Predicting Learnt Clauses Quality in Modern SAT Solvers

Gilles Audemard

CRIL - Lens - France

Laurent Simon

LRI - Paris - France

SAT at a glance

Tremendeous progresses

- No more traditional backtracks and lookahead thinkings
- Unassigning variables is *free* and detecting new unit clauses is cheap (and lazy)
- The solver doesn't know where it is... But know a lot about its past activities: All components are lookback-oriented

Bad points:

We don't fully understand our 1000-lines of code

Good points:

The SAT community is one of the leading community in the *scientific* study of algorithms.

SAT COMPONENTS

ON *Applications* PROBLEMS

WATCHED LITERALS : Lazy detection of unit clauses / Backtracking is free

BLOCKED LITERALS : Handle memory bandwidth bottlenecks

FAST RESTARTS : Restart every 32 conflicts (!!)... Follows special laws (Luby series)

PHASE CACHING : Allow fast restarts to be efficient

LEARNING : Learn a new clause at each conflict (leaf of the tree). Forget useless clauses

DECAYING HEURISTIC : Branches on variables that were often and recently used in conflict analysis

If you don't do one of this points, you loose.

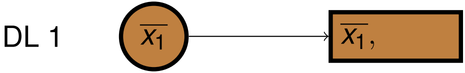
A short overview of a CDCL solver

decision – Propagation

$$\phi_1 = x_1 \vee x_4$$
$$\phi_2 = x_1 \vee \overline{x_3} \vee \overline{x_8}$$
$$\phi_3 = x_1 \vee x_8 \vee x_{12}$$
$$\phi_4 = x_2 \vee x_{11}$$
$$\phi_5 = \overline{x_3} \vee \overline{x_7} \vee x_{13}$$
$$\phi_6 = \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$$
$$\phi_7 = x_8 \vee \overline{x_7} \vee \overline{x_9}$$

A SHORT OVERVIEW OF A CDCL SOLVER

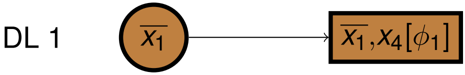
DECISION – PROPAGATION



$$\begin{aligned}
\phi_1 &= {\color{red}x_1} \vee x_4 \\
\phi_2 &= {\color{red}x_1} \vee \overline{x_3} \vee \overline{x_8} \\
\phi_3 &= {\color{red}x_1} \vee x_8 \vee x_{12} \\
\phi_4 &= x_2 \vee x_{11} \\
\phi_5 &= \overline{x_3} \vee \overline{x_7} \vee x_{13} \\
\phi_6 &= \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9 \\
\phi_7 &= x_8 \vee \overline{x_7} \vee \overline{x_9}
\end{aligned}$$

A SHORT OVERVIEW OF A CDCL SOLVER

DECISION – PROPAGATION



DL 1

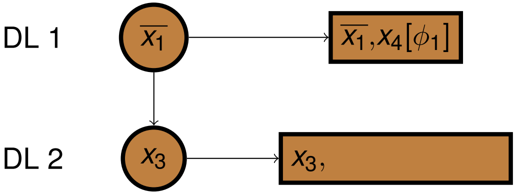
$\overline{x_1}$

$\overline{x_1}, x_4[\phi_1]$

$$\phi_1 = x_1 \vee x_4$$
$$\phi_2 = x_1 \vee \overline{x_3} \vee \overline{x_8}$$
$$\phi_3 = x_1 \vee x_8 \vee x_{12}$$
$$\phi_4 = x_2 \vee x_{11}$$
$$\phi_5 = \overline{x_3} \vee \overline{x_7} \vee x_{13}$$
$$\phi_6 = \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$$
$$\phi_7 = x_8 \vee \overline{x_7} \vee \overline{x_9}$$

A short overview of a CDCL solver

decision – Propagation



$$\phi_1 = x_1 \lor x_4$$
$$\phi_2 = x_1 \lor \overline{x_3} \lor \overline{x_8}$$
$$\phi_3 = x_1 \lor x_8 \lor x_{12}$$
$$\phi_4 = x_2 \lor x_{11}$$
$$\phi_5 = \overline{x_3} \lor \overline{x_7} \lor x_{13}$$
$$\phi_6 = \overline{x_3} \lor \overline{x_7} \lor \overline{x_{13}} \lor x_9$$
$$\phi_7 = x_8 \lor \overline{x_7} \lor \overline{x_9}$$

