Exercises for the Lecture Practical SAT Solving | Summer Semester 2016 | Assignment 5

Institute for Theoretical Informatics, Karlsruhe Institute of Technology (KIT)
Tomas Balyo, Carsten Sinz

30.06.2016

Due 14.07.2016

Exercise 1 (Coloring by Incremental SAT Challenge) [7(+7) points]

Implement an incremental SAT based graph vertex coloring tool. Your application should take a single command line argument – a DIMACS file with a graph and find the smallest number of colors needed to color the graph. The application should print the number of colors required and the color of each vertex.

For the technical details of implementing an incremental solver based application follow the benchmark submission instructions of the 2016 SAT Competition Incremental Track: `http://baldur.iti.kit.edu/sat-competition-2016/index.php?cat=incremental`.

Test instances available at `http://mat.gsia.cmu.edu/COLOR/instances.html` (use the *.col instances, the *.col.b are in binary format).

Exercise 2 (Sliding Puzzle by Incremental SAT Challenge) [10(+10) points]

Implement an incremental SAT based sliding puzzle solver. Your application should take a single command line argument – a string with the initial state of the puzzle (example: "123-480-765", goal state is always "123-456-780", "0" represents the empty slot) and print the sequence of steps required to solve it (movements of the empty slot, in the example above: "DLURD"). The puzzle will have at most 15 pieces (0123456789ABCDEF) and is not necessarily a square.

Some example inputs for testing: "023-145", "1034-9267-D5B8-E6FC", "162-530-478", "012-345-678", "876-543-210", you can create your own inputs, be careful not to create unsatisfiable ones :)

For technical details of incremental SAT see Exercise 1.

Exercise 3 (DRUP and DRAT Proofs) [4 points]

Generate proofs in DRUP and DRAT format for the formula $(x_3 \vee x_4 \vee \bar{x}_1 \vee x_5) \wedge (\bar{x}_3 \vee x_4 \vee x_5) \wedge (x_3 \vee \bar{x}_4 \vee \bar{x}_1) \wedge (x_1 \vee x_2) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_5) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee x_5)$

Exercise 4 (Core-Guided MaxSAT solving) [4 points]

Solve the formula form exercise 3 as a MaxSAT problem using the Core-Guided MaxSAT algorithm from Lecture 11.

Exercise 5 (Weighted Partial MaxSAT to MaxSAT) [4 points]

Show that any Weighted Partial MaxSAT problem instance (with positive integer weights) can be translated into a MaxSAT problem instance.

Exercise 6 (Longest Path as Weighted Partial MaxSAT) [6 points]

Describe how to encode the Longest Path Problem `https://en.wikipedia.org/wiki/Longest_path_problem` for a graph with positive edge weights as a Weighted Partial MaxSAT problem.