CS 142 & Math 167
Fall 2008
Homework 9
Due Friday, December 12 by 5 PM

PLEASE submit each problem on a separate sheet and *include your name on each sheet*. The sheets will be distributed to the presenters to grade. You do not need to submit a solution to a problem if you assigned it. If you have questions regarding a problem, please first try to talk to the person or group who assigned that problem. If you are unable to find them or feel that you need additional help, please come and talk to Ran.

1. [**Elaine, Andrew, and Adrian's Presentation**] In this problem, you will revisit the "traditional" method for error reduction we saw in class. This is an amplification result for Arthur-Merlin games – very similar in spirit to analogous results that we saw in Devin and Andrew's presentation on randomized complexity classes!

   Given A, an Arthur-Merlin proof system for the language L that has error probability $1/3$, show how to construct an Arthur-Merlin proof system A^* for L with error probability 2^{-k}, where k is a polynomial. Recall that if e is the error probability of a proof system, then for any $w \in L$, Arthur (A) must accept with probability $1 - e$, and if $w \notin L$, A cannot accept with probability greater than e. For notational consistency, let ℓ denote the number of random bits Arthur sends to Merlin in each round of A. Let g be the number of rounds in A.

   We will break this problem into the following parts:

   (a) Describe how A^* works. Please be more specific than saying "it plays multiple copies of the game in parallel". Please give sufficient detail so that your response to parts 1b and 1c follow easily. A few sentences should suffice.

   (b) Describe why this technique achieves the desired error probability of 2^{-k}. Recall that, when we have n events of constant probability p, the probability that at most k of them occur is given by the cumulative binomial distribution. You will probably want to use the upper bound:

$$\text{BinomCdf}(k, n, p) \leq \exp\left(-2\frac{(np-k)^2}{n}\right)$$

(c) Observe the number of bits A^* must send at each round, compare this to the more efficient result we stated in class.

2. [**David and Kevin's Presentation**] Let's define a "new" type of reducibility: We'll call this E-reducibility. We say that a reduction (f, g) is a E-reduction from problem A to problem B if a polynomial p and positive constant β such that

   (a) For any x that is an instance of A, $\text{opt}_B(f(x)) \leq p(|x|)\text{opt}_A(x)$.

   (b) For any x that is an instance of A, and for any y that is a solution of $f(x)$,

   $$R_A(x, g(x, y)) \leq 1 + \beta(R_B(f(x), y) - 1).$$

   (a) Our goal here is to show that E-reductions preserve membership in PTAS and APX (in fact, they preserve membership in just about all relevant approximation classes, but we won't go into that here!). Begin by showing that any E-reduction is also an A-reduction by finding a suitable $c_A(r)$ function.

   (b) Now, show a suitable $c_P(r)$ function that prove that any E-reduction is also a P-reduction. If you do this correctly, it should be a quick transformation from your function in part a!

3. [**Richard's Presentation**] Recall the predicate $\text{INTDIV}(x, y, q, r)$ which says $x = yq + r$.

   (a) In class, we asserted that $\geq$ and $>$ could be expressed in terms of the various primitive operators making up the alphabet of L, including $=$. Show how to express $\geq$ and $>$.

   (b) In class, we asserted that PRIME could be expressed as a predicate (i.e., a predicate $\phi(x)$ which is true only if x is prime). Show how to do this.

   (c) In this problem, we will make a predicate for $\text{POW}_p(x)$, which is true only if x is a power of the prime p. Prof. I. Lai. suggests

   $$\text{POW}_p(x) \Leftrightarrow x = 1 \vee (\exists y \text{INTDIV}(x, y, p, 0) \wedge \text{POW}_p(y))$$

   Explain why Prof. Lai is wrong, and give a correct predicate.

4. [**Devin and Andrew's Presentation**] Show that RP $\cap$ co-RP $\subseteq$ ZPP.

5. **[Steven and Josh's Presentation]** Recall that $Bad(f)$ is the probability that a given circuit will accept a random complete $k-1$ partite graph, and $pluck(f)$ will replace a set of m clique indicators with a single clique indicator corresponding to the kernel of the sunflower. Prove that $Bad(pluck(f)) \leq Bad(f) + \left(\frac{\binom{l}{2}}{k-1}\right)^p$. (Hints: You can assume that the kernel of the sunflower you remove is empty. Why is this useful? Why is this true? Each petal has an independant probability of being somewhere where f does not accept but $pluck(f)$ does.)

