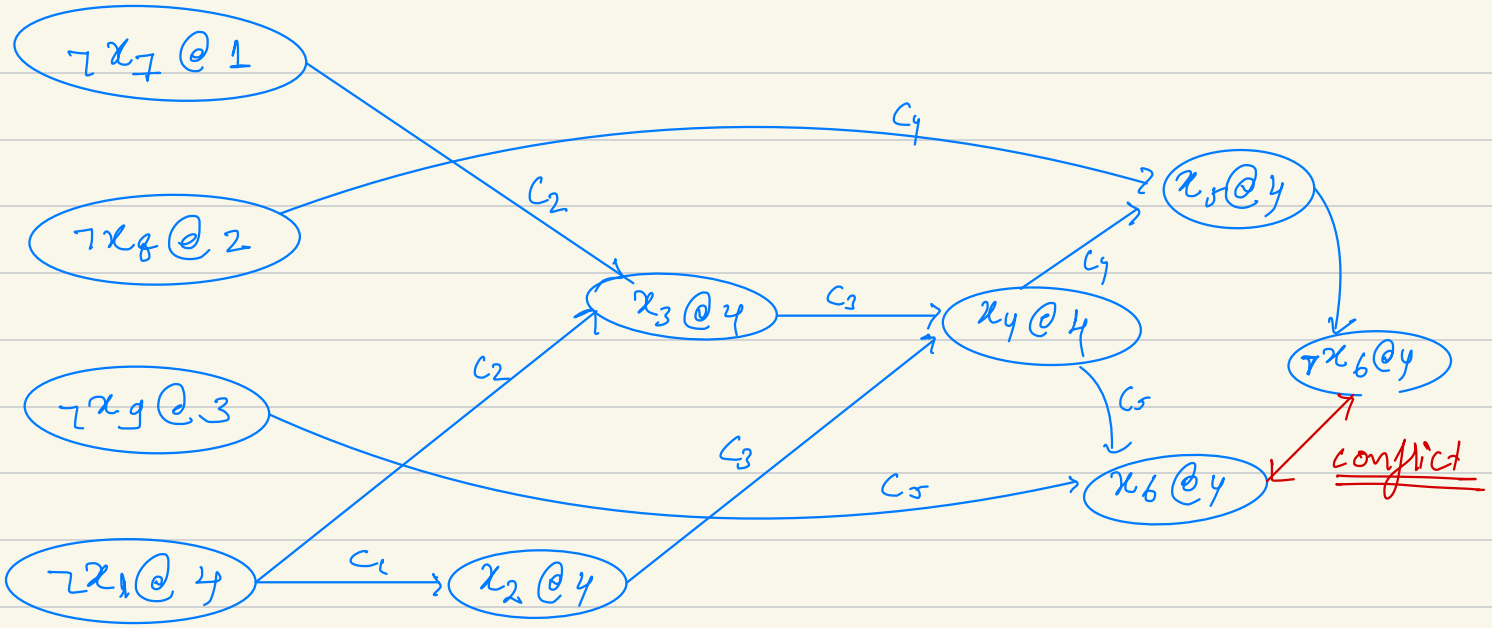
choices for conflict clauses

- Smaller conflict clauses prune more search space -

1. Cuts in the implication graphs
2. 1-UIP
   - Unique implication point

Unique Implication point (UIP) :-

UIP is any node at the current decision level such that any path from the decision variable to the conflict node must pass through it.



