CS764 — Computational Complexity — Spring 2014

Revised version: changes in italics

Basic problems (required): submit solutions by *Monday, June 2.*
Advanced problems (optional): submit solutions by the end of lectures.

Basic problems

These problem can be solved using the results and techniques discussed in class, and some bits of basic mathematics.

1. Let FVAL be the language of Boolean expressions that evaluate to 1. Here an expression has only constants (0 and 1), connectives ($\wedge$, $\vee$ and $\neg$) and parentheses.

   Show that FVAL $\in SPACE(\log n)$.

2. For each integer $k \geq 2$, show that there are circuits to compute parity that

   - use only AND and OR gates (with arbitary in-degree),
   - *have negated inputs available (e.g., $\neg x_i$ is an available input bit, for each i),*
   - have depth k, and
   - have size $2^{O(n^{1/(k-1)})}$ for inputs of length n.

   For example, the case $k = 2$ asks to represent parity in either either sum-of-products or product-of-sums form, using $2^{O(n)}$ minterms or maxterms.

   (The parity problem: output 1 iff the number of 1s in the input is even.)

3. Fix a function f that satisfies $f(n+1) > f(n) > n$ for every n. Assume that the value of $f(n)$ can be computed from n in time proportional to $f(n)$, or less.

   For every string x, let $expand_f(x)$ denote the string $x01^{f(|x|)-|x|-1}$; that is, x *expanded to length* $f(|x|)$ by the necessary number of 1s (with a separator).

   For a language A, let $expand_f(A)$ denote the set

   $$expand_f(A) = \{\, expand_f(x) \mid x \in A \,\} \quad .$$

   (a) Suppose that $expand_f(A) \in TIME(t(n))$. Show that $A \in TIME(t(f(n)))$.
   ***Note**: the original version of the problem asked for $A \in TIME(f(t(n)))$. The current $A \in TIME(t(f(n)))$ is correct.*

   (b) Show that if $TIME(n) = NTIME(n)$ then $TIME(f(n)) = NTIME(f(n))$.
   (One can prove that $TIME(n) \neq NTIME(n)$—that, is, the hypothesis fails—but that is definitely the hard way to solve this problem. Using the notion of expansion makes it much easier.)
   (Note: the revision to the definition of "$expand_f$" does not affect the problem in a significant way—it just makes some of the notation simpler in a solution. Use the original formulation if you wish.)

Advanced problems

These problems require techniques not discussed in class, and/or some inventiveness—but solutions lie within reach.

4. Give an algorithm for a Turing machine whose space usage is $O(\log \log n)$, but not constant. The language computed doesn't matter; just focus on the space used.

   (Hints: (a) By the next problem, the solution must use its input tape heavily. (b) The actual space used need not be the same for every input of length n; it may even be constant for some of them—but not all.)

5. (a) Suppose that a language A can be accepted in space $o(\log n)$ by a Turing machine M that never moves its input head to the left. Show that M must use only constant space (which implies that A is regular).

   (b) Let $s(n) = o(\log \log n)$. Show that $SPACE\big(s(n)\big)$ is the set of regular languages.

6. Write an algorithm that has the following properties.

   - When the input is a satisfiable Boolean formula, the algorithm produces a satisfying assignment. Otherwise, it rejects.
   - If $P = NP$, the algorithm runs in polynomial time on satisfiable formulas (but may use exponential time on unsatisfiable formulas).

   Hint: If $P = NP$, then some algorithm exists, but you must do more: show that, if $P = NP$, then your particular algorithm has a polynomially bounded run time on satisfiable formulas.

7. Let B be accepted by a one-tape Turing machine in time t. Show that $B \in SPACE(\sqrt{t})$.

   (Hint: Divide the tape of the one-tape machine into blocks of size $\sqrt{t}$, and simulate each block as independently as possible.)