Proof Complexity and Solving LAB

Conflict-Driven Clause Learning (CDCL)

Dr. Joshua Blinkhorn

Friedrich-Schiller-Universität Jena
https://github.com/JoshuaBlinkhorn/SAT-LAB

Goals

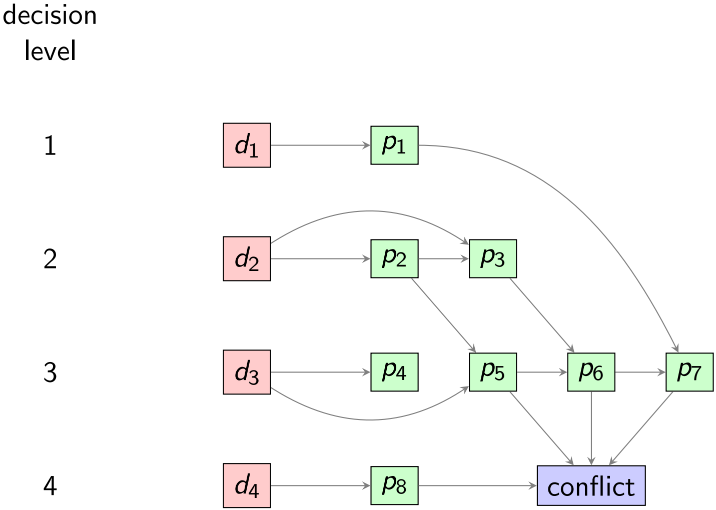
- Implemenatation of SAT solving algorithms
  - (a) 2-SAT (polynomial time)
  - (b) DPLL
  - (c) CDCL
    - clause learning
    - watched literals
    - decision heuristics
    - restart strategy
  - (d) QBF expansion..

- Practical programming experience
  - use your favourite language (Python, C, C++, Java, ..)
  - recommended: Python

CDCL versus DPLL

- Similarities
  - unit propagation
  - backtracking search
- Algorithmic improvements in CDCL
  - clause learning: the formula grows
  - more accurate backtracking (a.k.a backjumping)
- Further innovations
  - watched literals (fast propagation)
  - decision heueristics (e.g. VSIDS)
  - restarts (randomisation of initial conditions)
  - phase saving

Reaching Conflict



CDCL Pseudocode

```
function CDCL-solver($\Phi$)                          #assuming $\Phi$ is preprocessed
  decision-level $\leftarrow 0$
  while there are unassigned variables
    decision-level++
    decide()                                          #adds assignment to trail
    $C_{\text{conflict}} \leftarrow$ propagate()      #returns conflict clause or nothing
    while $C_{\text{conflict}}$ is not null
      if decision-level $= 0$ return UNSAT
      $C_{\text{learned}} \leftarrow$ analyse-conflict($C_{\text{conflict}}$)
      $\Phi \leftarrow \Phi \wedge C_{\text{learned}}$
      backtrack($C_{\text{learned}}$)                 #changes trail and decision level
      $C_{\text{conflict}} \leftarrow$ propagate()
    apply-restart-policy()
  return SAT
```

Clause Learning

- Learned clauses:
  - result from deep analysis
  - try to explain and store the reason for conflict
  - are always asserting: they propatagate at a lower decision level
  - that decision level may be arbitrarily low
- Basic clause learning:
  - learn the negation of the decision literals at point of conflict
  - propagates negation of last decision at previous decision level
  - emulates DPLL

Propagation, Heuristics and Restarts

- Use naive propagation as we did in DPLL
- Use any naive decision heuristic
- Forget about restarts for now

CDCL Task

- implement a CDCL solver with:
  - basic clause learning
  - naive propagation
  - default heuristics
  - no restarts
- include a README
- test your solver on random k-SAT formulas
- print statistics, e.g.
  - solving time
  - memory consumption
  - number of decisions
  - number of unit propagations
  - number of conflicts
  - ..?