CSE 531: Complexity theory — Winter 2007

Problem Set #1 — Instructor: Venkatesan Guruswami

Due on **January 24, 2007** in class.

Instructions: You are allowed to collaborate with 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, you must acknowledge for each problem the people you worked with on that problem. I also urge you think about each problem by yourself for at least 30 minutes before choosing to collaborate with others.

You are expected to refrain from referring to any source other than Sipser's text and your class notes in coming up with your solutions. The problems have been carefully chosen for their pedagogical value and hence might be similar or identical to those given out in past offerings of this course at UW, or similar courses at other schools. Using any pre-existing solutions from these sources, or using solution material from the Web is, needless to say, strictly prohibited.

Most of the problems require only one or two key ideas for their solution – spelling out these ideas should give you most of the 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.

Each problem is worth 10 points unless noted otherwise.

1. Prove that if P = NP, there exists a polynomial time algorithm that, given a Boolean formula ϕ, actually produces a satisfying assignment for ϕ if it is satisfiable.

2. (5 points) Show that the language $\mathrm{ALL}_{\mathrm{DFA}} = \{\langle M\rangle \mid M \text{ is a DFA that accepts all inputs}\}$ can be decided in polynomial time.

3. Show that $\mathrm{PRIMES} = \{n \mid n \text{ is a prime number in binary}\} \in \mathrm{NP}$.

   Hint: You may use the following fact without proving it: A number p is prime iff there exists an integer x, $1 < x < p$, that is relatively prime to p, such that $x^{p-1} \equiv 1 \pmod{p}$ and $x^i \not\equiv 1 \pmod{p}$ for $1 \le i < p-1$.

4. Define MAJ-3SAT to be the language of 3-CNF formulas ϕ for which there is an assignment to the variables which satisfies *at least two* out of the three literals in *every* clause. For example, $(x \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee \bar{z}) \wedge (y \vee z \vee w)$ is in MAJ-3SAT since the assignment $x = 0$, $y = 1$, $z = 0$ and $w = 1$ satisfies at least two literals in each clause. Prove that MAJ-3SAT is in P.

   (Hint: See also Problem 7.37 in Sipser's book.)

5. (5 + 10 = 15 points) (Non-uniform classes) For a language class $\mathcal{C}$ and an integer valued function $a : \mathbb{N} \to \mathbb{N}$, define the complexity class $\mathcal{C}/a(n)$ as follows. A language $A \in \mathcal{C}/a(n)$ iff there exists some language $B \in \mathcal{C}$ and a collection of advice strings $z_0, z_1, z_2 \ldots$ (one for each input length) with the length of z_i equal to $a(i)$ such that $A = \{x \mid \langle x, z_{|x|}\rangle \in B\}$. In other words, membership in A can be decided with the aid of an algorithm for B and an advice string that depends only on the length of the input.

   (a) Define $P/\mathrm{poly} = \bigcup_{c \ge 0} \mathrm{P}/n^c$. Prove that P/poly is precisely the class of languages that admit a circuit family of polynomial size.

(b) (Somewhat tricky) Define $\mathrm{P}/\log = \bigcup_{c \geq 1} \mathrm{P}/c \log n$. Prove that if $\mathrm{NP} \subseteq \mathrm{P}/\log$, then $\mathrm{NP} = \mathrm{P}$.

Hint: Run over the polynomially many advice strings in search of one that can certify satisfiability (use the idea from Problem 1 for this, together with a "padding" trick).

6. In the 3-set splitting problem, one is given as input a universe U and a family $\mathcal{F}$ of 3-element subsets of U. The goal is to determine if there is a partition of U into two disjoint parts as $U = U_1 \cup U_2$ so that every set $S \in \mathcal{F}$ crosses the partition, i.e., $S \cap U_1 \neq \emptyset$ and $S \cap U_2 \neq \emptyset$. (For those who know the following terminology: we are given a 3-uniform hypergraph and the goal is to determine if it is 2-colorable.) Formulate 3-set splitting as a language and show that it is NP-complete.

   Hint: Solve Problem 7.22 in Sipser's book, showing that Not-All-Equal-3SAT is NP-complete. Then develop a gadget that forces two elements to be in opposite partitions in order to capture negations.