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

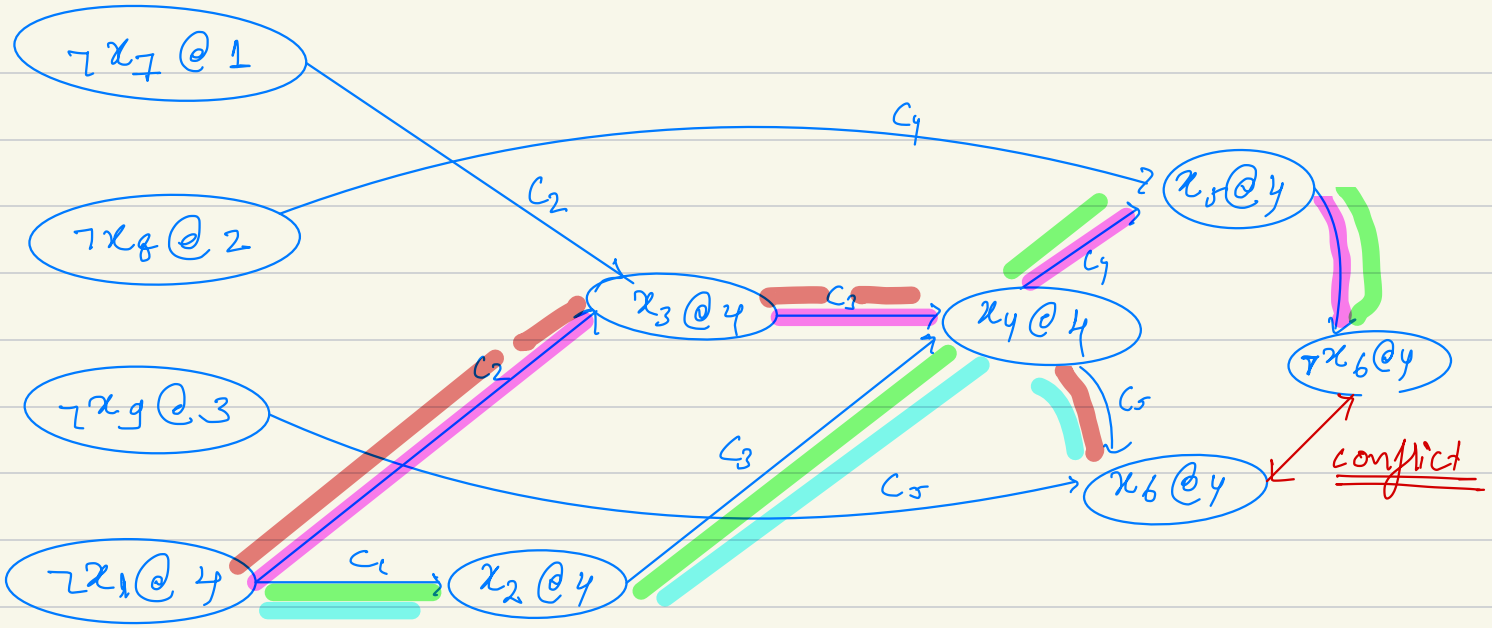
→ current decision level is 4

At decision level 4 we assigned $x_1 = 0$ & that lead to conflict.

Order

$\Rightarrow \langle x_7, x_8, x_9, x_1, x_2, x_4, x_5, x_3, x_6 \rangle$

polarity always 0.



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

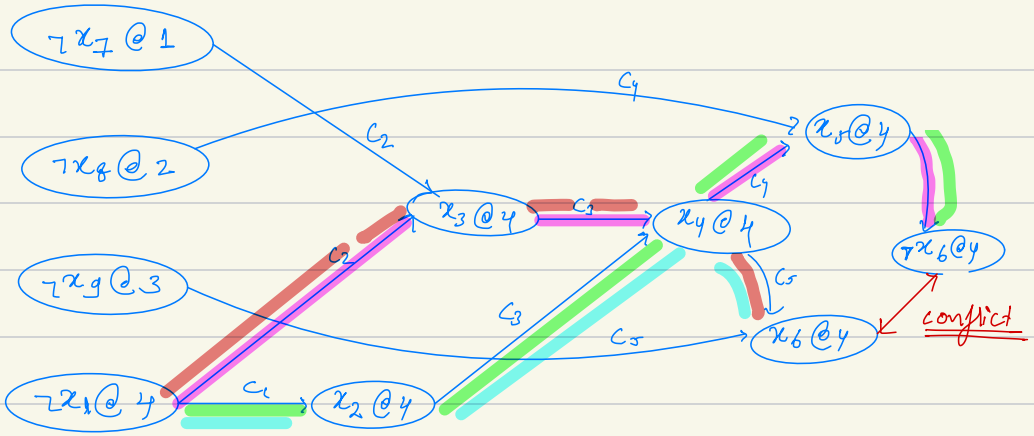
→ current decision level is 4

At decision level 4 we assigned $x_1 = 0$ & that lead to conflict.

Order

$\Rightarrow \langle x_7, x_8, x_9, x_1, x_2, x_4, x_5, x_3, x_6 \rangle$

polarity always 0.



UIP : $\{\neg x_1 @ 4,\ x_4 @ 4\}$

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

Order

$\langle x_7, x_8, x_9, x_1, x_2, x_4, x_5, x_3, x_6 \rangle$

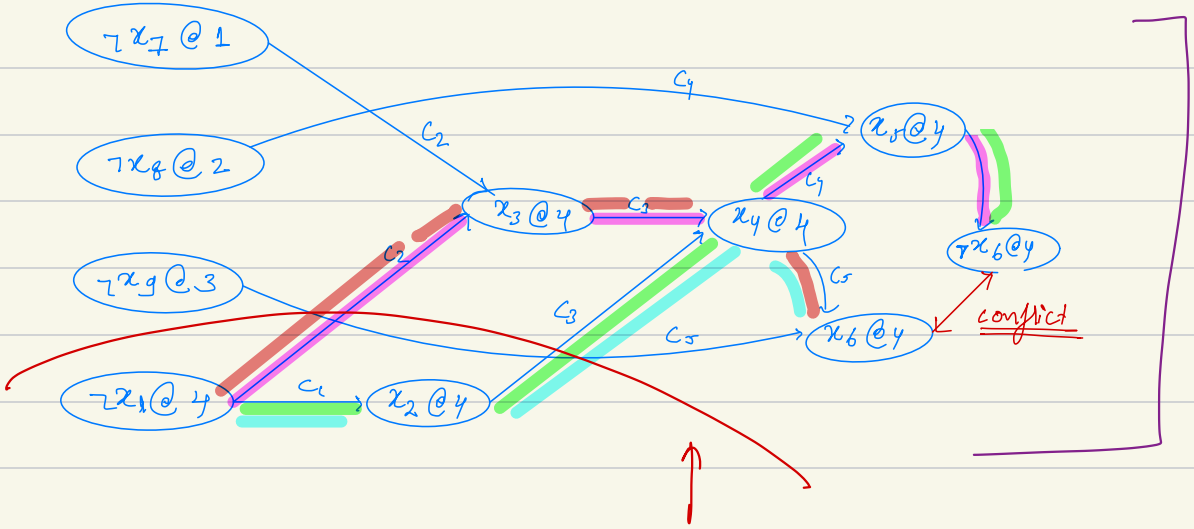
polarity always 0.