A short overview of a CDCL solver

decision – Propagation

$\phi_1 = x_1 \vee x_4$

$\phi_2 = x_1 \vee \overline{x_3} \vee \overline{x_8}$

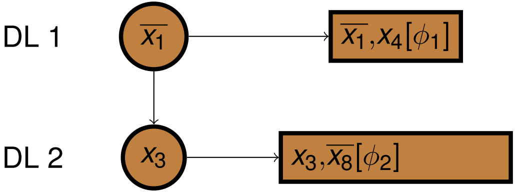
$\phi_3 = x_1 \vee x_8 \vee x_{12}$

$\phi_4 = x_2 \vee x_{11}$

$\phi_5 = \overline{x_3} \vee \overline{x_7} \vee x_{13}$

$\phi_6 = \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$

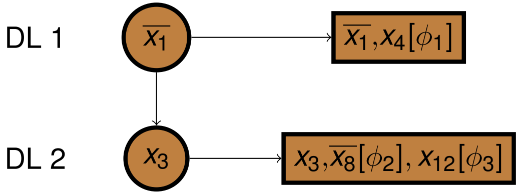
$\phi_7 = x_8 \vee \overline{x_7} \vee \overline{x_9}$



A SHORT OVERVIEW OF A CDCL SOLVER

DECISION – PROPAGATION

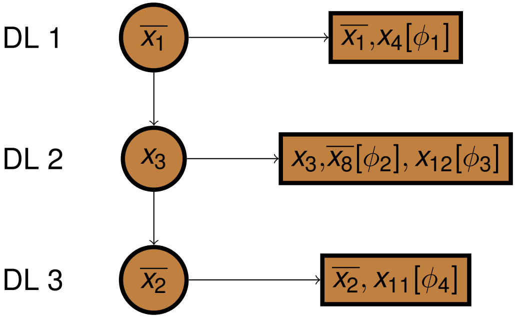
$$
\begin{aligned}
\phi_1 &= x_1 \vee x_4 \\
\phi_2 &= x_1 \vee \overline{x_3} \vee \overline{x_8} \\
\phi_3 &= x_1 \vee x_8 \vee x_{12} \\
\phi_4 &= x_2 \vee x_{11} \\
\phi_5 &= \overline{x_3} \vee \overline{x_7} \vee x_{13} \\
\phi_6 &= \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9 \\
\phi_7 &= x_8 \vee \overline{x_7} \vee \overline{x_9}
\end{aligned}
$$


DL 1
$\overline{x_1}$
$\overline{x_1}, x_4[\phi_1]$
DL 2
x_3
$x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3]$

A short overview of a CDCL solver

decision – Propagation

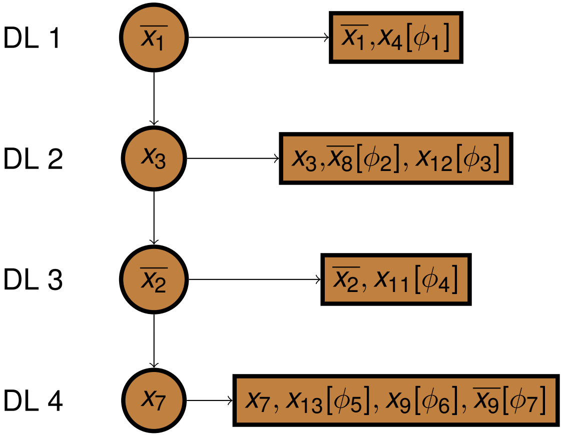
$$\phi_1 = x_1 \vee x_4$$
$$\phi_2 = x_1 \vee \overline{x_3} \vee \overline{x_8}$$
$$\phi_3 = x_1 \vee x_8 \vee x_{12}$$
$$\phi_4 = x_2 \vee x_{11}$$
$$\phi_5 = \overline{x_3} \vee \overline{x_7} \vee x_{13}$$
$$\phi_6 = \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$$
$$\phi_7 = x_8 \vee \overline{x_7} \vee \overline{x_9}$$


DL 1
$\overline{x_1}$
$\overline{x_1}, x_4[\phi_1]$
DL 2
x_3
$x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3]$
DL 3
$\overline{x_2}$
$\overline{x_2}, x_{11}[\phi_4]$

A short overview of a CDCL solver

decision – Propagation

$$\phi_1 = x_1 \lor x_4$$
$$\phi_2 = x_1 \lor \overline{x_3} \lor \overline{x_8}$$
$$\phi_3 = x_1 \lor x_8 \lor x_{12}$$
$$\phi_4 = x_2 \lor x_{11}$$
$$\phi_5 = \overline{x_3} \lor \overline{x_7} \lor x_{13}$$
$$\phi_6 = \overline{x_3} \lor \overline{x_7} \lor \overline{x_{13}} \lor x_9$$
$$\phi_7 = x_8 \lor \overline{x_7} \lor \overline{x_9}$$



