Artificial Intelligence in Theorem Proving

Homework 1

Please send your solutions to `bartosz@mimuw.edu.pl` by **25.04.2019**.

Davis-Putnam algorithm This algorithm takes as its input a propositional formula in clausal form (CNF as set of sets of literals) and returns either SAT or UNSAT depending on satisfiability of the formula. The algorithm consists of the following rules:

(1) *the unit clause rule*: if there is some clause consisting of a single literal l (a *unit clause*), remove instances of $-l$ from other clauses and then remove any clauses containing l (including the unit clause itself);

(2) *the pure literal rule*: if there is a literal appearing in the formula *only positively* or *only negatively*, remove all clauses containing it;

(3) *the tautology rule*: if some clause contains complementary literals, l and $-l$, remove this clause;

(4) *the resolution rule*: choose a literal l and split the set of clauses S into a set of clauses containing l only positively, a set of clauses containing l only negatively, and the rest:

$$\begin{aligned} S_1 &= \{C \in S \colon l \in C \text{ and } -l \notin C\} \\ S_2 &= \{C \in S \colon -l \in C \text{ and } l \notin C\} \\ S_3 &= S \setminus (S_1 \cup S_2). \end{aligned}$$

Then the output set of clauses is defined as

$$S' = \{(C_1 \cup C_2) \setminus \{l, -l\}) \colon C_1 \in S_1, C_2 \in S_2\} \cup S_3 \tag{1}$$

The DP algorithm terminates either if the set of caluses is empty, returning SAT, or if the set of clauses contains an empty clause, returning UNSAT. Otherwise it applies the first of the rules (1), (2), (3), (4) to succeed and then continues recursively on the new set of clauses.

Davis–Putnam–Logemann–Loveland algorithm This is a modification of the DP algorithm. Instead of the resolution rule there is

(4') *the splitting rule*: having set of clauses S choose some literal l appearing in S and test satisfiability for $S \cup \{\{l\}\}$ and $S \cup \{\{-l\}\}$.

Exercise 1. (*6 points*) Implement DP and DPLL procedures. Propose some heuristic for choosing literals in rules (4) and (4'). The implementations should be programs taking one argument: a path to a file with a SAT problem in CNF DIMACS format[1] and returning either SAT or UNSAT. Test these procedures on several different inputs (for instance taken from SATLIB[2]) and compare time and memory usage of DP and DPLL procedures on these problems.

Ramsey theorem For each $s, t \in \mathbb{N}$ there is some $n \in \mathbb{N}$ such that any graph with n vertices either has a completely connected subgraph of size s or a completely disconnected subgraph of size t. Moreover, if the *Ramsey number* $R(s, t)$ denotes the minimal such n for a given s and t we have:

$$R(s, t) \leq R(s - 1, t) + R(s, t - 1).$$

Exercise 2. (*6 points*) Formulate a propositional formula $\phi_{s,t,n}$ such that it is a taulology iff $R(s, t) \leq n$. Using this formula (negated and transformed to CNF[3] and one of the available SAT solvers try to find Ramsey numbers $R(s, t)$ for $s, t \leq 7$. Try also to do this with the DP and DPLL procedures implemented in Exercise 1.

The solutions should consist of an implementation and a concise report.

[1] `https://logic.pdmi.ras.ru/~basolver/dimacs.html`

[2] `https://www.cs.ubc.ca/~hoos/SATLIB/benchm.html`

[3] For transformation to CNF you can use one of the available tools, e.g. `https://docs.sympy.org/0.7.6/modules/logic.html#sympy.logic.boolalg.to_cnf`