→ current decision level is 4

at decision level 4 we assigned $x_1 = 0$ & that lead to conflict

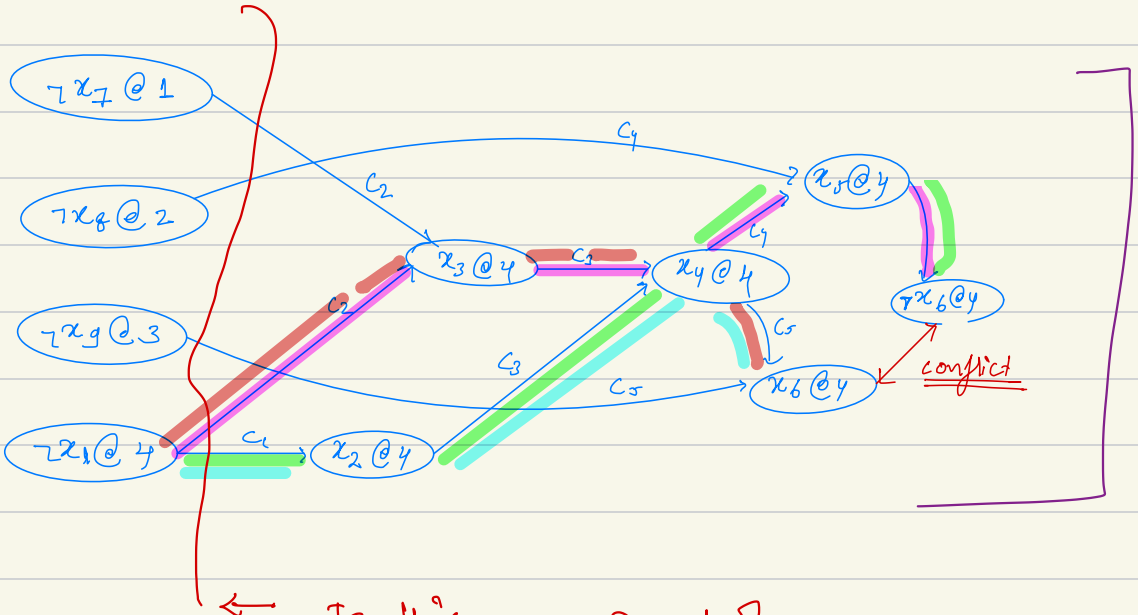
UIP cut

If l is a UIP, then corresponding UIP cut is (A, B), where B contains all the successors of l from which there is a path to $\perp$ and A contains the rest.



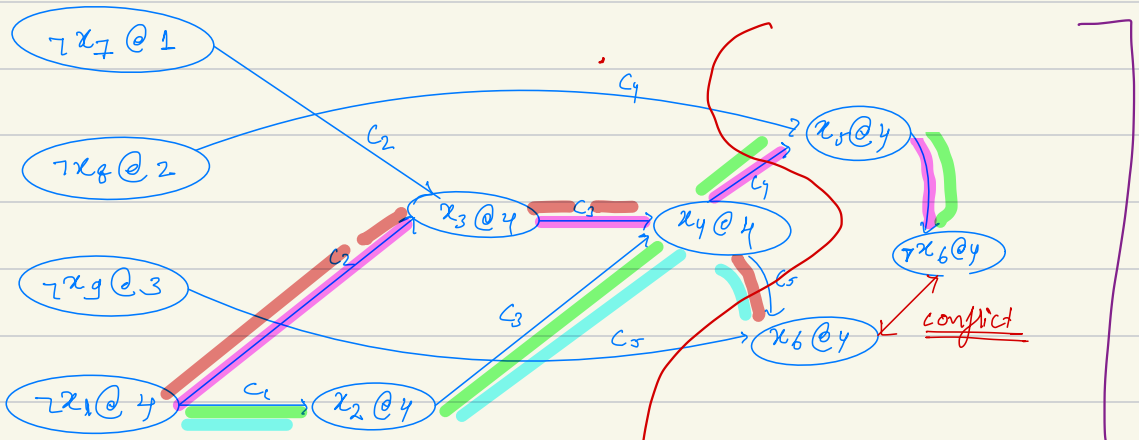
UIP : $\{\neg x_1 @ 4, x_4 @ 4\}$

Is this a UIP cut?



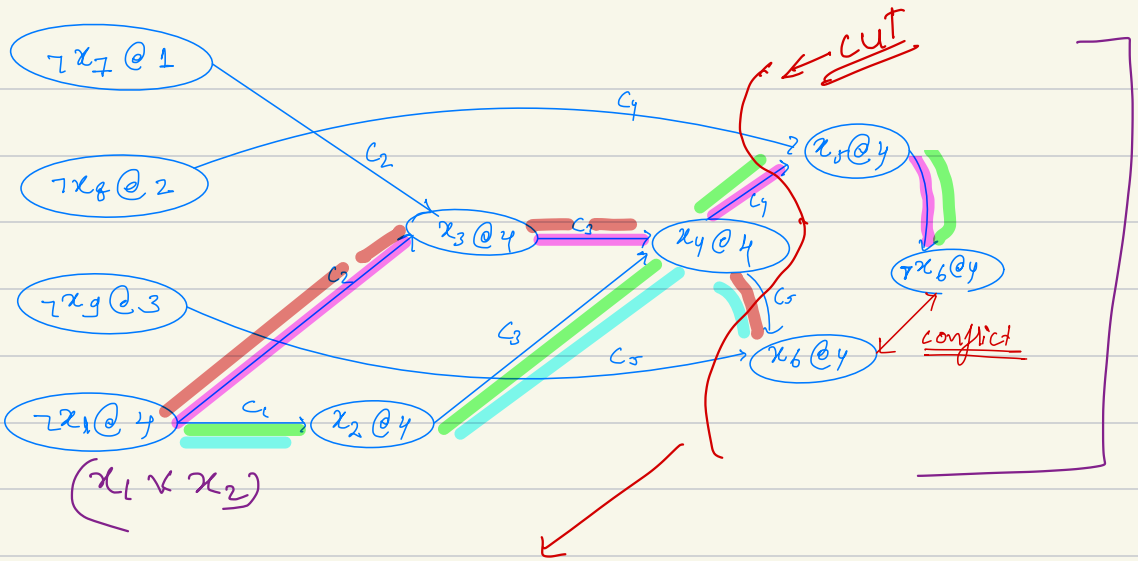
UIP : $\{\neg x_1@4, x_4@4\}$

← Is this a UIP cut?