DL 1 $\overline{x_1}$ $\overline{x_1}, x_4[\phi_1]$

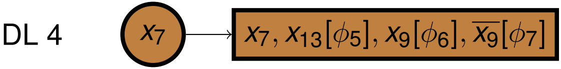
DL 2 x_3 $x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3]$

DL 3 $\overline{x_2}$ $\overline{x_2}, x_{11}[\phi_4]$

DL 4 x_7 $x_7, x_{13}[\phi_5], x_9[\phi_6], \overline{x_9}[\phi_7]$

A short overview of a CDCL solver

Conflict Analysis



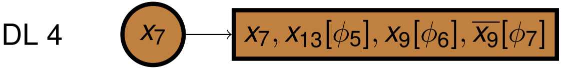
DL 4

x_7

$x_7, x_{13}[\phi_5], x_9[\phi_6], \overline{x_9}[\phi_7]$

A short overview of a CDCL solver

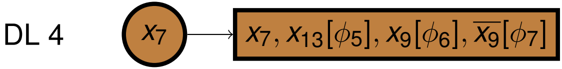
Conflict Analysis



$$\beta_1 = res(x_9, \phi_7, \phi_6) = \overline{x_3} \vee x_8 \vee \overline{x_7} \vee \overline{x_{13}}$$

A short overview of a CDCL solver

Conflict Analysis



DL 4

x_7

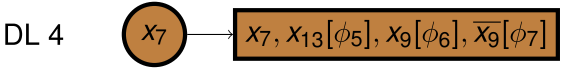
$x_7, x_{13}[\phi_5], x_9[\phi_6], \overline{x_9}[\phi_7]$

$$\beta_1 = res(x_9, \phi_7, \phi_6) = \overline{x_3} \vee x_8 \vee \overline{x_7} \vee \overline{x_{13}}$$

$$\beta = res(x_{13}, \beta_1, \phi_5) = \overline{x_3} \vee x_8 \vee \overline{x_7}$$

A short overview of a CDCL solver

Conflict Analysis



DL 4 $\xrightarrow{}$ x_7 $\rightarrow$ $\boxed{x_7, x_{13}[\phi_5], x_9[\phi_6], \overline{x_9}[\phi_7]}$

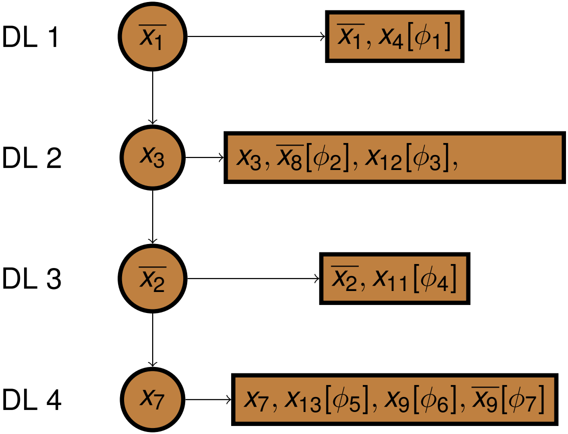
$$\beta_1 = res(x_9, \phi_7, \phi_6) = \overline{x_3} \vee x_8 \vee \overline{x_7} \vee \overline{x_{13}}$$
$$\beta = res(x_{13}, \beta_1, \phi_5) = \overline{x_3} \vee x_8 \vee \overline{x_7}$$

- First resolvent contains only one literal from the last decision level
- This is the "**First UIP**" scheme
- β is added to the clause database
- and ...

A short overview of a CDCL solver

Backjumping

$$
\begin{aligned}
\phi_1 &= x_1 \vee x_4 \\
\phi_2 &= x_1 \vee \overline{x_3} \vee \overline{x_8} \\
\phi_3 &= x_1 \vee x_8 \vee x_{12} \\
\phi_4 &= x_2 \vee x_{11} \\
\phi_5 &= \overline{x_3} \vee \overline{x_7} \vee x_{13} \\
\phi_6 &= \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9 \\
\phi_7 &= \overline{x_8} \vee \overline{x_7} \vee \overline{x_9}
\end{aligned}
$$



