Exercises for the Lecture Practical SAT Solving | Summer Semester 2016 | Assignment 3

Institute for Theoretical Informatics, Karlsruhe Institute of Technology (KIT) 2.06.2016

Tomas Balyo, Carsten Sinz Due 16.06.2016

Exercise 1 (CDCL) [5 points]

Simulate CDCL (from Slide 20 of Lecture 6 slides) by hand on the formula below. Select branching literals in the order $x_1, x_2, x_3, \ldots$ Draw the implication graph for each conflict and learn the 1-UIP clause.

$(x_3 \vee x_4 \vee \bar{x}_1 \vee x_5) \wedge (\bar{x}_3 \vee x_4 \vee x_5) \wedge (x_3 \vee \bar{x}_4 \vee \bar{x}_1) \wedge (x_1 \vee x_2) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_5) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee x_5)$

Exercise 2 (Pythagorean Triples) [3 points]

Design an encoding for the Pythagorean Triples Coloring problem: For a given n is it possible to assign each integer $1 \ldots n$ one of two colors such that if $a^2 + b^2 = c^2$ then a, b and c do not all have the same color. Estimate the number of variables and clauses in your encoding (as a function of n).

Exercise 3 (Tseitin Encoding) [4 points]

Encode the following formula into CNF using the Tseitin Encoding.

$(\overline{x_1} \wedge \overline{(x_3 \iff x_2)}) \vee ((x_3 \implies \overline{x_4}) \wedge (x_1 \implies (x_2 \wedge \overline{x_3})) \wedge (x_4))$

Exercise 4 (Unit Propagation Challenge) [8(+10) points]

Implement a formula preprocessor that does unit propagation. Given a CNF formula F in the DIMACS format (`http://www.satcompetition.org/2004/format-solvers2004.html`) containing some unit clauses output F after unit propagation also in the DIMACS format. For a working preprocessor you get 8 points. The author of the fastest implementation receives a bonus of 10 points.

Exercise 5 (Hidden Horn $\subseteq$ SLUR) [10 points]

Prove that every Hidden Horn formula is a SLUR formula. A formula is SLUR (Single Lookeahead Unit Resolution) if it can be solved by the following algorithm (it never gives up regardless of the selection of the branching literal).

```
if unitProp(F) == conflict then return UNSAT else return SLUR(F)

function SLUR(F)
  if hasNoClauses(F) then return SAT
  lit l = selectLiteral(F)
  Fp = unitProp(F+(l))
  Fn = unitProp(F+(-l))
  if Fn == conflict and Fp == conflict then return GIVEUP
  if Fn == conflict then return SLUR(Fp) else return SLUR(Fn)
```