SAT Solver- Variant 2

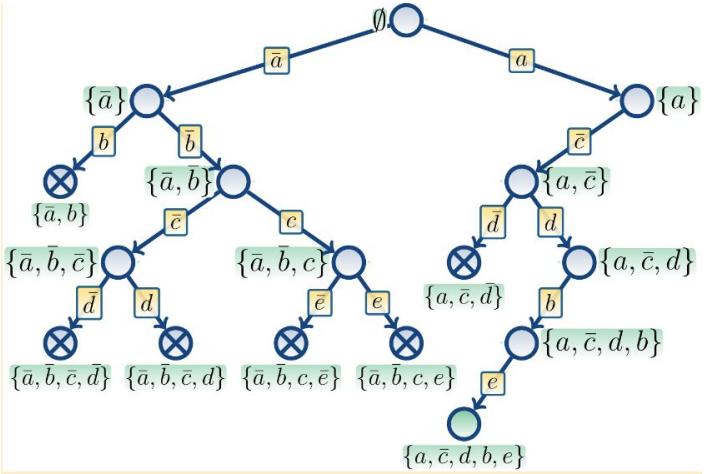
Progress Slides 2

Next Steps from Progress Slides 1

- Fully solidify my understanding of DPLL and SAT concepts, such as backtracking, decision heuristics and conflict resolution
- Plan requirements and design for my solver
- Decide on techniques I will use
- Decide what technologies I would like to use
- Prepare to begin coding

Understanding of DPLL and SAT

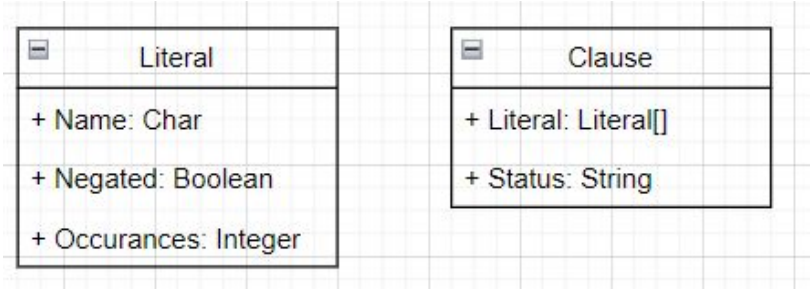
- Backtracking
  - Assigns a truth value to literals at each step
  - If conflict is found, backtrack until no conflict
- Decision Heuristics
  - Help decide which literal to assign at each step
  - Speeds up algorithm, find SAT or UNSAT faster
- Termination
  - When SAT found or,
  - When impossible to backtrack any longer (UNSAT, search space exhausted)



https://users.aalto.fi/~tjunttil/2020-DP-AUT/notes-sat/dpll.html

Requirements and Design

- Requirements
  - Build a working SAT Solver implementing a DPLL algorithm, that can handle large CNF inputs
  - Implement various techniques and decision heuristics to increase efficiency of the search
- Design
  - OOP Approach, still not 100% certain on the design
  - 3 Classes, Literal, Clause and Solver



Techniques and Technology

- Python
  - Easy languages to use and I am very familiar with it already
  - Supports OOP
  - Has libraries that could be helpful
- Decision Heuristics
  - I will need to experiment, here are a few I’ve been looking at:
  - Unit Clause Heuristic, if a clause if a unit clause, set last unassigned variable to true
  - Pure Symbol Heuristic, if a clause only has 1 literal, set it to true
  - Dynamic Largest Individual Sum, decide on literal with highest occurrence first
  - These are subject to change once I start coding if I find a more efficient solution

Literature Review

- An Extensible SAT-Solver by Niklas Een & Niklas Sörensson
  - Gave me a great overview of a finished SAT solver, searching and learning
- The Quest for Efficient Boolean Satisfiability Solvers by Lintao Zhang & Sharad Malik
  - Insight into how to improve efficiency of a solver
- An Implementation of the DPLL Algorithm by Tanbir Ahmed
  - Another great, in heavy detail of DPLL, and outcomes of heuristics
- https://www.diag.uniroma1.it/~liberato/ar/dpll/dpll.html
  - Again, provides in detail examples on backtracking, unit propagation and pure literals

Next Steps

- Begin Coding!!!
- I want to see how DPLL works in practice, so I can start implementing and experimenting with heuristics and search tactics
- Get a basic sat solver near completed/completed