15-855: An Introduction to Computational Complexity Theory Fall 2015
Problem Set #1 V. Guruswami & M. Wootters
Due on **September 24th, 2015** (in class, or by email to yuzhao1@cs.cmu.edu *before* class).

Instructions: You are allowed to collaborate with up to two other fellow students taking the class in solving problem sets, but you *must write up your solutions on your own.* If you do collaborate in solving problems, please acknowledge your collaborators for each problem. We also require you to think about *each* problem by yourself for **at least 30 minutes** before any collaboration on that problem.

We expect you to solve the problems **without looking for hints or solution approaches in any external sources** (in particular you should not attempt to look up solutions on the web). If you are stuck, you may ask the instructors or TA for help, either during office hours or on Piazza, clearly articulating the difficulty you are having. If for some reason you end up having to look up some material related to the problem, you **must** acknowledge the source, and also highlight what key insight you learned from the source that you were missing in your own attempts at the problem.

Most of the problems require only one or two key ideas for their solution – spelling out these ideas correctly should give you some partial credit for the problem even if you err in some finer details. So, make sure you clearly write down the main idea(s) behind your solution even if you could not figure out a complete solution.

We prefer (but do not require) solutions typeset in LaTeX. If you are handwriting your solutions, please write clearly and legibly, and turn in a polished version.

You are strongly urged to start early and make good use of the two week timeframe, as some of the problems may require a fair bit of thought.

Pick any 6 problems to solve out of the 8 problems. *If you turn in solutions to more than 6 problems, we will take your top 6 scoring problems.*

1. A bipartite graph is a graph whose vertices can be partitioned into two disjoint parts each of which is an independent set. Formally a bipartite graph $H = (X, Y, E)$ has vertex set $X \cup Y$ for disjoint sets X, Y and each edge in its edge set E has one endpoint in X and one in Y. A k-bipartite clique of H is a pair of subsets $S \subseteq X$ and $T \subseteq Y$ with $|S| = |T| = k$ such that $(s, t) \in E$ for each $s \in S$ and $t \in T$ (informally all "cross-edges" exist between S and T). Define the language

   $$\text{BIPARTITE-CLIQUE} = \{\langle H, k\rangle \mid H \text{ is a bipartite graph that has a } k\text{-bipartite clique}\} .$$

   Prove that BIPARTITE-CLIQUE is NP-complete. (You may use the fact that the Clique problem, to determine if a graph G has a clique of size k, is NP-complete.)

2. A language is called unary if every string in it is of the form 1^i (the string of i ones) for some $i \geq 0$; in other words it is a subset of $\{1\}^*$.

   (a) Show that if a unary language is NP-complete then P = NP.

   (b) Prove that if every unary NP-language is in P then EXP = NEXP.

3. Define the Boolean function $\text{MAJORITY}_n : \{0,1\}^n \to \{0,1\}$ as:

$$\text{MAJORITY}_n(x_1, x_2, \ldots, x_n) = 1 \text{ if and only if } \sum_{i=1}^{n} x_i \geq n/2 \ .$$

Thus MAJORITY_n returns the majority vote of its inputs. Show that MAJORITY_n can be computed with $O(n)$ size Boolean circuits (with NOT gates and fan-in 2 AND and OR gates).

4. POLYNOMIAL-IDENTITY-TESTING (PIT) is the problem of determining whether a polynomial $p(x_1, \ldots, x_m)$ is identically zero.

In the lecture we mentioned that PIT $\in$ BPP because of the fact that a low degree polynomial evaluates to zero on only a small fraction of its domain (Schwartz-Zippel Lemma).

