KTH Computer Science and Communication

Homework VII, Complexity Theory Fall 2011

Due on October 21 at 17.00. To be put into the mail box of Johan Håstad which is located on level 4, Lindstedtsvägen 3 (enter through the doors with the label "NADA", after going a small distance to the left proceed straight and you find the department mail boxes on your right).

The general rules on homework solutions available at the course home-page apply. In particular, discussions of ideas in groups of up to at most two people are allowed but solutions should be written down individually.

Some of the problems below are "classical" and hence their solutions is probably posted on the Internet. It is not allowed to use such solutions in any way. The order of the problems is "random" and hence do not expect that the lowest numbered problems are the easiest.

Any corrections or clarifications on this problem set will be posted under "homework" on the course home-page http://www.csc.kth.se/utbildning/kth/kurser/DD2446/kplx11/uppgifter.

1 (15p) This is information gathering problem and should be solved *individually*. One more time we study 3-Sat but this time in an average time complexity setting. You are only supposed to supply references (journal articles or text book sources) to the answers, not motivations or proofs.

Suppose we have n Boolean variables, then we construct an instance by picking m random clauses of size 3 independently. A random clause is constructed by picking 3 literals randomly with the constraint that no two literals correspond to the same variable. Clearly, the larger value of m the less likely it is that the formula is satisfiable as we are adding more constraints.

1a (3p) For what values of m is it known that formula is satisfiable?

1b (3p) For what values of m is it known that formula is not satisfiable?

1c (3p) What value of m is the conjectured threshold of satisfiability?

1d (3p) For what values of m is there an efficient (expected polynomial time) algorithm that finds a satisfying assignment?

1e (3p) For what values of m is there an efficient (expected polynomial time) algorithm that with high probability finds a proof that the formula is not satisfiable?

2 (10p) In class we constructed a circuit of size $n2^n$ for an arbitrary function $f : \{0,1\}^n \mapsto \{0,1\}$. This is not optimal.

2a (3p) Show how to construct a circuit of size $O(2^n)$ for any function f.

2b (7p) Show how to construct a circuit of size $O(2^n/n)$ for any function f.

Hint: For a suitable d, first use some circuitry to compute all functions of the first d variables. Now use, more or less, the original construction on the remaining $n - d$ variables. This time not to pick an output but rather a suitable function of the first d variables.

If you solve the second problem correctly, there is no need to solve the first problem for a full score.

3 (20p) A branching program is a directed acyclic graph where a typical node is labeled with a variable, x_i, for some $1 \leq i \leq n$, and has out-degree two where one outgoing edge is labeled "0" and the other is labeled "1". Each node that does not have out-degree 2, has out-degree 0 and is labeled by a constant (0 or 1). There is also a designated start node.

A branching program computes a Boolean function $\{0,1\}^n \mapsto \{0,1\}$ in a natural way. Namely, start at the start node and whenever we arrive at a node labeled with a variable x_i we follow the outgoing edge labeled 0 if $x_i = 0$ and otherwise we follow the edge labeled 1. When we arrive at node with out-degree 0 the label of that node is the output. The size of branching program is the number of nodes.

As with circuits, a branching program only works for a fixed input size and thus a language in the "usual" sense is recognized by a sequence of branching programs, one for each size, n, of the input.

3a (10p) Prove that if $A \in \mathrm{L}$ then A can be recognized by a sequence of branching program of polynomial size.

3b (5p) Is the converse true? I.e. if A can be recognized by a sequence of polynomial size branching programs, then can we conclude that $A \in \mathrm{L}$? Maybe if you add a natural condition?

3c (5p) It is not hard to see that any function of n bits can be computed by a branching program. What is the minimal size you need to compute an arbitrary function?