Mini-tutorial on conflict-driven clause learning solvers

Marijn J. H. Heule

The University of Texas at Austin

20/01/2014 @ BIRS
The Satisfiability (SAT) problem

\[(x_5 \lor x_8 \lor \overline{x}_2) \land (x_2 \lor \overline{x}_1 \lor \overline{x}_3) \land (x_8 \lor \overline{x}_3 \lor \overline{x}_7) \land (\overline{x}_5 \lor \overline{x}_3 \lor x_8) \land \]
\[(\overline{x}_6 \lor \overline{x}_1 \lor \overline{x}_5) \land (x_8 \lor \overline{x}_9 \lor x_3) \land (x_2 \lor x_1 \lor x_3) \land (\overline{x}_1 \lor x_8 \lor x_4) \land \]
\[(\overline{x}_9 \lor \overline{x}_6 \lor x_8) \land (x_8 \lor x_3 \lor \overline{x}_9) \land (x_9 \lor \overline{x}_3 \lor x_8) \land (x_6 \lor \overline{x}_9 \lor x_5) \land \]
\[(x_2 \lor \overline{x}_3 \lor \overline{x}_8) \land (x_8 \lor \overline{x}_6 \lor \overline{x}_3) \land (x_8 \lor \overline{x}_3 \lor \overline{x}_1) \land (\overline{x}_8 \lor \overline{x}_6 \lor \overline{x}_2) \land \]
\[(x_7 \lor x_9 \lor \overline{x}_2) \land (x_8 \lor \overline{x}_9 \lor x_2) \land (\overline{x}_1 \lor \overline{x}_9 \lor x_4) \land (x_8 \lor x_1 \lor \overline{x}_2) \land \]
\[(x_3 \lor \overline{x}_4 \lor \overline{x}_6) \land (\overline{x}_1 \lor \overline{x}_7 \lor \overline{x}_5) \land (\overline{x}_7 \lor x_1 \lor x_6) \land (\overline{x}_5 \lor x_4 \lor \overline{x}_6) \land \]
\[(\overline{x}_4 \lor x_9 \lor x_8) \land (x_2 \lor x_9 \lor x_1) \land (x_5 \lor \overline{x}_7 \lor x_1) \land (\overline{x}_7 \lor \overline{x}_9 \lor \overline{x}_6) \land \]
\[(x_2 \lor x_5 \lor x_4) \land (x_8 \lor \overline{x}_4 \lor x_5) \land (x_5 \lor x_9 \lor x_3) \land (\overline{x}_5 \lor \overline{x}_7 \lor x_9) \land \]
\[(x_2 \lor \overline{x}_8 \lor x_1) \land (\overline{x}_7 \lor x_1 \lor x_5) \land (x_1 \lor x_4 \lor x_3) \land (x_1 \lor \overline{x}_9 \lor \overline{x}_4) \land \]
\[(x_3 \lor x_5 \lor x_6) \land (\overline{x}_6 \lor x_3 \lor \overline{x}_9) \land (\overline{x}_7 \lor x_5 \lor x_9) \land (x_7 \lor \overline{x}_5 \lor \overline{x}_2) \land \]
\[(x_4 \lor x_7 \lor x_3) \land (x_4 \lor \overline{x}_9 \lor \overline{x}_7) \land (x_5 \lor \overline{x}_1 \lor x_7) \land (x_5 \lor \overline{x}_1 \lor x_7) \land \]
\[(x_6 \lor x_7 \lor \overline{x}_3) \land (\overline{x}_8 \lor \overline{x}_6 \lor \overline{x}_7) \land (x_6 \lor x_2 \lor x_3) \land (\overline{x}_8 \lor x_2 \lor x_5) \land \]

