Homework 3: Conflict Analysis Algorithms for Intelligent Decision Making (CS4210-A)

Maarten Flippo, Konstantin Sidorov, Emir Demirović

Feb 27, 2023

1 Assignment

Your task is to implement a conflict analysis procedure into a SAT solver. You are expected to correctly implement 1-UIP (Universal Implication Point) learning in the skeleton that is provided to you. The solver should be able to solve the test instances within a few seconds, and provide correct output.

The solver you will be working with is called `Pumpkin`, a prototype SAT/CP solver developed in the Algorithmics group at TU Delft. It is written in the programming language Rust. In case you do not have much experience with Rust, you may familiarize yourself with the language via "The Rust Programming Language" book. For the purposes of this course, it is sufficient to have a good command of **its first nine chapters**, up to the "Error handling" chapter.

Implement the function `ConstraintSatisfactionSolver::analyse_conflict`, which lives in the file `src/engine/constraint_satisfaction_solver.rs`. From the method signature you can see the function should return a `ConflictAnalysisResult`, which combines the learnt clause and the decision level to backjump to.

Note 1: The solver expects the asserting literal to be the *0-th* index literal in `ConflictAnalysisResult::learned_literals`. The *1-st* index literal literal should be another literal with the highest decision level. These invariants are required by the two-watch literal scheme.

At the bottom of the file you will work in, there is a unit-test which verifies the learning performed is actually 1-UIP. You can also use this test to debug any issues you might run in to.

2 Submission

Upload `src/engine/constraint_satisfaction_solver.rs` to WebLab. Do not change the unit test in that file, only work on the method mentioned above. You may work in pairs.

3 Grading

In the ZIP file that contains the solver code, there is a folder called `instances`. This folder contains DIMACS files which will be used to test the correctness of your implementation. There are satisfiable and unsatisfiable instances, and we expect the solver output to match what is expected for each file.

If the correct output is produced for all test instances, full points will be awarded. In **any other situation**, zero points will be awarded. Each successful homework contributes 1% towards your grade.

Please submit the homework by the end of March 6 (Monday). Late submissions will **not** accepted.

4 Questions

For questions and issues that may arise, please contact the TAs:

- Imko (I.C.W.M.Marijnissen@student.tudelft.nl)
- Ana (A.Tatabitovska@student.tudelft.nl)

- Alin (A.E.Dondera@student.tudelft.nl)