DL 1 $\overline{x_1}$ → $\overline{x_1}, x_4[\phi_1]$

DL 2 x_3 → $x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3],$

DL 3 $\overline{x_2}$ → $\overline{x_2}, x_{11}[\phi_4]$

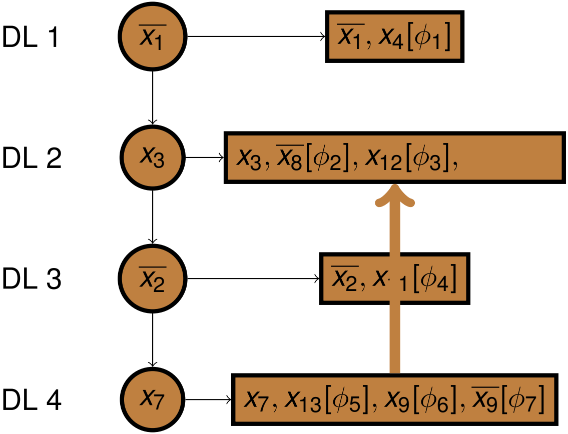
DL 4 x_7 → $x_7, x_{13}[\phi_5], x_9[\phi_6], \overline{x_9}[\phi_7]$

$$\beta = \overline{x_3} \vee x_8 \vee \overline{x_7}$$

A short overview of a CDCL solver

Backjumping

$$\begin{aligned}
\phi_1 &= x_1 \vee x_4\\
\phi_2 &= x_1 \vee \overline{x_3} \vee \overline{x_8}\\
\phi_3 &= x_1 \vee x_8 \vee x_{12}\\
\phi_4 &= x_2 \vee x_{11}\\
\phi_5 &= \overline{x_3} \vee \overline{x_7} \vee x_{13}\\
\phi_6 &= \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9\\
\phi_7 &= \overline{x_8} \vee \overline{x_7} \vee \overline{x_9}
\end{aligned}$$

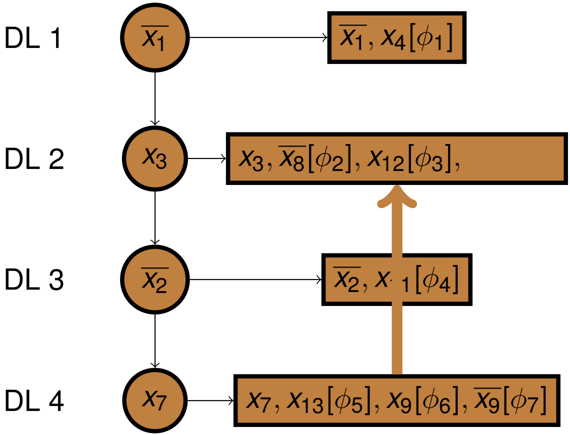

DL 1
$\overline{x_1}$
$\overline{x_1}, x_4[\phi_1]$
DL 2
x_3
$x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3],$
DL 3
$\overline{x_2}$
$\overline{x_2}, x_{11}[\phi_4]$
DL 4
x_7
$x_7, x_{13}[\phi_5], x_9[\phi_6], \overline{x_9}[\phi_7]$

$$\beta = \overline{x_3} \vee x_8 \vee \overline{x_7}$$

A short overview of a CDCL solver

Backjumping

$$\begin{aligned}
\phi_1 &= x_1 \vee x_4\\
\phi_2 &= x_1 \vee \overline{x_3} \vee \overline{x_8}\\
\phi_3 &= x_1 \vee x_8 \vee x_{12}\\
\phi_4 &= x_2 \vee x_{11}\\
\phi_5 &= \overline{x_3} \vee \overline{x_7} \vee x_{13}\\
\phi_6 &= \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9\\
\phi_7 &= \overline{x_8} \vee \overline{x_7} \vee \overline{x_9}
\end{aligned}$$

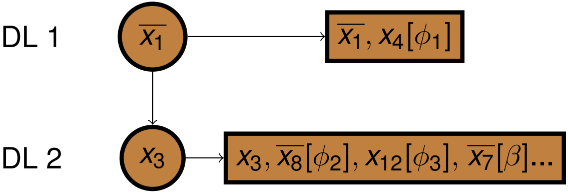

DL 1
$\overline{x_1}$
$\overline{x_1}, x_4[\phi_1]$
DL 2
x_3
$x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3],$
DL 3
$\overline{x_2}$
$\overline{x_2}, x_{11}[\phi_4]$
DL 4
x_7
$x_7, x_{13}[\phi_5], x_9[\phi_6], \overline{x_9}[\phi_7]$

$$\beta = \overline{x_3} \vee x_8 \vee \overline{x_7}$$

A SHORT OVERVIEW OF A CDCL SOLVER

BACKJUMPING

$$\phi_1 = x_1 \vee x_4$$
$$\phi_2 = x_1 \vee \overline{x_3} \vee \overline{x_8}$$
$$\phi_3 = x_1 \vee x_8 \vee x_{12}$$
$$\phi_4 = x_2 \vee x_{11}$$
$$\phi_5 = \overline{x_3} \vee \overline{x_7} \vee x_{13}$$
$$\phi_6 = \overline{x_3} \vee \overline{x_7} \vee \overline{x_{13}} \vee x_9$$
$$\phi_7 = \overline{x_8} \vee \overline{x_7} \vee \overline{x_9}$$