UIP : $\{\neg x_1 @ 4, x_4 @ 4\}$

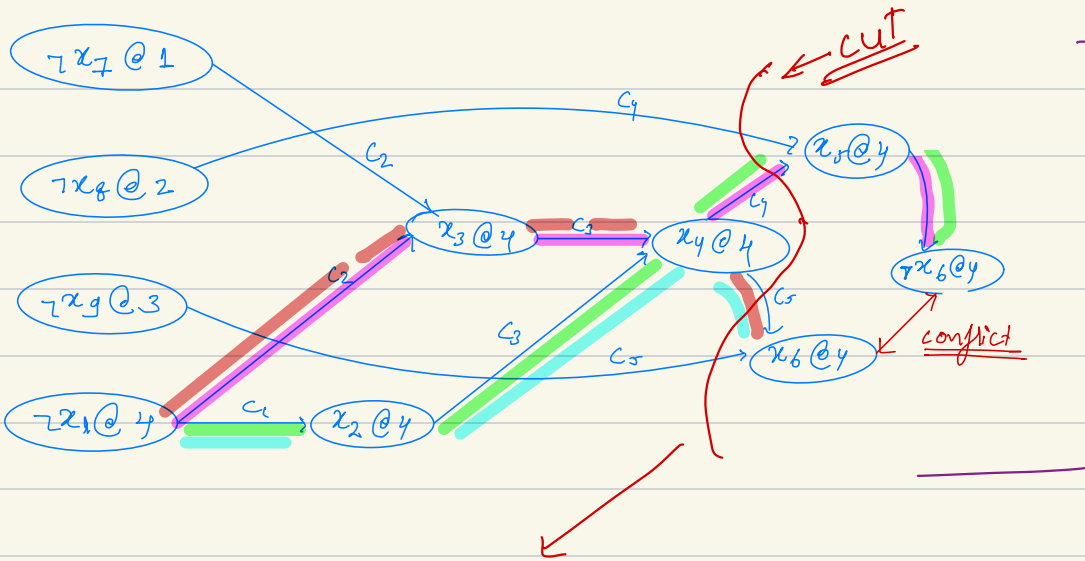
Is this a UIP cut ?



UIP : { ¬x1 @ 4, x4 @ 4 }

1-UIP cut or first UIP cut

The cut that divides the implication graph right before the first UIP encountered on the path leading from the conflict node back to decision node.



UIP : $\{\neg x_1 @ 4, x_4 @ 4\}$

conflict clause :-

$\neg(\neg x_8 \wedge x_4 \wedge \neg x_9)$

$x_8 \vee \neg x_4 \vee x_9$

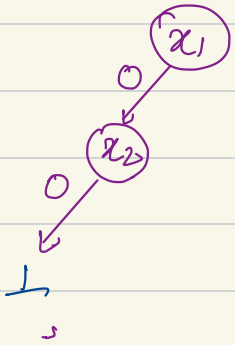
1-UIP cut or first UIP cut

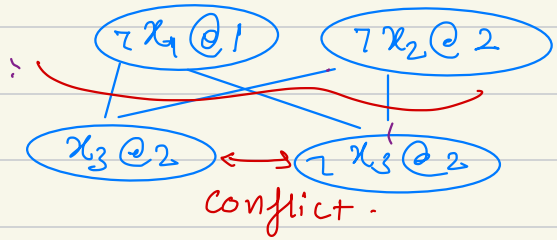
The cut that divides the implication graph right before the first UIP encountered on the path leading from the conflict node back to decision node.

Check if F is SAT or UNSAT using CDCL Algo?

$$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6) \wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6)$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6) \wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6)$$



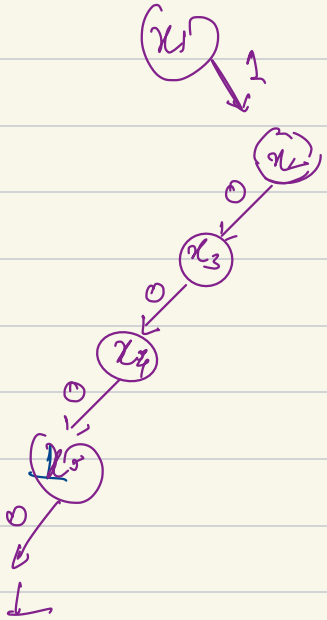


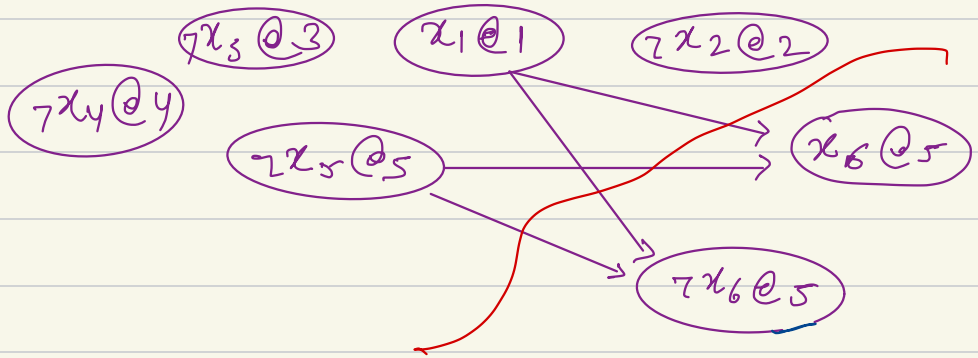
first UIP cut : learned clause

$x_1 \vee x_2$

Backtrack to level 1, set $x_1 = 1$

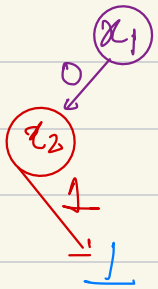
$$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6) \wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6) \wedge (x_1 \vee x_2)$$

