CSE 531: Complexity theory — Winter 2007

Problem Set #5 — Instructor: Venkatesan Guruswami

Due in class on Wednesday, **March 7, 2007**,
or to Prasad by 5pm on Friday, Mar 9.

Instructions: Same as for Problem set 1.

---

1. (15 points) Define the *randomized* version of the complexity class NP, denoted PrNP, to consist of exactly those languages L for which there exists a randomized polynomial time Turing machine M, and polynomials p_1, p_2 such that

   Completeness: $x \in L \implies \exists w \in \{0,1\}^{p_1(|x|)}$ s.t. $\text{Prob}_{r \in \{0,1\}^{p_2(|x|)}}[M(x,w) \text{ accepts }] \geq 2/3$.

   Soundness: $x \notin L \implies \forall w \in \{0,1\}^{p_1(|x|)}$, we have $\text{Prob}_{r \in \{0,1\}^{p_2(|x|)}}[M(x,w) \text{ accepts }] \leq 1/3$.

   The one-sided version of PrNP, denoted RNP, is defined similarly, with the probability 2/3 in the completeness case replaced by 1.

   (a) Prove that PrNP = RNP.

   (b) Prove that $\mathsf{PrNP} \subseteq \Sigma_2^P \cap \Pi_2^P$.

2. (10 points) Prove the following Karp-Lipton style collapse theorem for PSPACE:
   If $\mathsf{PSPACE} \subseteq \mathsf{P/poly}$, then $\mathsf{PSPACE} = \Sigma_2^P$.
   (Hint: Consider the IP characterization of PSPACE — what is the complexity of the prover for the interactive protocol for TQBF? Use this and try to prove the stronger conclusion $\mathsf{PSPACE} \subseteq \mathsf{PrNP}$.)

3. (10 points)

   (a) Prove that the class IP remains unchanged if we allow the prover to be probabilistic, i.e., the prover's strategy can be chosen at random from some distribution on functions.

   (b) Define IP′ to be the class of languages that have an interactive proof where the verifier is a deterministic polynomial time Turing machine. What class does IP′ correspond to?

   (c) Define SimpleIP to the class of languages that have an interactive protocol where the prover sends a single message and then the verifier makes an accept/reject decision based on this message (so there is in fact no interaction). Prove that SimpleIP is unlikely to equal IP by showing $\mathsf{SimpleIP} \subseteq \Sigma_2^P$.

4. (15 points) The general task of program verification, i.e., deciding whether or not a given program solves a certain computational problem, is undecidable. In this exercise, we will investigate a weaker notion called instance checking where we check correctness of a program on an input by input basis. Formally, let A be a program solving a decision problem Π (viewed as a Boolean function). An instance checker for Π is a randomized polynomial time oracle TM C, such that for any input x, the following hold:

   - If A is a correct program for Π (i.e., $\forall y,\ A(y) = \Pi(y)$), then C^A (i.e., C with oracle access to A) accepts $A(x)$ with probability at least 2/3.

- For all A, if $A(x) \neq \Pi(x)$, then C^A accepts $A(x)$ with probability at most $1/3$.

For problems in BPP, such instance checking is of course trivial. Surprisingly, many problems not known to be in BPP (and thus not known to be efficiently solvable) admit such checkers.

Prove that the language $\mathsf{GNI} = \{\langle G_1, G_2 \rangle \mid G_1 \text{ and } G_2 \text{ are non-isomorphic graphs}\}$ admits an instance checker.