DL 1

$\overline{x_1}$

$\overline{x_1}, x_4[\phi_1]$

DL 2

x_3

$x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3], \overline{x_7}[\beta]$...

$$\beta = \overline{x_3} \vee x_8 \vee \overline{x_7}$$

Other components and Motivations

- CDCL solvers contain a lot of additional features and tricks
- heuristic
  - Dynamic: Award variables used during recent conflicts analyses
  - Progress saving
- Restarts : Static or Dynamic

- A hidden component: learnt clauses cleaning
  - To avoid memory blow up, one needs to remove some of the learnt clauses
  - Which ones ?
  - Currently: good clauses are supposed to be the reasons of unit propagation seen during recent conflicts analyses (follows the success of the VSIDS idea)

identifing good clauses during search

OUTLINE

- An empirical observation
- Identifying good clauses
- An agressive strategy to clean learnt clauses
- Experiments

An Observation

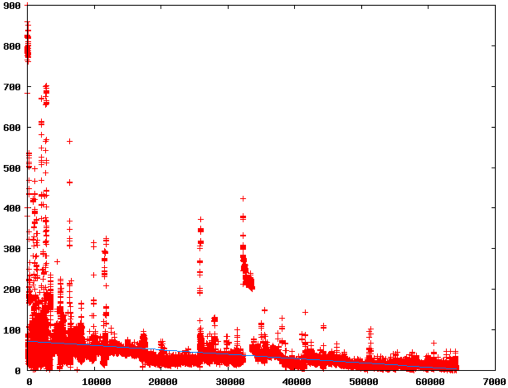
- Before CDCL solvers: Solvers implement ideas (look-ahead, Mom's heuristics...)
  explaining performances was simple

- With CDCL: Look-back solvers (VSIDS heurisitics...)
  explaining performances is hard

We need strong empirical studies in order to understand and improve performances

SOME PLOTS ...

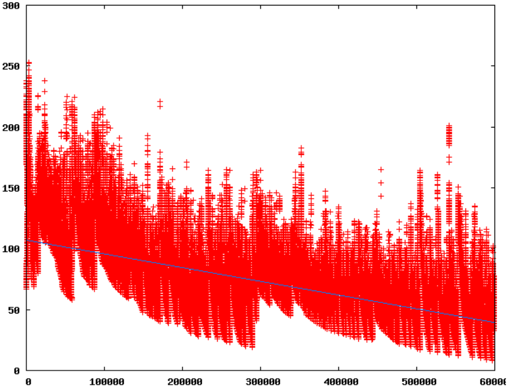
EEN-PICO-PROP05-50 – UNSAT – 13,000 VARS AND 65,000 CLAUSES



- For each conflict, we store the decision level where it occurs
- We also compute the linear regression on these points
- Gives an idea of the global behavior of the computation

SOME PLOTS . . .

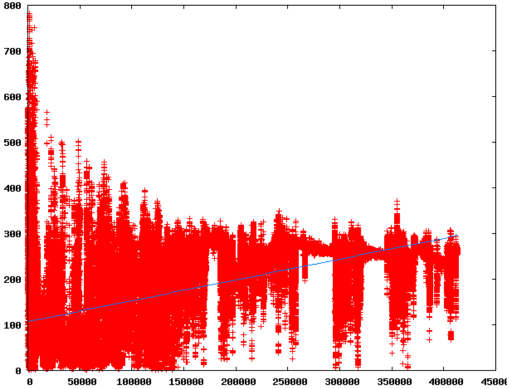
GRIEU-VMPC-S05-25 – SAT – 625 VARS AND 76,000 CLAUSES



- For each conflict, we store the decision level where it occurs
- We also compute the linear regression on these points
- Gives an idea of the global behavior of the computation

SOME PLOTS . . .