Does there exist an assignment satisfying all clauses?
Search for a satisfying assignment (or proof none exists)

\[
\begin{align*}
(x_5 \lor x_8 \lor \overline{x}_2) & \land (x_2 \lor \overline{x}_1 \lor \overline{x}_3) & \land (\overline{x}_8 \lor \overline{x}_3 \lor \overline{x}_7) & \land (\overline{x}_5 \lor x_3 \lor x_8) \\
(\overline{x}_6 \lor \overline{x}_1 \lor \overline{x}_5) & \land (x_8 \lor \overline{x}_9 \lor x_3) & \land (x_2 \lor x_1 \lor x_3) & \land (\overline{x}_1 \lor x_8 \lor x_4) \\
(\overline{x}_9 \lor \overline{x}_6 \lor x_8) & \land (x_8 \lor x_3 \lor \overline{x}_9) & \land (x_9 \lor \overline{x}_3 \lor x_8) & \land (x_6 \lor \overline{x}_9 \lor x_5) \\
(x_2 \lor \overline{x}_3 \lor \overline{x}_8) & \land (x_8 \lor \overline{x}_6 \lor \overline{x}_3) & \land (x_8 \lor \overline{x}_3 \lor \overline{x}_1) & \land (\overline{x}_8 \lor x_6 \lor \overline{x}_2) \\
(x_7 \lor x_9 \lor \overline{x}_2) & \land (x_8 \lor \overline{x}_9 \lor x_2) & \land (\overline{x}_1 \lor \overline{x}_9 \lor x_4) & \land (x_8 \lor x_1 \lor \overline{x}_2) \\
(x_3 \lor \overline{x}_4 \lor \overline{x}_6) & \land (\overline{x}_1 \lor \overline{x}_7 \lor x_5) & \land (\overline{x}_7 \lor x_1 \lor x_6) & \land (\overline{x}_5 \lor x_4 \lor \overline{x}_6) \\
(\overline{x}_4 \lor x_9 \lor \overline{x}_8) & \land (x_2 \lor x_9 \lor x_1) & \land (x_5 \lor \overline{x}_7 \lor x_1) & \land (\overline{x}_7 \lor \overline{x}_9 \lor \overline{x}_6) \\
(x_2 \lor x_5 \lor x_4) & \land (x_8 \lor \overline{x}_4 \lor x_5) & \land (x_5 \lor \overline{x}_9 \lor x_3) & \land (\overline{x}_5 \lor \overline{x}_7 \lor x_9) \\
(x_2 \lor \overline{x}_8 \lor x_1) & \land (\overline{x}_7 \lor x_1 \lor x_5) & \land (x_1 \lor x_4 \lor x_3) & \land (x_1 \lor \overline{x}_9 \lor \overline{x}_4) \\
(x_3 \lor x_5 \lor x_6) & \land (\overline{x}_6 \lor x_3 \lor \overline{x}_9) & \land (\overline{x}_7 \lor x_5 \lor \overline{x}_9) & \land (x_7 \lor \overline{x}_5 \lor \overline{x}_2) \\
(x_4 \lor x_7 \lor x_3) & \land (x_4 \lor \overline{x}_9 \lor \overline{x}_7) & \land (x_5 \lor \overline{x}_1 \lor x_7) & \land (x_5 \lor \overline{x}_1 \lor x_7) \\
(x_6 \lor x_7 \lor \overline{x}_3) & \land (\overline{x}_8 \lor \overline{x}_6 \lor \overline{x}_7) & \land (x_6 \lor x_2 \lor x_3) & \land (\overline{x}_8 \lor x_2 \lor x_5)
\end{align*}
\]
\((x_1 \lor x_4) \land (x_3 \lor \bar{x}_4 \lor \bar{x}_5) \land (\bar{x}_3 \lor \bar{x}_2 \lor \bar{x}_4) \land F_{\text{extra}}\)
\[(x_1 \lor x_4) \land (x_3 \lor \overline{x_4} \lor \overline{x_5}) \land (\overline{x_3} \lor \overline{x_2} \lor \overline{x_4}) \land F_{\text{extra}}\]

