

COMP4415 Assignment 1 – 2003

March 12, 2003

Assignments will contain a set of problems of various degrees of complexity. To pass, you need to solve at least the EASY problems. Some of the problems may be very hard indeed, so just do your best and submit what you can. If you have some ideas of how to solve a problem, but no complete solution, you may get some credit for a good explanation of your idea.

1. (EASY, but its recommended that you do some simple resolution problems from the book before you attempt it.) Use resolution to show that a triangle is not 2-colourable, as follows:
 - (a) Let the vertices of the triangle be called A, B, C and the colours be black and white, represented by b, w . Use propositions of the form XY , where A is a vertex and Y is a colour. For example, the proposition Aw represents that A is white. Write a set of clauses that expresses that the triangle has been 2-coloured. Break your set into groups and explain what intuitive constraint each group is representing.
 - (b) Now use resolution to show that this set of clauses is not satisfiable. Draw your proof as a proof tree. (Hint: first convince yourself by intuitive reasoning that the triangle cannot be 2-coloured. Now try to mirror the proof using resolution.)
2. (HARDER) Call a clause a *2-clause* if it contains at most two literals.
 - (a) Prove that if S is a set of 2-clauses and $S \vdash_R C$ then C is a 2-clause.
 - (b) Using the fact proved in the previous part, design an algorithm that takes as input a set S of 2-clauses, and
 - returns “yes” if this set is satisfiable, and “no” if this set is not satisfiable.
 - always halts in time $p(|S|)$, where p is some polynomial, and $|S|$ is the number of symbols required to write out S .

It is enough to present your algorithm in psuedo-code, and detailed design of data-structures is not required (you can assume a set-data type if you like). Explain why your algorithm produces the correct output, and why it halts in a polynomial amount of time.

3. (HARDEST) Call a clause an m -clause if it contains at most m literals. Given a set S of clauses and a clause C , say that a resolution proof $C_1, \dots, C_n = C$ of C from S is m -bounded if C_i is an m -clause for all $i = 1 \dots n$. (That is, the proof does not need clauses containing more than m literals.) It follows from the first part of the previous question that if S an unsatisfiable set of 2-clauses then there exists a 2-bounded proof of the empty clause from S . What is the smallest number m (as a function of k) for which you can prove that if S is an unsatisfiable set of 3-clauses then there exists an m -bounded proof of this fact? Show by means of an example that you cannot do better than the number you are claiming.