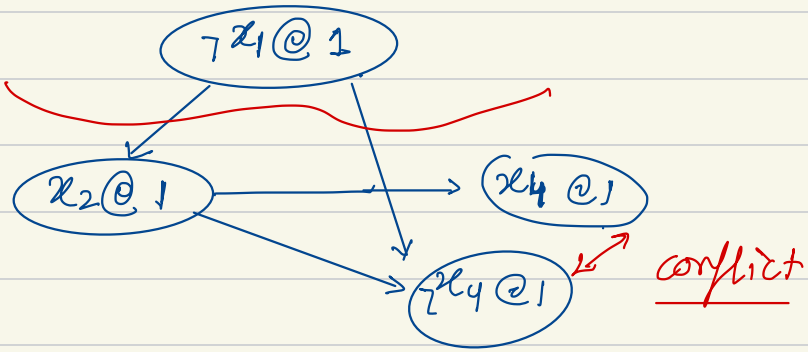




clause learned $\neg x_1 \vee x_5$

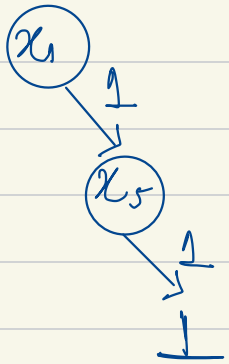
$$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge$$
$$(x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6)$$
$$\wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6)$$
$$\wedge (x_1 \vee x_2) \wedge (\neg x_1 \vee x_5)$$

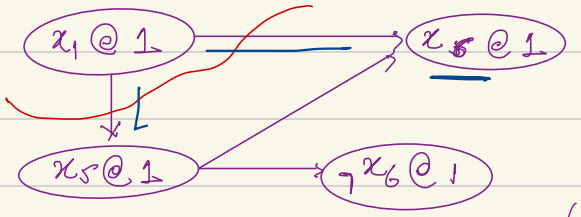




clause learned x_1

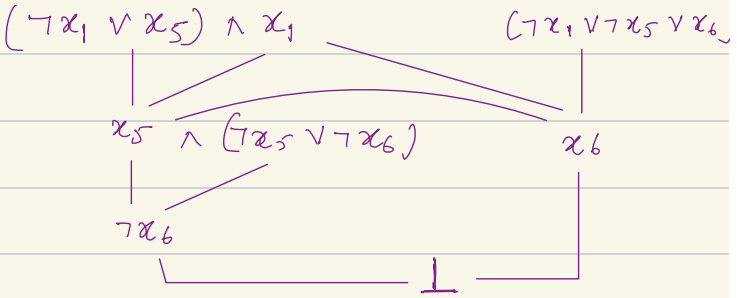
$$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge$$
$$(x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee x_6)$$
$$\wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6)$$
$$\wedge (x_1 \vee x_2) \wedge (\neg x_1 \vee x_5) \wedge x_1$$





conflict clause $\neg x_1$

Formula is UNSAT



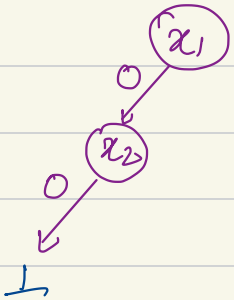
F is UNSAT

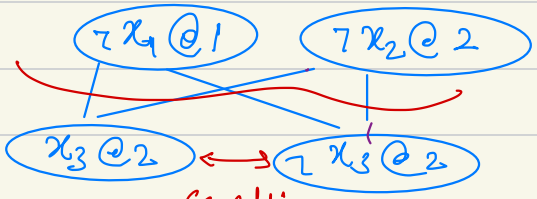
Different Backtracking approach ?

Check if F is SAT or UNSAT using CDCL Algo?

$$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6) \wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6)$$

$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge$
$(x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6)$
$\wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_4 \vee \neg x_5 \vee x_6)$





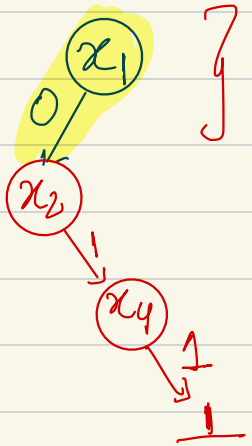
conflict.

first UIP cut : learned clause

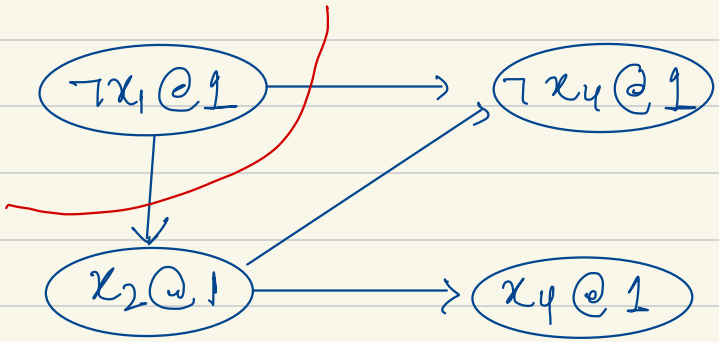
$x_1 \vee x_2$

Backtrack to level 1

$$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge$$
$$(x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6)$$
$$\wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6)$$
$$\wedge (x_1 \vee x_2)$$



