Fundamental Algorithms, Assignment ω

Due May 4/5 in Recitation.

The Final Exam is Monday, May 15, 5:10-7:00, ciww 109 (our regular classroom)
Check website for special office hours, comments on final, etc.

> To me, it does not seem unlikely that on some shelf of the universe there lies a total book. I pray the unknown gods that some man - even if only one man, and though it have been thousands of years ago! - may have examined and read it. If honor and wisdom and happiness are not for me, let them be for others. May heaven exist, though my place be in hell. Let me be outraged and annihilated, but may Thy enormous Library be justified, for one instant, in one being.
> from The Library of Babel by Jorge Luis Borges

1. Which of the following problem classes are in P and which are probably not in P. (By probably not we mean that we do not as of today know that it is in P but of course tomorrow somebody might come up with a clever algorithm.)

   (a) `PRIME`. The input here would be integers n and Yes would be returned iff n is prime.

   (b) I gave the above problem twenty years ago. What was the answer then?

   (c) `CONNECTED-GRAPH`. The input here would be a graph G and Yes would be returned iff the graph was connected.

   (d) `TRAVELING-SALESMAN`. The input here would be a graph G together with a positive integer weight $w(e)$ for each edge e and an integer B. Yes would be returned iff there was a Hamiltonian Cycle which had total weight at most B.

   (e) `SPANNING-TREE`. The input here would be a graph G together with a positive integer weight $w(e)$ for each edge e and an integer B. Yes would be returned iff there was a spanning tree which had total weight at most B.

   (f) `ALMOSTDAG`. The input here would be a directed graph G. Yes would be returned iff there was a set of at most 10 edges of G that could be removed from G so that the remaining graph is a `DAG`. (Your argument should work with 10 replaced by any *constant* value.)

2. Show that the following problem classes are in NP. (That is, describe the certificate that the Oracle gives and describe the procedure that Verifier will take. Warning: Do not trust Oracle! For example, if Oracle gives you n distinct vertices you have to verify that they are indeed distinct!)

   (a) `PRIME-INTERVAL` The input here would be integers n, a, b. Yes would be returned iff there was a prime p which divided n and for which $a \leq p \leq b$.

   (b) `TRAVELING-SALESMAN` As described above.

   (c) `COMPOSITE` The input here would be an integer n. Yes would be returned if n was composite. For this problem I want two solutions. One (the easier one) uses the Agarwal, Kayal, Saxena algorithm. The second should *not* use the Agarwal, Kayal, Saxena algorithm.

   (d) `3-COLOR`. The input here would be a graph G. Yes would be returned if there was a three coloring of the vertices such that no two adjacent vertices v, w had the same color.

   (e) `NEAR-DAG`. The input here would be a directed graph G and an integer B. Yes would be returned if there was a set of at most B edges that could be removed from G so that the remaining graph was acyclic. (This is like `ALMOST-DAG` with the critical distinction that B is not restricted to 10, or any constant value. Rather, B can depend on the number of vertices of G.)

3. For the following pairs L_1, L_2 of problem classes show that $L_1 \leq_P L_2$. That is, given a "black box" that will solve any instance of L_2 in unit time, create a polynomial time algorithm that will solve any instance of L_1 in polynomial time.

   (a) Let L_2 be `TRAVELLING-SALESMAN DESIGNATED PATH`. The input here would be a graph G, two designated vertices, a source v_1 and a sink v_n, together with a positive integer weight $w(e)$ for each edge e and an integer B. Yes would be returned iff there was a Hamiltonian Path (i.e., one goes through all the vertices $v_1, \ldots, v_n$ in some order (starting and ending at the designated vertices) but does *not* return from v_n back to v_1) which had total weight at most B. L_1 is `TRAVELLING-SALESMAN` as described above.

   (b) Let L_2 be `CLIQUE`. The input here would be a graph G together with a positive integer B. Yes would be returned iff there was

a clique with at least B vertices. (A set of vertices in a graph G is a clique if *every* pair of them are adjacent.) Let L_1 be `INDEPENDENT-SET`. The input here would be a graph G together with a positive integer B. Yes would be returned iff there was a independent set with at least B vertices. (A set of vertices in a graph G is an independent set if *no* pair of them are adjacent.)

4. Assume `PRIME-INTERVAL` (defined above) is in P. Using it as a black box give a polynomial time algorithm with input integer $n \geq 2$ that returns some prime factor p. (Caution: This means polynomial in the number of digits in n, or what is sometimes called polylog n, meaning $O(\lg^c n)$ for some constant c.) (*) Further, give a polynomial time algorithm with input integer $n \geq 2$ that returns the entire prime factorization of n.

   You just keep right on thinking there Butch, thats what you're good at.
   Robert Redford to Paul Newman
   Butch Cassidy and the Sundance Kid