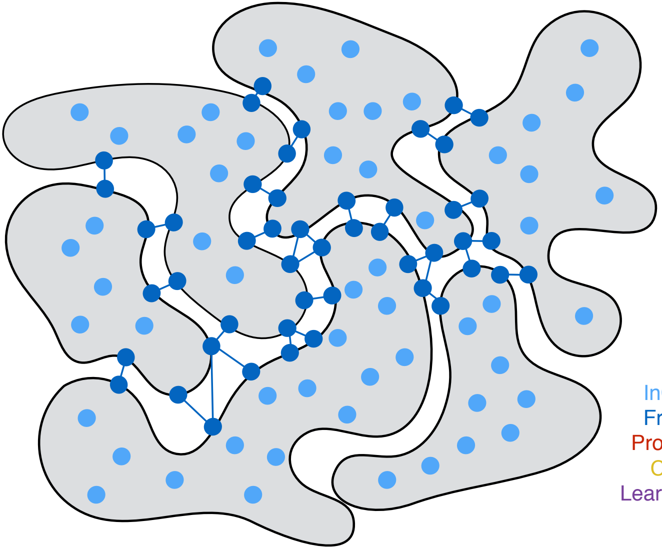


InCluster Variable
Frontiere Variable
Propagated Variable
Conflict Variable
Learnt Clause Variable

(Possible) Illustration of learnt clause mechanism

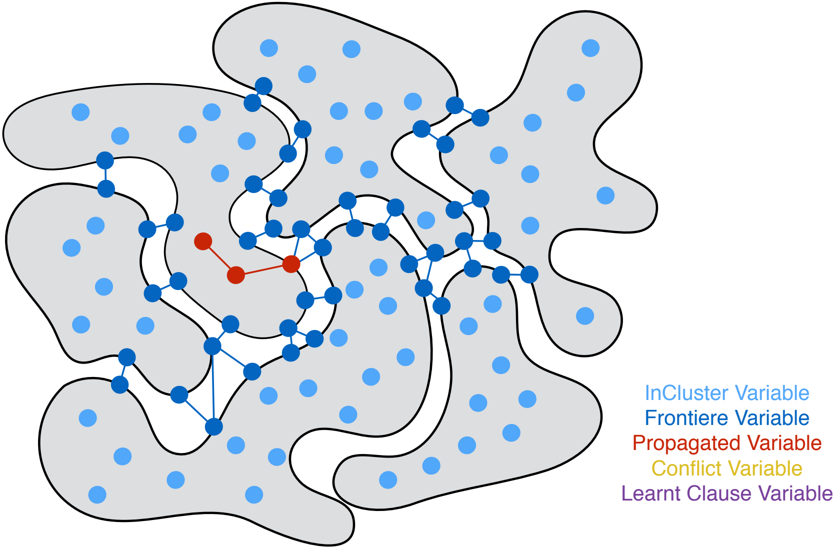


(Possible) Illustration of learnt clause mechanism

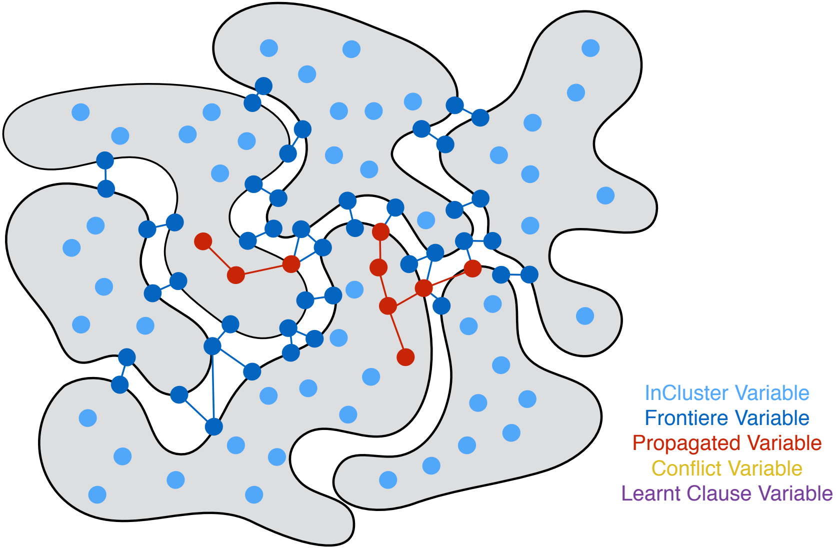


InCluster Variable
Frontiere Variable
Propagated Variable
Conflict Variable
Learnt Clause Variable