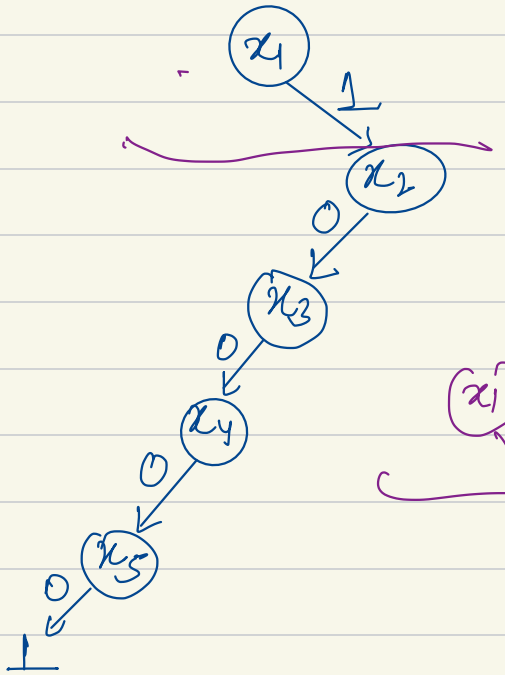
Backtrack to level 1, keep its priority

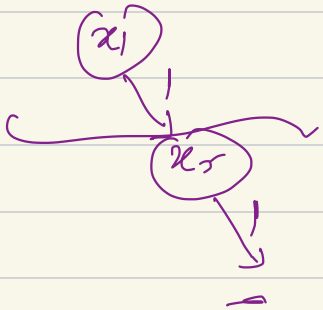


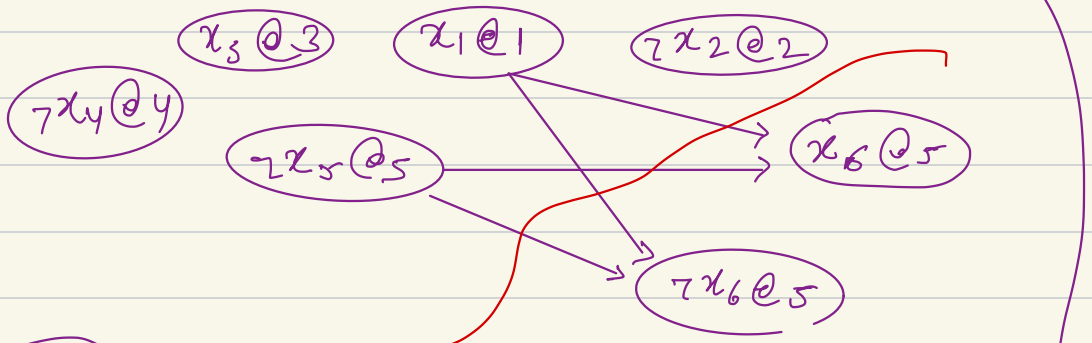
learned clause x_1

Backtrack to level 0

$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge$
$(x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6)$
$\wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6)$
$\wedge (x_1 \vee x_2) \wedge x_1$



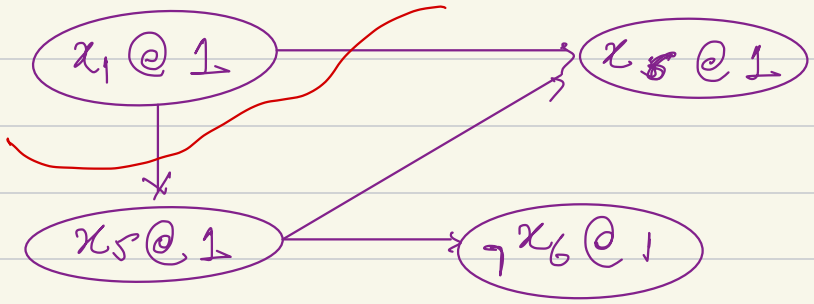




clause learned $\neg x_1 \vee x_5$

Backtrack to level 1.

$F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge$
$(x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6) \wedge (\neg x_1 \vee x_5 \vee \neg x_6)$
$\wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_1 \vee \neg x_5 \vee x_6)$
$\wedge (x_1 \vee x_2) \wedge x_1 \wedge (\neg x_1 \vee x_5)$



conflict clause $\neg x_1$.

Formula is UNSAT

FACTS

- DPLL could handle formulas with ≤ 2000 clauses
- Modern SAT solvers cope with industrial instances of 100,000,000 clauses.
  - the introduction of CDCL was the birth of modern highly scalable SAT solving.

All UIP clause learned & different backtracking → 1996 – GRASP by Marques-Silva & Karem A Sakallah ICCAD, GRASP- A new search algorithm for SAT solving.

CAV award 2009

2001 chaff by Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, Sharad Malik DAC, Chaff: Engineering an efficient SAT solver.