\[x_5 = 1\]
\((x_1 \lor x_4) \land (x_3 \lor \bar{x}_4 \lor \bar{x}_5) \land (\bar{x}_3 \lor \bar{x}_2 \lor \bar{x}_4) \land F_{\text{extra}}\)
\[
(x_1 \lor x_4) \land
(x_3 \lor \overline{x_4} \lor \overline{x_5}) \land
(\overline{x_3} \lor \overline{x_2} \lor \overline{x_4}) \land
F_{\text{extra}}
\]
\[(x_1 \lor x_4) \land (x_3 \lor \overline{x_4} \lor \overline{x_5}) \land (\overline{x_3} \lor \overline{x_2} \lor \overline{x_4}) \land F_{\text{extra}}\]
\((x_1 \lor x_4) \land (x_3 \lor \bar{x}_4 \lor \bar{x}_5) \land (\bar{x}_3 \lor \bar{x}_2 \lor \bar{x}_4) \land F_{\text{extra}}\)
Conflict-driven SAT solvers: Search and Analysis

\[
\begin{align*}
(x_1 \lor x_4) \land \\
(x_3 \lor \overline{x}_4 \lor \overline{x}_5) \land \\
(\overline{x}_3 \lor x_2 \lor \overline{x}_4) \land \\
F_{\text{extra}}
\end{align*}
\]

Marijn J. H. Heule (UT)
\[(x_1 \lor x_4) \land (x_3 \lor \overline{x}_4 \lor \overline{x}_5) \land (\overline{x}_3 \lor \overline{x}_2 \lor \overline{x}_4) \land F_{\text{extra}}\]

\[
\begin{array}{c}
\text{7} \\
\text{1} \\
\text{0} \\
\text{2} \\
\text{6} \\
\text{7}
\end{array}
\]

\[
\begin{array}{c}
\text{x}_1 = 0 \\
\text{x}_4 = 1 \\
\text{x}_2 = 1 \\
\text{x}_3 = 0 \\
\text{x}_5 = 1 \\
\text{x}_4 = 1
\end{array}
\]

Marijn J. H. Heule (UT)  
Mini-tutorial on CDCL solvers  
BIRS, January 2014
\[(x_1 \lor x_4) \land (x_3 \lor \bar{x}_4 \lor \bar{x}_5) \land (\bar{x}_3 \lor \bar{x}_2 \lor \bar{x}_4) \land F_{\text{extra}}\]

\[
\begin{align*}
\bar{x}_2 \lor \bar{x}_4 \lor \bar{x}_5 \\
\end{align*}
\]

\[
\begin{align*}
0 \\
1 \\
2 \\
6 \\
7
\end{align*}
\]

\[
\begin{align*}
0 = 1 \\
1 = 1 \\
2 = 1 \\
6 = 1 \\
7 = 1
\end{align*}
\]

\[
\begin{align*}
x_1 = 0 & \quad x_4 = 1 \\
x_5 = 1 & \\
x_2 = 1 & \quad x_3 = 0 \\
x_1 = 0 & \\
x_4 = 1 & \\
x_3 = 1 & \\
x_3 = 0 &
\end{align*}
\]
Conflict-driven SAT solvers: Search and Analysis

\[(x_1 \lor x_4) \land (x_3 \lor \overline{x}_4 \lor \overline{x}_5) \land (\overline{x}_3 \lor \overline{x}_2 \lor \overline{x}_4) \land F_{\text{extra}}\]

\[
\begin{array}{c}
\text{x}_1 = 0 \\
\text{x}_2 = 1 \\
\text{x}_3 = 0 \\
\text{x}_4 = 1 \\
\text{x}_5 = 1
\end{array}
\]

\[
(\overline{x}_2 \lor \overline{x}_4 \lor \overline{x}_5)
\]
\((x_1 \lor x_4) \land (x_3 \lor \neg x_4 \lor \neg x_5) \land (\neg x_3 \lor \neg x_2 \lor \neg x_4) \land F_{\text{extra}}\)
\[(x_1 \lor x_4) \land (x_3 \lor \bar{x}_4 \lor \bar{x}_5) \land (\bar{x}_3 \lor \bar{x}_2 \lor \bar{x}_4) \land \mathcal{F}_{\text{extra}}\]
Conflict-driven SAT solvers: Pseudo-code

