SAT Solver

Report by: Rashmi G R(200772) and Sharath Kumar V(200916)

Implementation:

We have used a partial DPLL algorithm and optimised the runtime by using the 'Maximum Occurrence in clauses of Minimum Size' (MOMS) heuristic. By partial DPLL algorithm, we have only focused on the unit propagation and not the pure literal elimination. Our implementation outputs a model if it's satisfiable or it outputs as "UNSATISFIABLE".

Pseudo Code:

```
function choose-literal-moms-heuristic(Φ)
   compute clauses-of-minimum size
   return literal with most occurrences in clauses-of-minimum-size

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

Assumptions:

- We have assumed that we can use an external module to take input from the .cnf file. So we have used the pysat module to take input from the .cnf file.

- We have assumed that the code user would be able to change the test cases by editing the code.

NOTE: Instructions for both the installation of pysat and changing the test cases are given the Readme file

Limitations:

- The SAT solver takes about 3 minutes to solve a 150-variable formula in the cnf form.