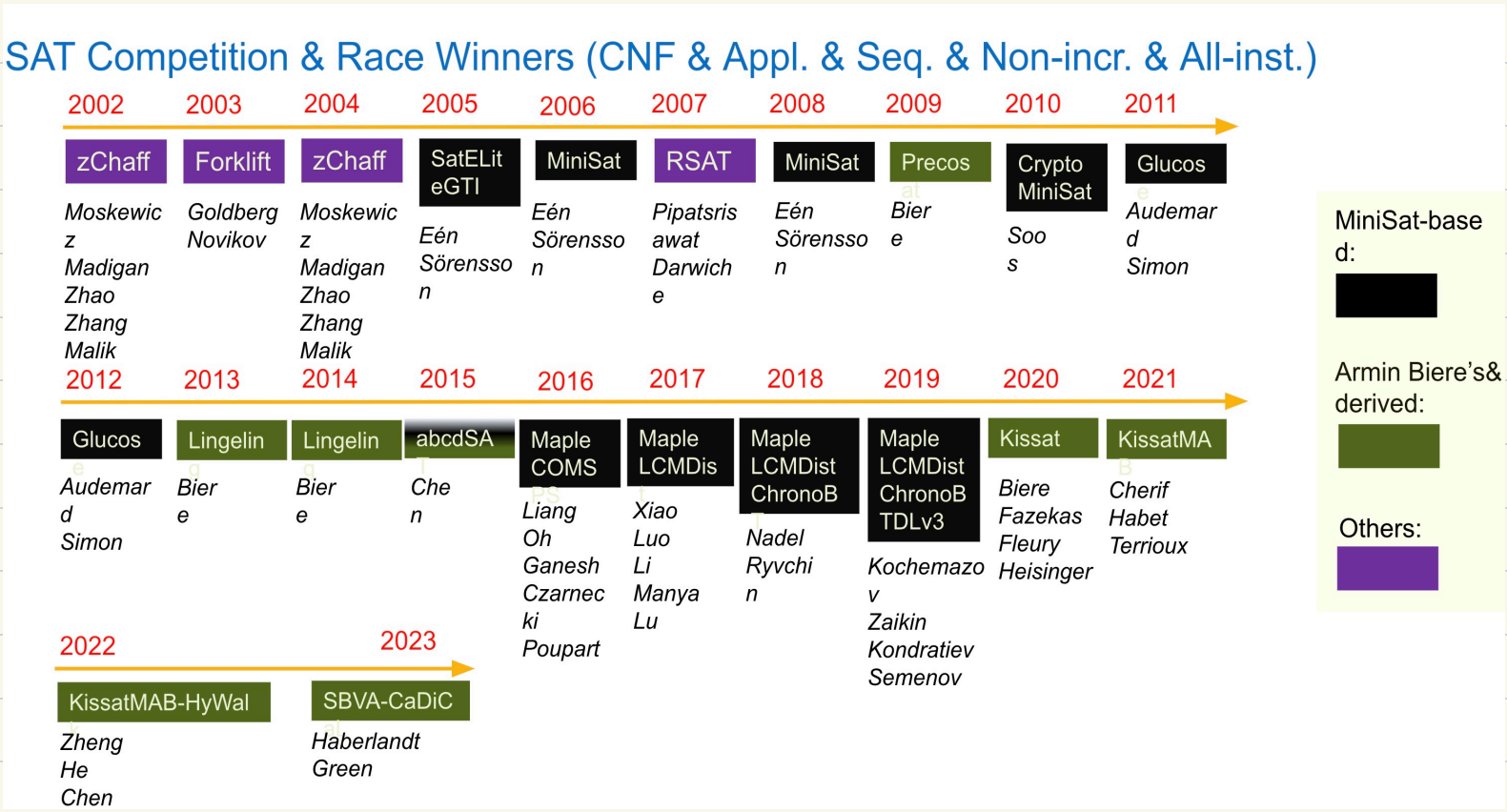
1996 - GRASP

- chronological Backtracking (CB)
- learned a clause for every other UIP of the conflict decision level.
- Boolean constant Propogration, UNIT propogration, UNIT Resolutions.

2001 - Chaff

- Non chronological Backtracking (NCB)
- learned a clause corresponding to 1-UIP of the conflict decision level.
- Boolean constant Propogration, UNIT propogration, UNIT Resolutions.
- Many advanced data structures.

Taken from Alexander Nadel 's SAT+SMT school '24 slides.



conflict analysis

1996 — GRASP — CB

2001 — Chaff — NCB

2018 — MapleSAT

MapleSAT: choose b/w CB & NCB

first 4000 conflict : NCB (warm-up phase)