1: while TRUE do
2: \( l_{\text{decision}} := \text{GET\_DECISION\_LITERAL}( ) \)
3: If no \( l_{\text{decision}} \) then return satisfiable
4: \( \mathcal{F} := \text{SIMPLIFY}( \mathcal{F}( l_{\text{decision}} \leftarrow 1 ) ) \)
5: while \( \mathcal{F} \) contains \( C_{\text{falsified}} \) do
6: \( C_{\text{conflict}} := \text{ANALYZE\_CONFLICT}( C_{\text{falsified}} ) \)
7: If \( C_{\text{conflict}} = \emptyset \) then return unsatisfiable
8: \( \text{BACK\_TRACK}( C_{\text{conflict}} ) \)
9: \( \mathcal{F} := \text{SIMPLIFY}( \mathcal{F} \cup \{ C_{\text{conflict}} \} ) \)
10: end while
11: end while
Learning conflict clauses

[Marques-SilvaSakallah’96]
Learning conflict clauses

\[ (\neg x_1 \lor \neg x_3 \lor x_5 \lor x_{17} \lor \neg x_{19}) \]

tri-asserting clause
Learning conflict clauses

\[(x_{10} \lor \neg x_8 \lor x_{17} \lor \neg x_{19})\]

first unique implication point
Learning conflict clauses

\[(x_2 \lor \neg x_4 \lor \neg x_8 \lor x_{17} \lor \neg x_{19})\]

second unique implication point
Data-structures

Watch pointers
Simple data structure for unit propagation

Variables

Clauses

\[
\begin{array}{c|c|c}
\vdots & -1 & -2 \\
\vdots & -1 & 2 \\
\vdots & 1 & -2 \\
1 & 1 & 2 \\
2 & 3 & -1 \\
3 & -3 & 1 \\
\vdots & \vdots & \vdots
\end{array}
\]
\[ \varphi = \{ x_1 = \ast, x_2 = \ast, x_3 = \ast, x_4 = \ast, x_5 = \ast, x_6 = \ast \} \]
\[ \varphi = \{ x_1 = \ast, x_2 = \ast, x_3 = \ast, x_4 = \ast, x_5 = 1, x_6 = \ast \} \]
\( \phi = \{ x_1 = \ast, x_2 = \ast, x_3 = 1, x_4 = \ast, x_5 = 1, x_6 = \ast \} \)
\[ \varphi = \{ x_1 = *, x_2 = *, x_3 = 1, x_4 = *, x_5 = 1, x_6 = * \} \]
\[ \varphi = \{ x_1 = 1, x_2 = *, x_3 = 1, x_4 = *, x_5 = 1, x_6 = * \} \]
\[ \varphi = \{ x_1 = 1, x_2 = *, x_3 = 1, x_4 = *, x_5 = 1, x_6 = * \} \]
\[ \varphi = \{ x_1 = 1, x_2 = *, x_3 = 1, x_4 = 0, x_5 = 1, x_6 = * \} \]
\[ \varphi = \{ x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0, x_5 = 1, x_6 = * \} \]
\[ \varphi = \{ x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0, x_5 = 1, x_6 = 1 \} \]
\[ \varphi = \{ x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0, x_5 = 1, x_6 = 1 \} \]
Only examine (get in the cache) a clause when both
- a watch pointer gets falsified
- the other one is not satisfied

While backjumping, just unassign variables
Conflict clauses $\rightarrow$ watch pointers
No detailed information available
Not used for binary clauses
Average Number Clauses Visited Per Propagation
Percentage visited clauses with other watched literal true

![Data-structures](image-url)
Heuristics
Variable selection heuristics

- aim: minimize the search space
- plus: could compensate a bad value selection
Most important CDCL heuristics

Variable selection heuristics
- aim: minimize the search space
- plus: could compensate a bad value selection