MIZH-SHA0-35-3 – SAT – 20,000 VARS AND 120,000 CLAUSES



- For each conflict, we store the decision level where it occurs
- We also compute the linear regression on these points
- Gives an idea of the global behavior of the computation

REMARKS

- Of course, we do not expect to feet curves
- We try to make observations of the behaviour of a CDCL solver

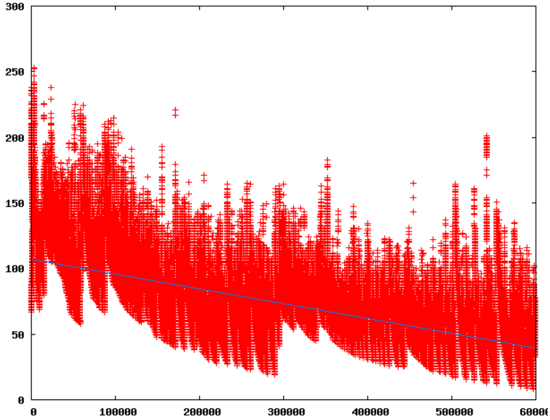
AND...

Decreasing appears in a lot of problems

Series	#Benchs	% Decr.
een	8	62%
goldb	11	100%
grieu	7	71%
hoons	5	100%
ibm-2002	7	71%
ibm-2004	13	92%
manol-pipe	55	91%
miz	13	0%
schup	5	80%
simon	10	90%
vange	3	66%
velev	54	92%
all	199	83%

Observe and predict

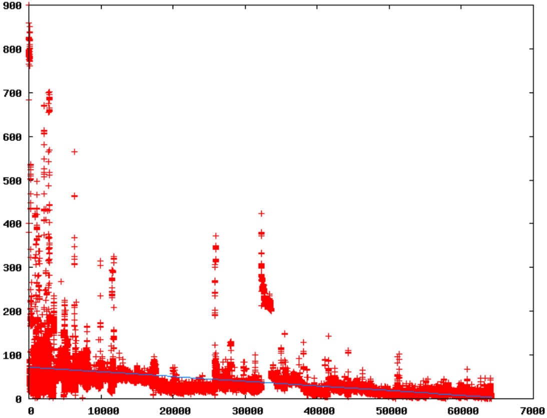
Where the solver does find a solution?



- Intersection of the linear regression with X-axis
- Lookback justification : one needs to do the search to compute this point (no prediction)

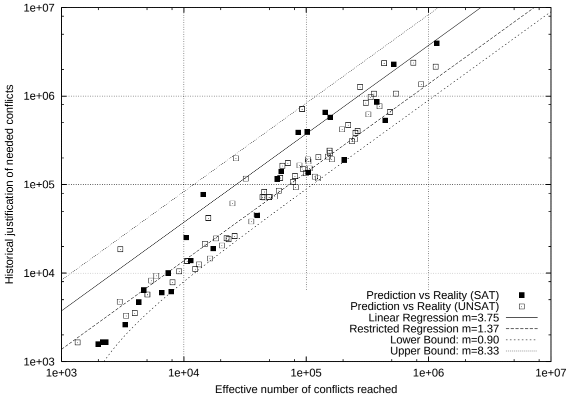
Observe and predict

Where the solver does find a solution?



- Intersection of the linear regression with X-axis
- Lookback justification : one needs to do the search to compute this point (no prediction)

Relationship



- A strong relationship between lookback justification and effective number of conflicts
- No distinction between SAT and UNSAT instances

SUMMARY

- Decision levels decrease along the search

- Relationship between lookback justification and effective number of conflicts

- Enforce the decreasing of decision levels will help to
  - Speed up the search
  - Protect learnt clauses that play this role

INTUITIONS

- A lot of dependencies between variables
  During search those variables will probably be propagated together inside blocks of propagations

- One needs to collapse independant blocks of propagated literals in order to reduce the decision level

A Static measure

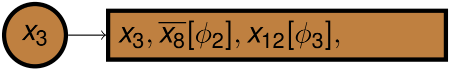
Literal Block distance – LBD

The LBD score of a nogood is the number of different blocks of propagated literals

- This measure is computed only one time (at the construction of the clause)
- Good clauses should have a small LBD !!!

A static measure

$$\beta = \overline{x_3} \vee x_8 \vee \overline{x_7}$$



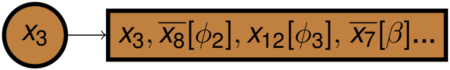
x_3

$x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3],$

- LBD==2
  - Only one literal from the last decision level (the assertive one)
  - This literal will be **glued** to the other block
  - binary clauses have LBD equal to 2
- VSIDS + progress saving: this should occurs a lot!!!