(a) Prove the following weaker version of Schwartz-Zippel Lemma:
Let $p(x_1, \ldots, x_m)$ be a polynomial of total degree d and S is any finite set of integers. When $x_1, \ldots, x_m$ are randomly chosen with replacement from S, then

$$\mathbf{Pr}[p(x_1, \ldots, x_m) = 0] \leq \frac{d}{|S|}$$

(b) If $G = (V, E)$ is an undirected graph then a perfect matching is some $E' \subseteq E$ such that every node appears exactly once among the edges of E'. Define PERFECT-MATCHING as the problem of determining whether an input graph G contains a perfect matching.
It is known that PERFECT-MATCHING $\in$ P.
In this problem, you are to prove (using PIT, without appealing to the known polynomial time deterministic algorithm) that PERFECT-MATCHING $\in$ BPP.
Hint: Consider the determinant of the matrix A defined in the following way:

$$A = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix}$$

where we identify the vertex set V with $\{1, 2, \ldots, n\}$ and define

$$a_{ij} = \begin{cases} X_{ij} & \text{if } \{i,j\} \in E \text{ and } i < j \\ -X_{ji} & \text{if } \{i,j\} \in E \text{ and } j < i \\ 0 & \text{otherwise.)} \end{cases}$$

where the X_{ij}'s are indeterminates.

5. Prove that if NP $\subseteq$ BPP then NP = RP.

6. Let L be an arbitrary language in BPP. Prove that there exists a 3-ary relation $R(x, y, z)$ that is decidable in time polynomial in $|x|$ with the following property:

- If $x \in L$, then $\exists y \ \forall z \big[R(x, y, z) = 1\big]$.
- If $x \notin L$, then $\exists z \ \forall y \big[R(x, y, z) = 0\big]$.

In what way is this a stronger inclusion than $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}}$?

(Hint: Extend the approach behind the proof that BPP is contained within the second level of the polynomial time hierarchy.)

7. (a) Let $s : \mathbb{N} \to \mathbb{N}$ be a function such that $n \leq s(n) < \frac{2^n}{9n}$ for $n \geq 10$. Prove that for all large enough n, there exists a function $g : \{0,1\}^n \to \{0,1\}$ that can be computed by a circuit of size $2 \cdot s(n) + O(1)$ but not by a circuit of size $s(n)$. (You may use the fact there there exists some function which cannot be computed by size $s(n)$ circuits when $s(n) < 2^n/(9n)$; this follows by a simple counting argument.)

   (b) Strengthen the result of Part (a) above by proving that there is a function $g : \{0,1\}^n \to \{0,1\}$ that can be computed by a circuit of size $s(n) + O(n)$ but not by a circuit of size $s(n)$.

8. **"Alternate Valiant-Vazirani proof."** Let X be a set of cardinality n. Let $\mathcal{F}$ be a nonempty collection of subsets of X. Let $w : X \to [N]$ be a random "weight" function, where each value $w(x)$ is chosen uniformly from $[N] = \{1, 2, \ldots, N\}$, independently for each $x \in X$. For a given set $C \in \mathcal{F}$, define its weight to be $w(C) = \sum_{x \in C} w(C)$; this is a random variable. After w is chosen, look at the minimum weight among sets in $\mathcal{F}$; there might be a unique minimum-weight set, or there might be multiple sets tied for the minimum. If the former case occurs, say that w is "isolating" for $\mathcal{F}$.

   (a) Show that $\mathbf{Pr}[w \text{ is isolating}] \geq 1 - n/N$. (Hint: For $x \in X$, upper bound the probability of the event A_x that there are two minimum-weight sets in F, one containing x and the other not containing x.)

   Valiant-Vazirani says that Unique-SAT $\in \mathsf{P}$ implies SAT $\in \mathsf{RP}$. Let's show the same thing for k-CLIQUE (which turns out to be an equivalent statement, via parsimonious interreductions.)

   (b) An algorithm is given an n-vertex graph G and a number k. It gives each vertex v a random weight $w(v)$ in the range $[2n]$. Show that if G had a k-clique, it has a unique maximum-weight k-clique with probability at least $1/2$.

   (c) Suppose we form G' by blowing up each node v in G to a clique of size $2nk + w(v)$; we put either all possible edges or no possible edges between the "u-clique" and the "v-clique", depending on whether or not (u, v) was an edge in G. Let $k' = 2nk^2 + r$, where $r \in [2nk]$ is chosen randomly. Show that if G's maximum clique has size $< k$, then with probability 1, G' has no k'-clique; and, if G's maximum clique has size k, then G' has a unique k'-clique with probability at least $1/(4nk)$.