After 4000, if Backtrack level < 100 : CB

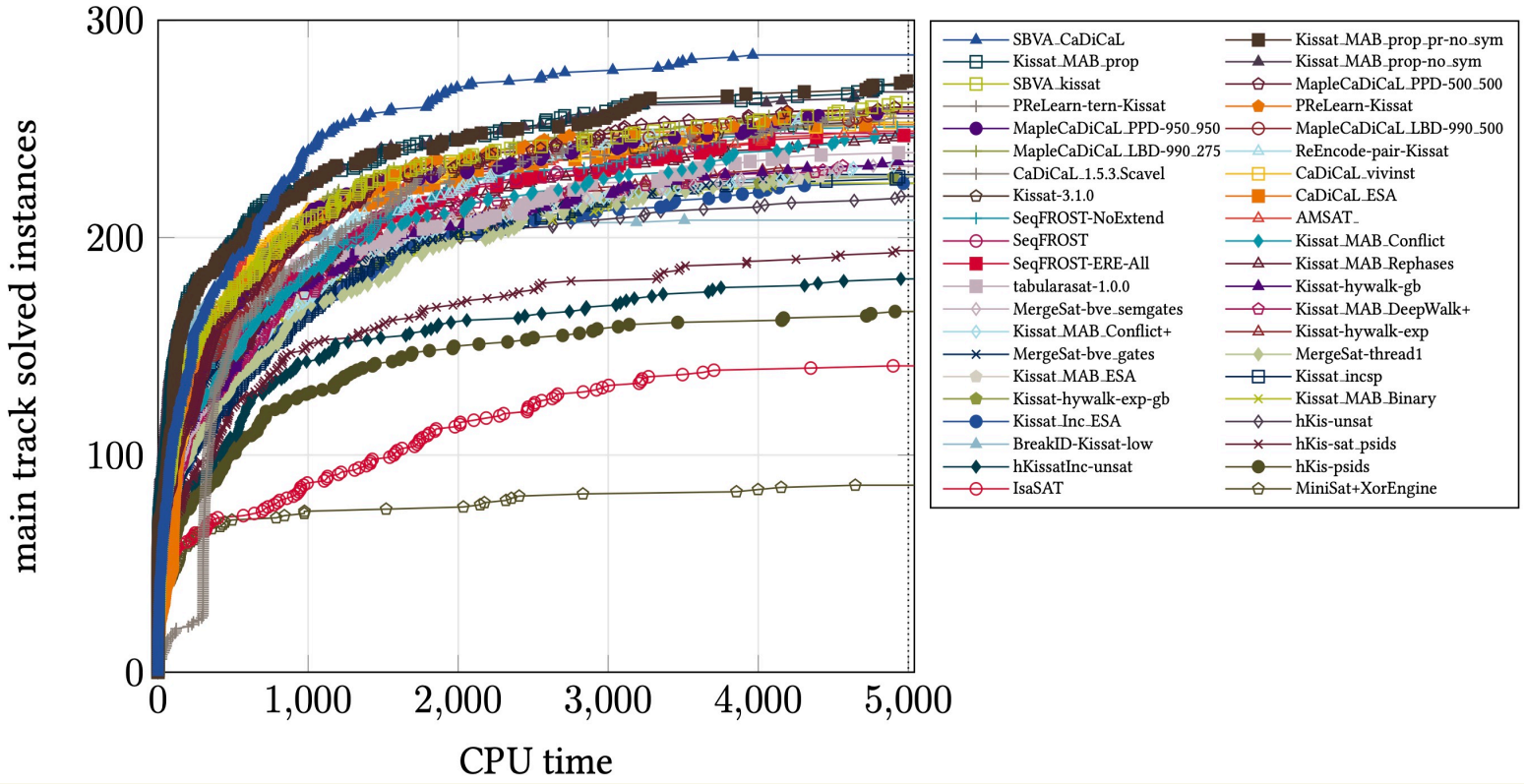
State of the art solvers

CaDiCal, kissat, CryptoMiniSAT also combines NCB & CB.

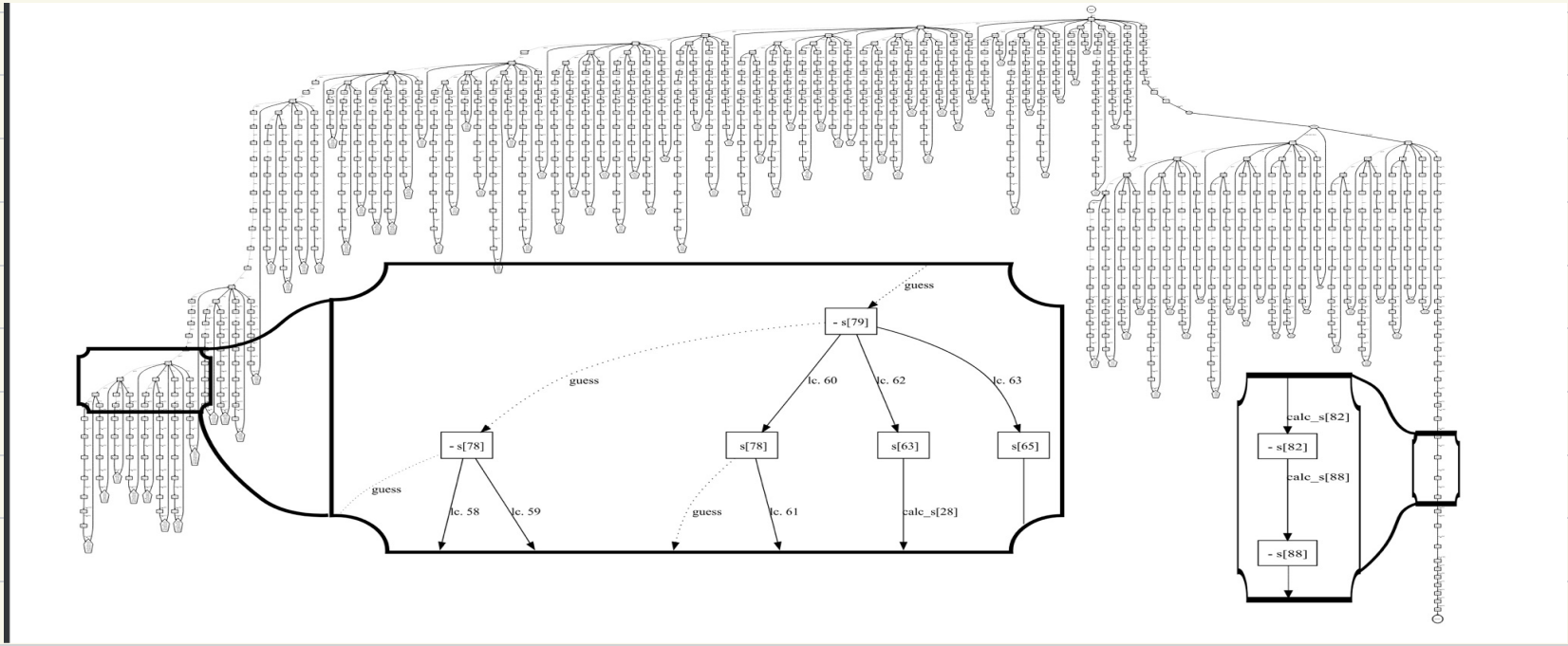
NCB: Non chronological Backtracking

CB: Chronological Backtracking.

SAT competition : 2023



Taken from Mate Soos (CryptominiSAT)'s slides.



Heuristics Plays an Important Role in Understanding this Graph,

heuristics to improve the performance of SAT solver

Lazy data structure

- 2 watched literals
- Pure literals

Runtime choices

- Variable ordering
- Restarts
- learned clause deletion
- Phase saving.

heuristics to improve the performance of SAT solver

Lazy datastructure

- 2 watched literals
- Pure literals

Runtime choices

- Variable ordering
- Restarts
- learned clause deletion
- Phase saving.

Optimal storage

- variables
- clauses
- occurrence maps

Pre/In processing techniques