A STATIC MEASURE

$$\beta = \overline{x_3} \vee x_8 \vee \overline{x_7}$$



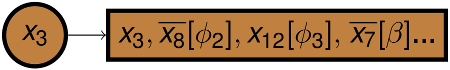
x_3

$x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3], \overline{x_7}[\beta]$...

- LBD==2
  - Only one literal from the last decision level (the assertive one)
  - This literal will be **glued** to the other block
  - binary clauses have LBD equal to 2
- VSIDS + progress saving: this should occurs a lot!!!

A STATIC MEASURE

$$\beta = \overline{x_3} \vee x_8 \vee \overline{x_7}$$



$x_3 \longrightarrow x_3, \overline{x_8}[\phi_2], x_{12}[\phi_3], \overline{x_7}[\beta]$...

- LBD==2
  - Only one literal from the last decision level (the assertive one)
  - This literal will be **glued** to the other block
  - binary clauses have LBD equal to 2
- VSIDS + progress saving: this should occurs a lot!!!

Good clauses are GLUE clauses

Introduction

- Solvers performances are tightly related to their clauses database managment
    - Keeping too many clauses will decrease BCP performances
    - Deleting too many clauses will break the learning benefit
- Currently, good learnt clauses are related to their recent usefulness in conflict analysis
- It is not a very good measure, so the number of learnt clauses follows a geometric progression

Use static LBD measure

AGRESSIVE STRATEGY

- Only LBD and size are used to identify good learnt clauses
  - Short LBD are good ones
  - In case of equality, prefer short clauses
- Remove half of learnt clauses every $20000 + 500 \times x$
- No matter the size of the initial formula

Comparison of 4 versions of Minisat

- ag : Agressive deletion strategy (instead of the classical one)
- lbd : static measure (instead of the dynamic one)
- 100 benchmarks from SAT-Race 2006
- Timeout : 1000 seconds

	#N (sat-unsat)
Minisat	70 (35 – 35)

COMPARISON OF 4 VERSIONS OF MINISAT

- ag : Agressive deletion strategy (instead of the classical one)
- lbd : static measure (instead of the dynamic one)
- 100 benchmarks from SAT-Race 2006
- Timeout : 1000 seconds

	#N (sat-unsat)
MINISAT	70 (35 – 35)
MINISAT +ag	74 (41 – 33)

Comparison of 4 versions of Minisat

- ag : Agressive deletion strategy (instead of the classical one)
- lbd : static measure (instead of the dynamic one)
- 100 benchmarks from SAT-Race 2006
- Timeout : 1000 seconds

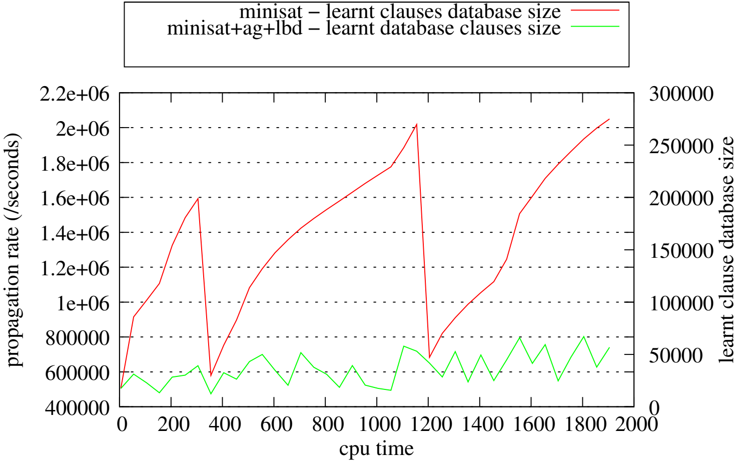
	#N (sat-unsat)
Minisat	70 (35 – 35)
Minisat +ag	74 (41 – 33)
Minisat +lbd	79 (**47** – 32)

Comparison of 4 versions of Minisat

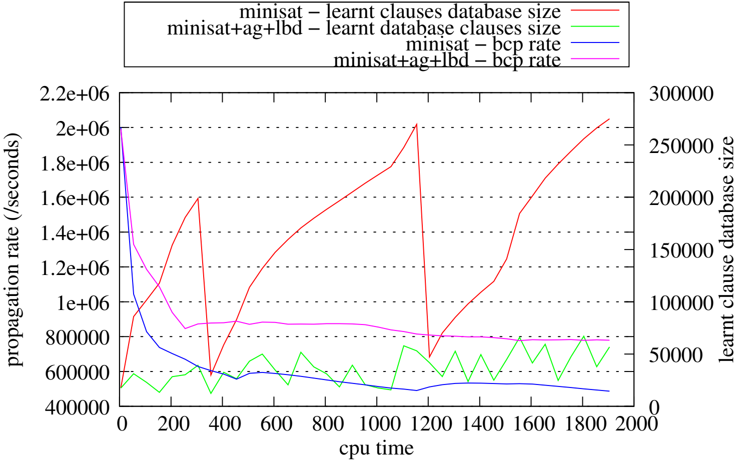
- ag : Agressive deletion strategy (instead of the classical one)
- lbd : static measure (instead of the dynamic one)
- 100 benchmarks from SAT-Race 2006
- Timeout : 1000 seconds