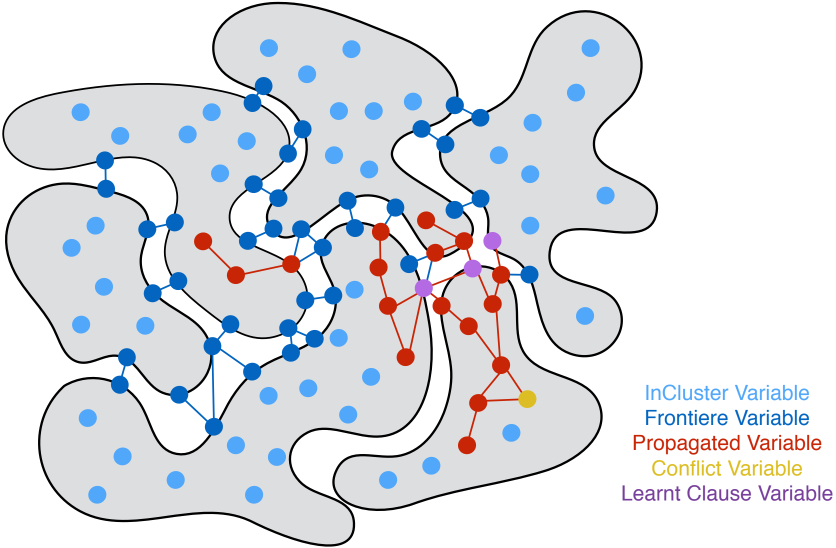
(Possible) Illustration of learnt clause mechanism



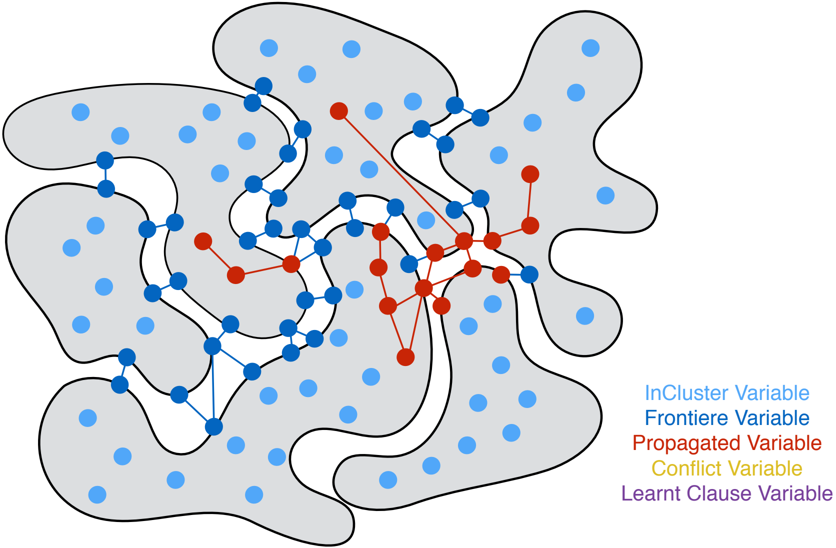
(Possible) Illustration of learnt clause mechanism



(Possible) Illustration of learnt clause mechanism



(Possible) Illustration of learnt clause mechanism



Conclusion – Deep Solving with SAT

SAT solvers are limited by Resolution

Experimenting is essential now in SAT research

Why are they so efficient?

- Beeing fast is beeing smart
- CDCL solvers are exploring and thinking at the same time
- They are not solving hard examples by applying methods we use on toy examples

We don't (fully) understand them, but we are working on it

Incremental SAT Solving is (also) an incredible driving force of the field