6. **[Ted's Presentation]** In this problem you will examine the running time that arises in the proof of Blum's Theorem. Let $M_1, M_2, \ldots$ be an enumeration of all Turing Machines. Let $h(n)$ be the function such that $h(0) = 2$ and $h(n) = 2^{h(n-1)}$. Thus $h(1) = 2^2, h(2) = 2^{2^2}$, and so on. Let u and n be fixed, where $u < n$. We want to determine, for each machine M_i and input 0^j such that $u \leq i \leq j \leq n$, whether the computation $M_i(0^j)$ halts in at most $h(j-i)$ steps. In other words, we want to compare the runtimes of each of the computations

$$\begin{array}{cccc} M_u(0^u) & M_u(0^{u+1}) & \cdots & M_u(0^n) \\ & M_{u+1}(0^{u+1}) & \cdots & M_{u+1}(0^n) \\ & & \ddots & \vdots \\ & & & M_n(0^n) \end{array}$$

with the corresponding value

$$\begin{array}{cccc} h(0) & h(1) & \cdots & h(n-u) \\ & h(0) & \cdots & h(n-u-1) \\ & & \ddots & \vdots \\ & & & h(0) \end{array}$$

Show that this can be done (for large n) in at most $h(n-u+1)$ time; you may assume that h itself takes no time to compute.

7. **[Shaun's Presentation]** Recall from lecture that the Circuit Value Problem (CVP) is P-complete for NC^1 reductions. This problem serves a similar function for P-completeness as SAT does for NP-completeness, as most proofs that show a problem is P-complete involve a reduction from CVP. In this problem you will show that a few other problems are P-complete.

CVP: Given a Boolean circuit α, inputs $x_1, \ldots, x_n$, and a designated output y, Is the output y of α TRUE on input $x_1, \ldots, x_n$.

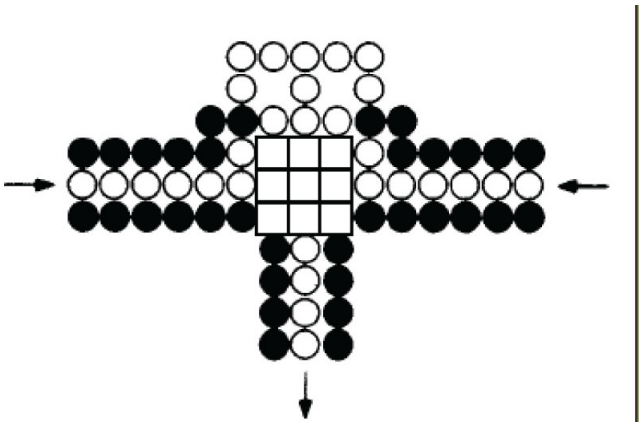
(a) Prove that PCVP is P-Complete where PCVP is defined as follows: Given an encoding of a planar boolean circuit (where no wires cross) α, inputs $x_1, \ldots, x_n$, and a designated output y, Is the output y of α TRUE on input $x_1, \ldots, x_n$.

(b) Prove that MCVP is P-Complete where MCVP is defined as follows: Given an encoding of a monotone boolean circuit, inputs $x_1, ..., x_n$, and a designated output y, Is the output y of α TRUE on input $x_1...Xn$. Recall, a circuit is monotone if it is made purely of AND and OR Gates.

Reductions like these are very important because the restricted form of MCVP and PCVP often make reductions to more complicated problems much simpler than reducing from pure CVP.

8. [**Denis's Presentation**]

This is an incomplete diagram of a graph joint vertex widget. White is trying to escape through one of the paths coming in from the sides. White plays first, and the direction of the moves should go from the outside edges, through the vertex, and down the center.



(a) Complete the above diagram, placing 5 black and 2 white stones and indicating spaces that are meant to remain empty with numbers, just like in the diagrams from lecture.

(b) Explain the sequence of plays in the empty spaces that makes this widget work, with white going first. For each forced move, explain why going anywhere else would result in an automatic lose for that player.