	#N (sat-unsat)
Minisat	70 (35 – 35)
Minisat +ag	74 (41 – 33)
Minisat +lbd	79 (**47** – 32)
Minisat +ag+lbd	**82** (45 – **37**)

JUSTIFICATION



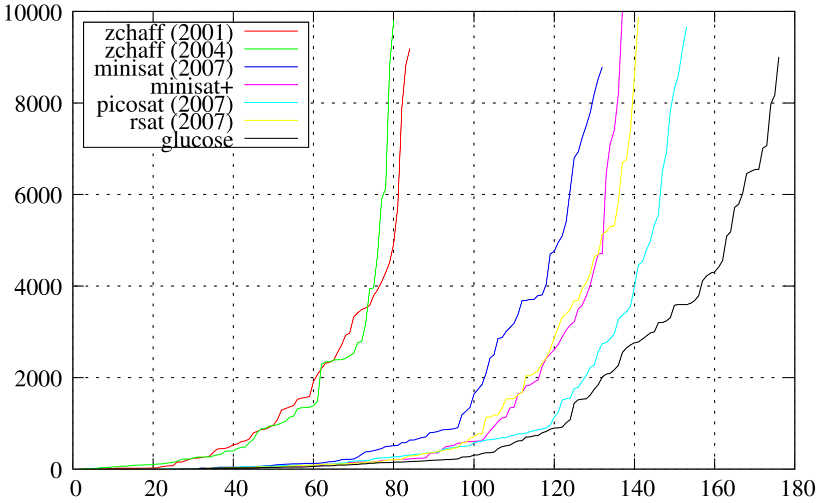
JUSTIFICATION



INTRODUCTION

- 7 SOTA solvers :
  - ZCHAFF: 2001 – 2004
  - RSAT: 2007
  - MINISAT: 2007 – 2008 (luby restart, progress saving)
  - PICOSAT: 2008
  - Our solver : GLUCOSE, based on MINISAT (luby restart 32, progress saving)
- 234 Industrial instances from SAT competition 2007
- timeout : 10,000 seconds
- All instances are pre-processed with SatELite

CACTUS PLOT


10000
8000
6000
4000
2000
0
0
20
40
60
80
100
120
140
160
180
zchaff (2001)
zchaff (2004)
minisat (2007)
minisat+
picosat (2007)
rsat (2007)
glucose

Some details

- #N : number of solved instances
- #U : unique solver to solve an instance
- #F : fast answer
- #S : speed on same subset of solved instances

solver	#N	(SAT-UNSAT)	#U	#F	#S
zChaff 01	84	(47 – 37)	0	13	2.9
zChaff 04	80	(39 – 41)	0	5	3.9
Minisat	132	(53 – 79)	1	16	2.1
Minisat+	136	(66 – 74)	0	15	1.5
Rsat	139	(63 – 75)	1	14	1.7
Picosat	153	(**75** – 78)	1	26	1.2
Glucose	**176**	(**75** – **101**)	**22**	**68**	-

SAT Competition 2009 - Introduction

- Enhanced version of GLUCOSE participated to the SAT competition 2009
  - Dynamic restart strategy that enhance decreasing
  - Some data-structures hacks : blocked literals,binary clauses
  - available at `http://www.lri.fr/~simon/glucose`
- 292 instances in application category
- 50 solvers submitted

SAT COMPETITION 2009 - RESULTS

- UNSAT problems :
  GOLD MEDAL : GLUCOSE: 127 instances solved
- SAT+UNSAT problems
  GOLD MEDAL : precosat : 204 instances solved in 153,127s
  SILVER MEDAL : GLUCOSE: 204 instances solved in 180,345s

CONCLUSION – PERSPECTIVES

- A static measure of good learnt clauses
- An agressive clauses deletion strategy
- An efficient solver GLUCOSE

- Other measures are needed
- Improve performances on SAT problems
- Continue empirical study

Thanks for your attention