Value selection heuristics
- aim: guide search towards a solution (or conflict)
- plus: could compensate a bad variable selection, cache solutions of subproblems [PipatsrisawatDarwiche’07]
Heuristics

Most important CDCL heuristics

Variable selection heuristics
- aim: minimize the search space
- plus: could compensate a bad value selection

Value selection heuristics
- aim: guide search towards a solution (or conflict)
- plus: could compensate a bad variable selection, cache solutions of subproblems [PipatsrisawatDarwiche’07]

Restart strategies
- aim: avoid heavy-tail behavior [GomesSelmanCrato’97]
- plus: focus search on recent conflicts when combined with dynamic heuristics
Variable selection heuristics

Based on the occurrences in the (reduced) formula

- examples: Jeroslow-Wang, Maximal Occurrence in clauses of Minimal Size (MOMS), look-aheads
- not practical for CDCL solver due to watch pointers
Variable selection heuristics

Based on the occurrences in the (reduced) formula examples: Jeroslow-Wang, Maximal Occurrence in clauses of Minimal Size (MOMS), look-aheads

not practical for CDCL solver due to watch pointers

Variable State Independent Decaying Sum (VSIDS)

original idea (zChaff): for each conflict, increase the score of involved variables by 1, half all scores each 256 conflicts

improvement (MiniSAT): for each conflict, increase the score of involved variables by $\delta$ and increase $\delta := 1.05\delta$
Heuristics

Visualization of VSIDS in PicoSAT

http://www.youtube.com/watch?v=M0jhFywLre8
Value selection heuristics

Based on the occurrences in the (reduced) formula

- examples: Jeroslow-Wang, Maximal Occurrence in clauses of Minimal Size (MOMS), look-aheads
- not practical for CDCL solver due to watch pointers
Value selection heuristics

Based on the occurrences in the (reduced) formula
- examples: Jeroslow-Wang, Maximal Occurrence in clauses of Minimal Size (MOMS), look-aheads
- not practical for CDCL solver due to watch pointers

Based on the encoding / consequently
- negative branching (early MiniSAT) [EenSörensson’03]
Value selection heuristics

Based on the occurrences in the (reduced) formula
- examples: Jeroslow-Wang, Maximal Occurrence in clauses of Minimal Size (MOMS), look-aheads
- not practical for CDCL solver due to watch pointers

Based on the encoding / consequently
- negative branching (early MiniSAT) [EenSörensson’03]

Based on the last implied value (phase-saving)
- introduced to CDCL [PipatsrisawatDarwiche’07]
- already used in local search [HirschKojevnikov’01]
Heuristics: Phase-saving

Selecting the last implied value remembers solved components

negative branching

phase-saving
Restarts in CDCL solvers:

- Counter heavy-tail behavior
- Unassign all variables but keep the (dynamic) heuristics

[GomesSelmanCrato’97]
Restarts in CDCL solvers:

- Counter heavy-tail behavior \[ \text{GomesSelmanCrato'97} \]
- Unassign all variables but keep the (dynamic) heuristics

Restart strategies: \[ \text{Walsh'99, LubySinclairZuckerman'93} \]

- Geometrical restart: e.g. 100, 150, 225, 333, 500, 750, …
- Luby sequence: e.g. 100, 100, 200, 100, 100, 200, 400, …
Restarts in CDCL solvers:

- Counter heavy-tail behavior  
  [GomesSelmanCrato’97]
- Unassign all variables but keep the (dynamic) heuristics

Restart strategies:  

- Geometrical restart: e.g. 100, 150, 225, 333, 500, 750, ...  
  [Walsh’99, LubySinclairZuckerman’93]
- Luby sequence: e.g. 100, 100, 200, 100, 100, 200, 400, ...  

