CS 4244 Project – Stage 1

Due on the last day of classes
For Feedback, submit by March 15, 2020, 11:59 PM (GMT +8)

1 Objective

The goal of this project is to design a SAT solver based on CDCL algorithm to check satisfiability of CNF formulas.

Input Formulas should be represented in DIMACS format, see: `https://logic.pdmi.ras.ru/~basolver/dimacs.html`

Output Output "UNSAT" if the formula is unsatisfiable, otherwise output a satisfying assignment.

2 CDCL Procedure

Please see the description of the algorithm as Algorithm 1 on page 6 of `https://www.cis.upenn.edu/~alur/CIS673/sat-cdcl.pdf`.

Note that the description leaves with several design and implementation choices. In particular, there are two components where you need to come up with your own heuristics:

(1) **PickBranchingVariable**: You should choose a heuristic for choosing branching variable and assigning truth value. To evaluate the effectiveness of your heuristic you should compare them to the *random-choice heuristic*, in which all choices are resolved randomly with a uniform distribution. You should also compare your heuristic to the *2-clause heuristic*, in which we choose propositions with maximum occurrences in 2-clauses, i.e., clauses with two literals, and break ties randomly. I expect you to find a heuristic that would be better than the 2-clause heuristic.

(2) **ConflictAnalysis**: In the class, we discussed a basic version of conflict analysis, which was to create a new clause using decision variable and literals assigned at decision levels less than current level. I expect you to implement a slightly better heuristic.

Programming Language

You are free to choose your favorite programming language.

Advanced Resource

Once you have a basic SAT solver working, you can read description of how to design an efficient SAT solver at `http://minisat.se/downloads/MiniSat.pdf`. Note that the description is fairly advanced, therefore, I do not suggest you to be distracted by this material until have a basic functioning SAT solver.

3 Testing

You will evaluate the performance of your solver on random formulas generated according to the fixed-clause-length model. In this model there are three parameters: the number N of variables, the number K of distinct literals per clause, and the number L of clauses. To keep things simple, in this project we will take K to be three; that is, we will focus on the 3-SAT problem. You should aim at handling values of N that are at least 150 (though you may not be able to handle so many variables with the random-choice heuristics).

For a given N and L, an instance of random 3-SAT is produced by randomly generating L clauses of length three. Each clause is produced by randomly choosing a set of three distinct propositions from the set of N propositions and negating each one with probability 0.5. You should generate a set of formulas in advance, and then evaluate the performance of different implementations on this set of formulas. You can use CryptoMiniSAT to evaluate satisfiability or unsastisfability of generated formulas.

4 Eintein's Puzzle

Supposedly, Albert Einstein wrote this riddle, and said 98% of the people could not solve it. There are five houses in five different colors. In each house lives a man with a different nationality. The five owners drink a certain type of beverage, smoke a certain brand of cigar, and keep a certain pet. No owners have the same pet, smoke the same brand of cigar or drink the same beverage. **The question is: "Who owns the fish?"**

Hints

- The Brit lives in the red house.
- The Swede keeps dogs as pets.

- The Dane drinks tea.
- The green house is on the left of the white house.
- The green house's owner drinks coffee.
- The person who smokes Pall Mall rears birds.
- The owner of the yellow house smokes Dunhill.
- The man living in the center house drinks milk.
- The Norwegian lives in the first house.
- The man who smokes Blends lives next to the one who keeps cats.
- The man who keeps the horse lives next to the man who smokes Dunhill.
- The owner who smokes Bluemasters drinks beer.
- The German smokes Prince.
- The Norwegian lives next to the blue house.
- The man who smokes Blends has a neighbor who drinks water.

In order to solve the puzzle, you have to make some assumptions: 1. The owner is the resident of each house. 2. One of the residents owns the fish. 3. The term neighbor in the last hint refers only to a directly adjacent neighbor. 4. The houses are on the same side of the street. 5. They are next to each other, and are ordered from left to right as you face them rather than standing in front of a house facing the street (i.e. facing the same direction as the house). You need to solve this puzzle by expressing it first as a CNF formula such that the satisfying assignment of the formula correspond to possible solutions of the puzzle

5 Final Submission

You will measure the performance of your solver by counting both total compute time as well as the total number of invocations of the **PickBranching Variable** subroutine required to find whether the input formula is satisfiable or not.

Your final submission should include:

1. The source code along with verbal explanation of your design choices. Remember, when you are working as a software engineer, your code will never make to production if your peers can not review it.

2. In addition, you should submit a written report of at most 10 pages (excluding references), including:

   (a) CNF encoding of Einstein's puzzle along with the answer.

   (b) You should include a written report of no with a verbal summary and an analysis of your findings. Your analysis can be intuitive, but you need to come up with an explanation of your findings.

   (c) I believe that you will be a leader in your field in a few years. As a leader, you are expected to evaluate yourself. you should include what letter grade would you have assigned to yourself if you were the instructor. Finally, your report should end with a paragraph explaining what you learned from this project.