50.001 2D Submission
Group No. 30

Daniel Low Yu Hian (1004372);
Sim Jia Ren (1004401);
Sean Gunawan (1004414);
Chan Jun Hern, Cawin (1004487);
Huang He (1004561)

October 26, 2020

Solver algorithm

Step 0: Check if the list of clauses is empty, if yes, returns the env mapping.

Step 1: Traverse through the list of clauses, trying to find a unit clause (that only contains one literal). In the meanwhile, keep track of the minimum size of clause (that contains the least number of literals).

If a unit clause is found, set the literal to True or False such that this clause is True, call *substitute*() to simplify the clauses, then, recursively call the solver with the new env mapping. Solver returns the env mapping if a solution is found and returns *null* if no solution is found.

The solver will proceed to Step 2 if no unit clause is found. But by now we have got the *smallestClauseSize* as an integer.

Step 2: We can get a clause that satisfy $clause.size() == smallestClauseSize$ by traverse through the list of clauses for the second time. We break out of the traversal loop once we find the first one that satisfies, it helps reduces some unnecessary running time.

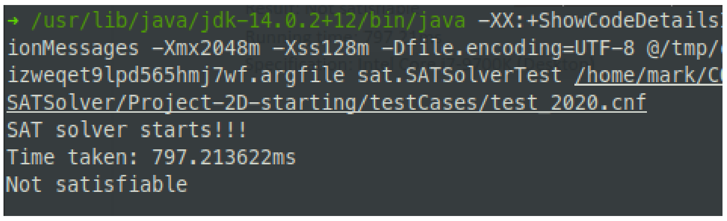
Step 3: Once we have picked one of the shortest clause, we simply pick the first literal from that clause, set the positive literal to True, call *substitute*() to simplify the clauses, then, recursively call the solver with the new env mapping. if the returned env is *null*, which signifies no solution, we get the negative literal and set it to False, call *substitute*() to simplify the clauses, then, recursively call the solver with the new env mapping.

In summary, the algorithm will try find a solution from setting unit clause to True. If there is no unit clause, it starts trying from the shortest clause first, until reaching the last clause. Whenever a solution is found, the program will return the env that contains the satisfiable solution. if no solution is found, the program will be done checking the last clause and return *null*.

Substitute algorithm

The simplification of clause list is done by *substitute*(), it is given a clause list and a literal, returns a new clause list after simplification assuming the literal is True. It simply traverse through the clause list, call the *reduce* function if the literal variable is present in the clause.

Test Result on test_2020



```
→ /usr/lib/java/jdk-14.0.2+12/bin/java -XX:+ShowCodeDetails
ionMessages -Xmx2048m -Xss128m -Dfile.encoding=UTF-8 @/tmp/
izweqet9lpd565hmj7wf.argfile sat.SATSolverTest /home/mark/C
SATSolver/Project-2D-starting/testCases/test_2020.cnf
SAT solver starts!!!
Time taken: 797.213622ms
Not satisfiable
```

Result: Not Satisfiable
Running Time: 797.21 ms
Machine Specs: Intel Core i7-9700K (Desktop) (System: Ubuntu 18.04) (JDK: 14.0.2)