Rapid restarts by reusing trail:  

- Partial restart same effect as full restart  
  [vanderTakHeuleRamos'11]
- Optimal strategy Luby-1: 1, 1, 2, 1, 1, 2, 4, ...
Conflict-Clause Minimization
Self-Subsumption

Use self-subsumption to shorten conflict clauses

\[
\frac{C \lor l \quad D \lor \overline{l}}{D} \quad C \subseteq D \quad \frac{(a \lor b \lor l) \quad (a \lor b \lor c \lor \overline{l})}{(a \lor b \lor c)}
\]

Conflict clause minimization is an important optimization.
Self-Subsumption

Use self-subsumption to shorten conflict clauses

\[
\frac{C \lor l}{D} \quad \frac{D \lor \bar{l}}{C \subseteq D}
\]

\[
\frac{(a \lor b \lor l) \quad (a \lor b \lor c \lor \bar{l})}{(a \lor b \lor c)}
\]

Conflict clause minimization is an important optimization.

Use implication chains to further minimization:

\[
\ldots (\bar{a} \lor b)(\bar{b} \lor c)(a \lor c \lor d) \ldots \quad \Rightarrow \\
\ldots (\bar{a} \lor b)(\bar{b} \lor c)(c \lor d) \ldots
\]
Conflict-clause minimization

\[ x_1 = 0 \quad x_2 = 1 \quad x_3 = 0 \]
\[ x_4 = 1 \quad x_5 = 0 \quad x_6 = 1 \quad x_7 = 0 \]
\[ x_8 = 1 \quad x_9 = 0 \quad x_{11} = 0 \quad x_{12} = 1 \quad x_{13} = 0 \]
\[ x_{10} = 1 \quad x_{14} = 1 \quad x_{15} = 0 \quad x_{13} = 1 \]
Conflict-clause minimization

(Sörensson & Biere'09)

first unique implication point

\[
\begin{align*}
&x_1 = 0 \\
x_4 = 1 \\
x_8 = 1 \\
x_{10} = 1 \\
&x_2 = 1 \\
x_5 = 0 \\
x_9 = 0 \\
x_{11} = 0 \\
&x_3 = 0 \\
x_6 = 1 \\
x_{12} = 1 \\
x_{13} = 0 \\
&x_7 = 0 \\
x_{14} = 1 \\
x_{15} = 0 \\
&x_{13} = 1
\end{align*}
\]
Conflict-clause minimization

(Sörensson | Biere'09)

x_1 = 0, x_2 = 1, x_3 = 0
x_4 = 1, x_5 = 0, x_6 = 1
x_8 = 1, x_9 = 0, x_7 = 0
x_{10} = 1, x_{11} = 0, x_{13} = 0

(x_1 \lor \overline{x}_4 \lor \overline{x}_8 \lor \overline{x}_{10})

last unique implication point
Conflict-clause minimization

\[ (\bar{x}_2 \lor x_5 \lor \bar{x}_6 \lor x_{11}) \]

reduced conflict clause
Conflict-clause minimization

\[ (\overline{x}_2 \lor x_5 \lor \overline{x}_{11}) \]

minimized conflict clause

\[ x_1 = 0 \]
\[ x_4 = 1 \]
\[ x_8 = 1 \]
\[ x_{10} = 1 \]
\[ x_2 = 1 \]
\[ x_5 = 0 \]
\[ x_9 = 0 \]
\[ x_{11} = 0 \]
\[ x_3 = 0 \]
\[ x_6 = 1 \]
\[ x_{12} = 1 \]
\[ x_{13} = 0 \]
\[ x_7 = 0 \]
\[ x_{14} = 1 \]
\[ x_{15} = 0 \]
\[ x_{13} = 1 \]
Conclusions: state-of-the-art CDCL solver

Key contributions to CDCL solvers:

- concept of conflict clauses (grasp)  
  [Marques-SilvaSakallaxah’96]
- restart strategies  
  [GomesSC’97,LubySZ’93]
- 2-watch pointers and VSIDS (zChaff)  
  [MoskewiczMZZM’01]
- efficient implementation (Minisat)  
  [EenSörensson’03]
- phase-saving (Rsat)  
  [PipatsrisawatDarwiche’07]
- conflict-clause minimization  
  [SörenssonBiere’09]

+ Pre- and in-processing techniques