CS 221: Computational Complexity

Problem Set 6 (Take-Home Final)

Assigned: Wed. Apr. 21, 2010 Due: Thu. May. 6, 2010 (5 PM)

Exam Policies. You must work on this exam *alone*. The only references you may use are notes from lecture and section, the problems sets and solutions, and the Arora–Barak . You may quote any result *proven* in class or in the text. Except on Problem 1, you may not use results that were only stated without proof (unless you provide justification of your own). No late days are permitted, and no credit will be given for work turned in after the deadline.

You must *type* your solutions. LaTeX, Microsoft Word, and plain ascii are all acceptable. Submit your solutions *via email* to `cs221-hw@seas.harvard.edu`. If you use LaTeX, please submit both the compiled file (`.pdf`) and the source (`.tex`). Please name your files `PS6-yourlastname.*`.

Problem 1. (Various relations) In this problem, you will describe everything that we know (from this course) about the pairwise relationships between the following statements. In doing so, you may use results stated without proof in class or in the texts. First, identify whether any of the statements below are known to be true or false. For the remaining statements, draw a directed graph showing all of the implications that follow from the results we have seen in class. (You do not need to consider negations of the statements. And if you have implications $A \Rightarrow B$ and $B \Rightarrow C$, you do not need to draw the implication $A \Rightarrow C$.) Briefly justify your answers.

1. Every function computable in exponential time can be computed in probabilistic polynomial time on all but a $1/n^{\log n}$ fractions of the inputs of length n, for all n.
2. The polynomial-time hierarchy collapses.
3. $\mathbf{NTIME}(n^2) \neq \mathbf{TIME}(n^4)$.
4. TQBF can be solved by logspace-uniform, polynomial-size boolean formulas.
5. $\mathbf{AM} = \mathbf{coAM}$.
6. Unique SAT is in **prP**.
7. There is a fully polynomial almost-uniform sampler for Independent Sets.

Problem 2. (2-CSPs and 2-Query PCPs)

1. Give a (1/4)-approximation algorithm for Max-2CSP.
2. Show that there are constants $1 > c > s > 0$ such that $\text{Gap}_{c,s}$Max-2SAT is **NP**-hard.
   [Hint: consider the gadget
   $(x_1 \vee x_2 \vee x_3) \mapsto (x_1) \wedge (x_2) \wedge (x_3) \wedge (\neg x_1 \vee \neg x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3 \vee \neg x_1) \wedge (y) \wedge (x_1 \vee \neg y) \wedge (x_2 \vee \neg y) \wedge (x_3 \vee \neg y)$.]

3. Prove that for every $s < c/4$, $\mathbf{PCP}_{c,s}(\log n, 2) = \mathbf{P}$.

4. Prove that for some $s < c$, $\mathbf{PCP}_{c,s}(\log n, 2) = \mathbf{NP}$.

5. Prove that for every $s < 1$, $\mathbf{PCP}_{1,s}(\log n, 2) = \mathbf{P}$.

Problem 3. (Testing Graph Properties) A graph $G = (V, E)$ is a *biclique* if there is a partition (V_1, V_2) of V such that $E = V_1 \times V_2$. Consider the following promise problem:

$$\begin{aligned}(\textsc{Test}_\varepsilon\textsc{BiClique})_Y &= \{G : G \text{ is a biclique}\}.\\ (\textsc{Test}_\varepsilon\textsc{BiClique})_N &= \{G : G \text{ is } \varepsilon\text{-far from every biclique}\},\end{aligned}$$

where we represent the graph G by its adjacency matrix, consisting of $N = n^2$ bits, and we say that two graphs are *ε-far* if their adjacency matrices differ on at least $\varepsilon \cdot N$ entries.

1. Show that for every constant $\varepsilon > 0$, $\textsc{Test}_\varepsilon\textsc{BiClique}$ is in $\mathbf{prBPTIME}(O(\log N))$, if we work in a computational model where the algorithm has random access (equivalently, oracle access) to its input. (Hint: fix an arbitrary vertex $v_0 \in V$, randomly pick $u, w \xleftarrow{R} V$, and check which of the edges (v_0, u), (v_0, w), and (u, w) are in G.)

2. Show that for some constant $\varepsilon > 0$, $\textsc{Test}_\varepsilon\textsc{BiClique}$ is not in $\mathbf{prDTIME}(o(N))$, if we work in a computational model where the algorithm has random access (equivalently, oracle access) to its input.

Thus, even though there is evidence that $\mathbf{BPP} = \mathbf{P}$ (and also $\mathbf{prBPP} = \mathbf{prP}$), when we consider *sublinear time*, randomization provably provides an exponential savings over deterministic algorithms.

Problem 4. (Depth Reduction for AlgP/poly) A polynomial $p(x_1, \ldots, x_n)$ over a field $\mathbb{F}$ is *homogeneous of degree d* if every nonzero term in p has degree exactly d. (Hence the zero polynomial is homogeneous of every degree.) Rvery polynomial $p(x_1, \ldots, x_n)$ of degree at most d can be written uniquely as $p = p_0 + \cdots + p_d$, where p_i is homogeneous of degree i; p_i is called the *homogeneous degree i part* of p. An algebraic circuit C is *homogeneous* if every gate in C computes a homogeneous polynomial. Below you may assume $\mathbb{F} = \mathbb{Q}$, but the results hold over any field.

1. Show that if $p(x_1, \ldots, x_n)$ is a degree d polynomial computable by an algebraic circuit of size s, then there is a a homogeneous algebraic circuit of size $O(d^2 s)$ computing each of the homogeneous parts of $p(x_1, \ldots, x_n)$.

2. Show that if $p(x_1, \ldots, x_n)$ is a (homogeneous) degree d polynomial (for $d \geq 2$) computable by a homogeneous algebraic circuit of size s, then $p = \sum_{i=1}^{s} q_i r_i$, where each of the q_i's and r_i's are homogeneous polynomials of degree at most $2d/3$ computable by homogeneous algebraic circuits of size $O(s)$. (Hint: write $p = p_0 + p_1 q + p_2 q^2$, where q is the polynomial computed by an appropriately chosen gate in the circuit, and induct on s.)

3. Deduce that if $p(x_1, \ldots, x_n)$ is a degree d polynomial computable by an algebraic circuit of size s, then p is computable by an algebraic circuit of depth $O((\log d) \cdot (\log s + \log d) + \log n)$.