Documentation of SAT-SOLVER

Anasuya Jana
Roll No: 194101004

Concept of Satisfiability:

Satisfiability problem is determining if there exists an assignment of truth values of variables of a given formula such that the formula evaluates to true. This Satisfiability problem is NP-complete and it can be used for solving any NP problems. Probably because of this reason SAT solvers become popular these days.

SAT Solver:

Given a boolean formula , a SAT solver determines whether the formula is satisfiable or not. If satisfiable then it returns assignment of truth values of variables. Here our consideration is to solve the problems which are in Conjunctive Normal form(CNF) as we can convert every formula to CNF in linear time.

Algorithm for SAT Solver:

Algorithms to test satisfiability that mean algorithms that are guaranteed to terminate and give a correct result on the given CNF(satisfiable/Not satisfiable). There are various approaches. First one is based on existential quantification. The second approach is to apply inference rules recursively until either a contradiction is found(Unsatisfiable CNF) or until the CNF becomes empty without contradiction(Satisfiable CNF).This approach also known as DPLL. The third approach is based on the search space of truth assignment.

I implement the DPLL Algorithm for SAT Solver.

DPLL Algorithm:

Formula in Conjunctive Normal form is: $(A \lor B \lor \neg C) \land (\neg A \lor D) \land (B \lor C \lor D)$
It can be expressed as:[[A, B, ¬C], [¬A, D], [B, C, D]]. This list based presentation is very helpful for the algorithm. In our algorithm CNF is satisfiable if CNF becomes an empty set and moreover,CNF is inconsistent if CNF contains empty clauses. If a CNF formula implies an empty clause then the formula is unsatisfiable.

Resolution:

Let P be a boolean variable and CNF contains clauses C1 and C2 such that P belongs to C1 and negation of P belongs to C2, Then according to resolution inference rule we can derive new clause (C1-{P}) ∪ (C2 − {¬P }) and we add this new clause to CNF. In resolution technique we assign true to P, then all the clauses where P belongs become true. So we remove those clauses from CNF. As we assign true to P, ¬P becomes false. So ¬P does not impact on the clauses. So we remove ¬P from the clauses. After applying resolution the size of CNF is always reduced as at least P will be removed from CNF. This guarantees that the algorithm will terminate. We apply resolution recursively until an empty clause will derive(Unsatisfiable) or until no more application of resolution possible.

1.{¬P, R}
2.{¬Q, R}
3.{¬R}
4. {P, Q}
5.{¬P } (1,3)
6.{¬Q} (2,3)
7.{Q}   (4,5)
8. {}   (6,7)
As an empty clause derived so the CNF is Unsatisfiable.

Unit_Clause:

Clause of length 1 is Known as unit_clause. We first find all the unit clauses from CNF. Then apply resolution on unit_clauses. If applying resolution leads to empty CNF that means CNF is satisfiable. So we return the Assignment. If CNF does not become empty and there are no remaining unit_clauses then we choose a decision variable from modified CNF and apply resolution. When we apply resolution on decision variables and we reach an empty clause that this assignment is inconsistent. So we backtrack and apply resolution on negation of that decision variable. After applying all the assignments of the variable if CNF implies any empty clause then we Declare that CNF is Unsatisfiable. Otherwise CNF becomes empty for any of the assignments and we return the assignment.

Basic Idea of Algorithms:

Apply Unit resolution as long as possible. If CNF becomes empty return Satisfiable and assignment list. If CNF implies any empty clause then return UNsatisfiable. Otherwise Choose a decision variable from CNF. apply unit resolution on that decision variable. If CNF becomes empty then return assignment list else backtrack and apply unit resolution on negation of the decision variable. This procedure will run recursively until cnf becomes empty or all the assignment of truth values exhausted.

dpll-(CNF Δ, depth d): returns a set of literals or unsatisfiable

```
1.if Δ = {} then
2.return {}
3.else if {} ∈ Δ then
4.return unsatisfiable
5.else if L =dpll-(Δ|P d+1 , d + 1)  = unsatisfiable then
6.return L ∪ {P d+1 }
7.else if L =dpll-(Δ|¬P d+1 , d + 1)  = unsatisfiable then
8.return L ∪ {¬P d+1 }
9.else
10.return unsatisfiable
```

Different heuristic techniques applied to choose decision variables.

1. Literal with maximum Count.
2. Weighted binaries heuristic
3. Backbone search heuristic
4. Maximum Difference heuristic.

Drawbacks:

Disadvantage of DPLL algorithm is huge space complexity as well as huge time complexity as it applies Chronological Backtracking. That is, if we try both values of a variable at level l, and each leads to a contradiction, we move to level l − 1, undoing all intermediate assignments in the process, and try another value at that level. Solution is Draw Implication graph and find cuts of that graph. Apply Backjump instead of backtracking everytime when contradictions occur. The approach Known as CDCL(conflict driven clause learning).
(Try to implement CDCL using the concept of implication graph and cut, but getting some errors. So implement DPLL)