MCS 401 – Computer Algorithms I

Spring 2017

Problem Set 4

Lev Reyzin

Due: 11/8/17 by the beginning of class

Instructions: Atop your problem set, write your name and whether you are an undergraduate or graduate student. Also write the names of all the students with whom you have collaborated on this problem set.
Important note: Problems labeled "(**U**)" and "(**G**)" are assigned to undergraduate and graduate students, respectively. Udergraduate students can get a small bonus for solving the graduate problems. Graduate students are encouraged to solve the undergraduate problems for practice.

1. [10 pts.] Consider the following coin changing problem. You are given a value N and an infinite supply of coins with values $d_1, d_2, \ldots, d_k$. Give an $O(Nk)$ algorithm for finding the smallest number of coins that add up to the value N.

2. [**10 pts**] Call string C a *braiding* of strings A and B if it contains all (and only) the characters of both A and of B and if their respective order is preserved in C. For example, $C = aacabbaa$ is a braiding of $A = aaba$ and $B = caba$ (demonstrated as follows: **aa***cab***ba***a*). Give an algorithm that, given strings A, B, and C, decides whether C is an braiding of A and B in polynomial time. Prove your answer correct.

3. [**10 pts**]

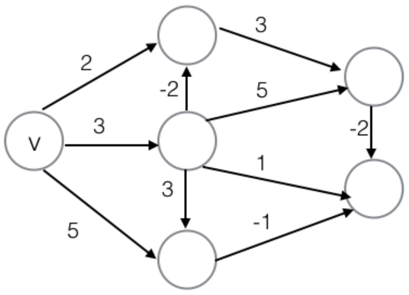
(**U**) A subsequence is a sequence that can be derived from another sequence by deleting some elements without changing the order of the remaining elements; e.g. "acef" is a subsequence of "abcdef." Consider the problem of finding the longest common subsequence of two sequences – this is a task versioning systems like git or cvs often solve. Show that this is a special case of the sequence alignment problem. Then, give a polynomial-time algorithm for finding the longest subsequence common to *three* sequences. Analyze its running time and argue why it is correct.

(**G**) You are given a complete binary tree on $n = 2^d - 1$ vertices (for $d \geq 1$), rooted at vertex r. Further, each vertex i in the tree is assigned a weight $w_i > 0$. The problem is to find the k-vertex subtree[1] (with $1 \leq k \leq n$) rooted at r for which the sum of the weights of its

[1]A subgraph of a graph G is another graph formed from a subset of the vertices and edges of G. The vertex subset must include all endpoints of the edge subset, but may also include additional vertices. A subtree is a connected subgraph of a tree.

included vertices is maximized. Give an algorithm that does this in time polynomial in n and k and argue about its correctness.

4. [10 pts] What would it mean to add memoization to Strassen's matrix multiplication algorithm? What asymptotic improvement (if any) does it yield in the worst case? Explain your answer.



5. [10 pts] Consider the weighted directed graph above. Draw 1) the shortest path tree rooted at v that would be found by Dijkstra's algorithm and 2) the shortest path tree rooted at v that would be found by the Bellman-Ford algorithm. (You must show your work for partial credit.) Which, if any (or both), of these trees give the correct shortest paths?