CS202

Mathematics in Computer Science -II

Indian Institute of Technology Kanpur

Assignment 2 Report

18nd February,2022

Harshit Raj (200433)

Shubhan R (200971)

We used the DPLL algorithm to implement our SAT Solver.

Below given is the pseudocode and explanation of the DPLL algorithm.

The following steps are done in the same order

Unit propagation

Suppose there is only one literal in one of the clauses. This means the literal **must be true** for the formula to be true and the corresponding value is in our model and all the clauses containing this literal evaluate to true and can be removed from our formula. From the clauses that have negation of this literal, we remove the negation from the clause as it cannot evaluate to true. While there are unit clauses, we keep refining our formula.

Pure literal

A literal that appears only as itself or it's negation in the whole formula is a pure literal. This literal can be taken as true in our model and all the clauses with the literal can be dropped from our formula. We keep doing this as long as there are pure literals in our formula.

Empty formula

The formula can be empty only if all the clauses have been dropped because they all were true. We drop a clause only when it is true. Therefore formula has a model.

If formula contains empty clause

This clause can never evaluate to true, and therefore by definition of CNF, the formula had a contradiction and is not satisfiable.

Finally

If the code has not evaluated after all the above steps and still has nonempty clauses, we randomly assign one of the literals to true and call the algorithm on the new formula. If it evaluates to false, it means that if the literal is true, there is a contradiction in the formula. Therefore, the literal must be false in the model.

So we assume this case and again do DPLL again. If this evaluates to true, our assumption was right and a model exists. If it evaluates to false, it means that the formula is a contradiction and has no model.

We also used **Mom's heuristic** in code to optimize it. Mom's heuristic states that the more times the literal occurs in short clauses, the more likely it will be true in the model.

```
function DPLL(Φ)
    while there is a unit clause {l} in Φ do
        Φ ← unit-propagate(l, Φ);
    while there is a literal l that occurs pure in Φ do
        Φ ← pure-literal-assign(l, Φ);
    if Φ is empty then
        return true;
    if Φ contains an empty clause then
        return false;
    l ← choose-literal(Φ);
    return DPLL(Φ ∧ {l}) or DPLL(Φ ∧ {not(l)});
```