HW 4 (due Monday, at noon, February 25, 2013)

CS 473: Fundamental Algorithms, Spring 2013 Version: **1.21**

Make sure that you write the solutions for the problems on separate sheets of paper. Write your name and netid on each sheet.

Collaboration Policy: The homework can be worked in groups of up to 3 students each.

---

1. (30 PTS.) Goodbye, and thanks for all the negative cycles.

You are given a directed graph G with (possibly negative) weights on the edges, and a source vertex s.

(A) (20 PTS.) Show how to modify Bellman-Ford so that it outputs a negative cycle it had found in the graph reachable from the source s. **Prove** that your algorithm indeed outputs a negative cycle in the graph.

(B) (10 PTS.) Describe an algorithm that computes for all the vertices in the given graph their distance from s.
Notice, that your algorithm needs to correctly handle vertices in G whose distance from s is $-\infty$ (there is a walk from s to such a vertex that includes a negative cycle).
For full credit, the running time of your algorithm must be $O(mn)$. (You do not have to use (A) to do this part.)

2. (30 PTS.) Longest common palindrome.

You are given two strings S and T of length n. Describe an algorithm, as fast as possible, that output the longest palindrome that appears, as a substring (a substring here might not necessarily a contiguous in the original string), in both S and T. For example, for

$$S = \text{aobracadabora} \qquad T = \text{ahomalo graphica},$$

the longest common palindrome, seems to be "aoaoa".

3. (40 PTS.) Recovering a message.

You are given a binary string S of length n that was transmitted using a new transmitted. Unfortunately, there is noise in the received string - new fake bits were added to it. Because of the way the message was encoded, you know that certain strings $s_1, \ldots, s_t$ can not appear consecutively in the original string.

Compute the longest substring (not necessarily consecutive) in S that does not contain any of the forbidden strings (consecutively). Formally, each of the forbidden strings can not appear as (consecutive) substrings of the output string. For example, consider the input string 1100001110000011100000111, and the forbidden substrings $s_1 = 00$ and $s_2 = 11$. Clearly, the longest legal output string here is 1010101.

(Hint: You might want to use material you learned in CS 373 in solving this question.)

(A) (20 PTS.) (This part is significantly harder than the other part – you might want to solve the other part first.) Describe an algorithm for the case that t is small (i.e., think about t as being a constant [hint: Solve the case $t = 1$ and $t = 2$ first]), but the strings $s_1, \ldots, s_t$ might be arbitrarily long. How fast is your algorithm? (Faster is better, naturally.)
For full credit for this part, it is enough if you solve the problem for $t = 1$ (the problem turned out to be harder than expected).

(B) (20 PTS.) Describe an algorithm for the case that t might be large, but the strings $s_1, \ldots, s_t$ are of length at most ℓ, and ℓ is relatively small (i.e., think about ℓ as being, say, 5). How fast is your algorithm? (Faster is better, naturally.)