

Proof Complexity and Solving LAB

New Year Recap

Dr. Joshua Blinkhorn

Friedrich-Schiller-Universität Jena

<https://github.com/JoshuaBlinkhorn/SAT-LAB>

Goals

- Implementation of SAT solving algorithms
 - (a) 2-SAT (polynomial time)
 - (b) DPLL
 - (c) CDCL
 - watched literals
 - clause learning
 - decision heuristics
 - restart strategy
 - (d) QBF expansion..
- Practical programming experience
 - use your favourite language (Python, C, C++, Java, ..)
 - recommended: Python

Reaching the First Conflict

decision
level

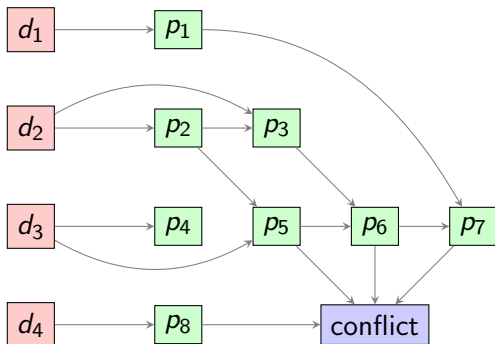
0

1

2

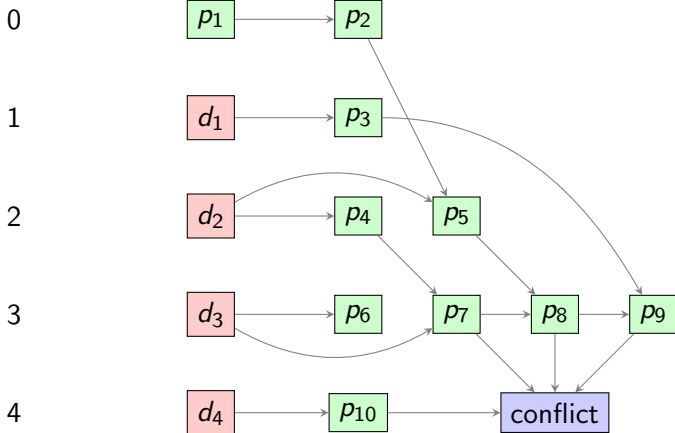
3

4



Reaching Later Conflicts

decision
level



Learning Unit Clauses (!?)

- a learned unit clause should propagate:
 - (a) immediately
 - (b) forever
- **result**: adding learned unit clauses is **unnecessary** ..
- .. just make the assignment at decision level 0 instead
- decision level 0 is never undone
- watched literals do not work with unit clauses

Reaching Later Conflicts (Revisited)

decision
level

0

a_1

p_1

1

d_1

p_2

2

d_2

p_3

p_4

3

d_3

p_5

p_6

p_7

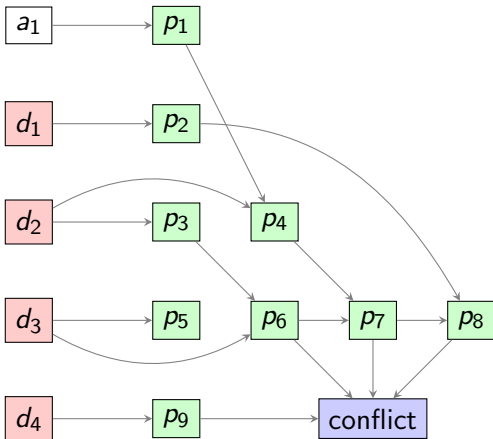
p_8

4

d_4

p_9

conflict



CDCL Pseudocode

```
function CDCL-solver( $\Phi$ )  
  decision-level  $\leftarrow$  0  
  while there are unassigned variables  
    decision-level++  
    decide()  
     $C_{\text{conflict}} \leftarrow \text{propagate}()$   
    while  $C_{\text{conflict}}$  is not null  
      if decision-level = 0 return UNSAT  
       $C_{\text{learned}} \leftarrow \text{analyse-conflict}(C_{\text{conflict}})$   
      if  $C_{\text{conflict}}$  is unit  
        backtrack(0)  
        assign unit literal  
      else  
        backtrack(asserting-level( $C_{\text{learned}}$ ))  
         $\Phi \leftarrow \Phi \wedge C_{\text{learned}}$   
         $C_{\text{conflict}} \leftarrow \text{propagate}()$   
      apply-restart-policy()  
  return SAT
```

