**CS 151 Complexity Theory** **Spring 2017**

# Midterm

*Out: May 3* *Due:* **May 10 at the beginning of class**

This is a midterm. Collaboration is not allowed. You may consult the course notes and the text (Papadimitriou), but not any other source or person. The full honor code guidelines can be found in the course syllabus.

Please attempt all problems. **To facilitate grading, please turn in each problem on a separate sheet of paper and put your name on each sheet. Do not staple the separate sheets.**

1. Show that **coNEXP** $\subseteq$ **NEXP**/$(n+1)$. Here the "/$(n+1)$" means that the nondeterministic machine takes exactly $(n+1)$ bits of advice on an input of length $n$. Hint: use an idea similar to one you used for problem 2 on Problem Set 2.

2. Let $f : \{0,1\}^n \to \{0,1\}$ be an arbitrary function, and consider the following scenario involving two parties, Alice and Bob. Alice is given an input $x$ for which $f(x) = 0$ and Bob is given an input $y$ for which $f(y) = 1$. They take turns sending bits to each other, and at the end of the protocol they must announce an index $i$ between 1 and $n$ on which $x$ and $y$ differ, i.e., $x_i \neq y_i$. Formally, in the first round Alice sends $A_1(x) = a_1$ to Bob; Bob sends $B_1(y, a_1) = b_1$ to Alice; Alice sends $A_2(x, b_1) = a_2$; Bob sends $B_2(x, a_1, a_2) = b_2$; Alice sends $A_3(x, b_1, b_2) = a_3$; and so on. In odd steps Alice sends a message that depends on her input $x$ and the messages she has received so far; in even steps Bob sends a message the depends on his input $y$ and the messages he has received so far. In the end, after $k$ rounds Alice computes $R_A(x, b_1, b_2, \ldots, b_k)$ and Bob computes $R_B(y, a_1, a_2, \ldots, a_k)$, and these final function evaluations should both produce the desired index $i$, on which $x_i \neq y_i$. The protocol must work for all pairs of inputs $x \in f^{-1}(0)$ and $y \in f^{-1}(1)$; the functions $A_i$ and $B_i$ together with $R_A$ and $R_B$ define a *protocol for* $f$.

   The *communication complexity* for $f$, denoted $C(f)$, is the minimum, over all protocols for $f$, of the number of bits exchanged during the protocol. Let $D(f)$ denote the minimum, over all fan-in two $(\wedge, \vee, \neg)$ Boolean circuits that compute $f$, of the depth of the circuit. Below you will prove the startling fact that these two quantities are essentially the same!

   (a) Show that $C(f) \leq c_1 D(f)$, where $c_1$ is a constant that does not depend on $f$. Hint: use induction on the depth of a minimum-depth circuit for $f$.

   (b) Show that $D(f) \leq c_2 C(f)$, where $c_2$ is a constant that does not depend on $f$. Hint: prove a stronger statement as follows. For every set $X \subseteq f^{-1}(0)$ and $Y \subseteq f^{-1}(1)$ we say that a *protocol for* $f$ *on* $X, Y$ is a protocol that is only required to work on input pairs $x \in X$ and $y \in Y$ (so a *protocol for* $f$ as defined above is a protocol for $f$ on $f^{-1}(0), f^{-1}(1)$). Define $C_{X,Y}(f)$ to be the minimum, over all protocols for $f$ on $X, Y$, of the number of bits exchanged during the protocol. Prove that for all $X \subseteq f^{-1}(0)$ and

$Y \subseteq f^{-1}(1)$ there is a circuit with depth at most $c_2 C_{X,Y}(f)$ that outputs 0 on inputs $x \in X$ and 1 on inputs $y \in Y$.

3. A *branching program* is a directed acyclic graph with three distinguished nodes, called *start*, *accept*, and *reject*. Every node except *accept* and *reject* is labeled by a positive integer $i$, and has exactly two outgoing edges, one labeled "0" and the other labeled "1". An input $x = x_1 x_2 \ldots x_n$ defines a path from the start node as follows: at a node labeled $i$, we follow the outgoing edge whose label coincides with bit $x_i$ in the input. The path terminates at a sink node (which is either *accept* or *reject*) and the input is accepted or rejected accordingly.

   Recall that **L/poly** is the class of languages decidable by a Turing machine in $O(\log n)$ space with poly$(n)$ bits of advice. Show that **L/poly** is exactly the class of languages decided by polynomial-size branching programs.

4. Show that $\mathbf{NP} \subseteq \mathbf{BPP}$ implies $\mathbf{NP} = \mathbf{RP}$. Hint: first use error reduction to reduce the error probability of the **BPP** machine.

5. (a) Let $f$ be a family of one-way permutations, and let $b = \{b_n\}$ be a hard bit for $f^{-1}$. Assume that both $f$ and $b$ are computable in polynomial time. Use $f$ and $b$ to describe a language $L$ for which $L \in (\mathbf{NP} \cap \mathbf{coNP}) - \mathbf{BPP}$.

   (This shows that the assumption we used to construct the BMY pseudo-random generator placed *a priori* bounds on the power of **BPP** – it presumed that **BPP** was not powerful enough to simulate $\mathbf{NP} \cap \mathbf{coNP}$.)

   (b) Fix a constant $\delta$, and let $g = \{g_n\}$ be a uniform family of functions for which:

   - each $g_n$ maps $t = O(\log n)$ bits to $m = n^\delta$ bits, and is computable in poly$(n)$ time, and
   - for all circuits $C : \{0,1\}^m \to \{0,1\}$ of size at most $m$,

$$\left| \Pr_{y \in \{0,1\}^m}[C(y) = 1] - \Pr_{z \in \{0,1\}^t}[C(g_n(z)) = 1] \right| < 1/6.$$

   Use $g$ to describe a language $L \in \mathbf{E}$ which does not have circuits of size $2^{\epsilon n}$, for some constant $\epsilon > 0$. Hint: refer to a function family obtained by truncating the output of $g$ to $t + 1$ bits.

   (Notice that $g$ is a "Nisan-Wigderson style" pseudo-random generator, which we were able to construct based on the assumption that there is some language in **E** that does not have circuits of size $2^{\epsilon n}$ for some constant $\epsilon$. This problem shows that this assumption is also *necessary* for the existence of such generators.)