#assuming Φ is preprocessed

#adds assignment to trail
#returns conflict clause or null

#changes trail and DL

Watched Literals

- when searching for conflict, we only care about unit clauses
- a clause becomes unit when:
 - it has exactly one unassigned literal, **and**
 - all other literals are falsified
- sufficient to watch just two literals in every clause
- maintain this invariant for each clause:
 - **either** both watched literals are unassigned
 - **or** at least one watched literal is satisfied
- **important:** if both watched literals are assigned, and one is falsified: its decision level should be **no lower than the satisfied one**

How is it done?

- many options - here's one:
 - (a) maintain a list of 'watched clauses' for each literal
 - (b) process a variable assignment by:
 1. visit watched clauses **for the falsified literal** in order
 2. make sure the invariant holds
 - you may need to 'swap the watch'
 3. if clause becomes unit, add unit assignment to trail
 - **note:** in this case, both watched literals have the same decision level
 - so there is no need to swap the watch
 - (c) if the invariant cannot be maintained, we reach conflict

Conflicts and Backtracking

decision
level

Invariant?

0

always

1

d_1

p_1

yes

2

d_2

p_2

p_3

yes

3

d_3

p_4

p_5

p_6

p_7

yes

4

d_4

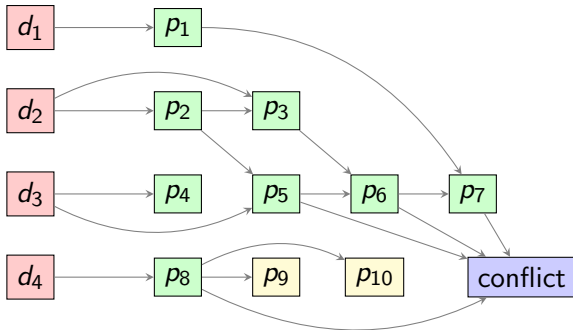
p_8

p_9

p_{10}

conflict

no



Swapping the Watch

$$x_1 \mapsto 0$$

$$x_2 \mapsto 1$$

$$x_3 \mapsto 0$$

$$x_4 \mapsto 1$$

$$x_5 \mapsto 1$$

trivial case: $(x_6 \vee \overline{x_7} \vee \overline{x_4}) \rightarrow (x_6 \vee \overline{x_7} \vee \overline{x_4})$

sat case: $(\overline{x_1} \vee \overline{x_4} \vee \overline{x_2}) \rightarrow (\overline{x_1} \vee \overline{x_4} \vee \overline{x_2})$

sat-swap case: $(\overline{x_4} \vee \overline{x_6} \vee x_3 \vee x_5) \rightarrow (\overline{x_4} \vee \overline{x_6} \vee x_3 \vee x_5)$

swap case: $(\overline{x_4} \vee \overline{x_6} \vee x_3 \vee x_7) \rightarrow (\overline{x_4} \vee \overline{x_6} \vee x_3 \vee x_7)$

Unit Clauses and Conflicts

$x_1 \mapsto 0$	$x_2 \mapsto 1$	$x_3 \mapsto 0$	$x_4 \mapsto 1$	$x_5 \mapsto 1$
-----------------	-----------------	-----------------	-----------------	-----------------

unit case: $(\overline{x_4} \vee \overline{x_6} \vee x_3) \rightarrow (\overline{x_4} \vee \overline{x_6} \vee x_3)$

add assignment: $x_6 \mapsto 0$

conflict case: $(\overline{x_4} \vee \overline{x_5} \vee x_3) \rightarrow (\overline{x_4} \vee \overline{x_5} \vee x_3)$

add assignment: $x_5 \mapsto 0$

\Rightarrow **CONFLICT**

Watched Literals Task

- implement unit propagation with watched literals in your CDCL solver
- ignore pure literal elimination
- check correctness
- compare the solving time to naive propagation

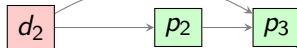
Clause Learning - Cutting the Implication Graph

decision
level

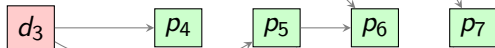
1



2



3



4

