CSE 202: Design and Analysis of Algorithms Spring 2016

Problem Set 2

Instructor: Kamalika Chaudhuri **Due on:** Fri Apr 29

Instructions

- For your proofs, you may use any lower bound, algorithm or data structure from the text or in class, and their correctness and analysis, but please cite the result that you use.
- Each problem is worth 10 points.
- If you do not prove that your algorithm is correct, we will assume that it is incorrect. If you do not provide an analysis of the running time, we will assume you do not know what the running time is.

Problem 1

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$ we wish to make change for a value v using at most k coins; that is, we wish to find a set of $\leq k$ coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and $k = 6$, then we can make change for 55 but not for 65. Give an efficient dynamic-programming algorithm for the following problem. The input is $x_1, \ldots, x_n, k, v$, and the output is whether it is possible to make change for v using $\leq k$ coins of denominations $x_1, \ldots, x_n$.

Problem 2

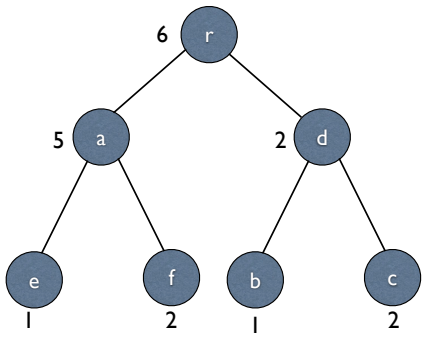
Suppose you are given a labelled directed *acyclic* graph $G = (V, E, L)$, where L represents the labels on the edges. Each edge $e \in E$ has one of three possible labels – ((an open parenthesis),) (a closed parenthesis) and 0. A path in the G is said to be *balanced* if the sequence of labels of the edges along the path include the same number of open and closed parentheses. For example, paths with label sequences $(, 0,), (, 0,)$ and $), 0, (, 0, 0$ are balanced, while a path with a label sequence $(,),)$ is not balanced.

Given G and two specified nodes s and t in G, design a dynamic programming algorithm to find if there exists a balanced path between s and t in G, and output the path if it exists.

Problem 3

Suppose you are given a rooted binary tree T in which each node x has a score $s(x)$. A pruning of T is a set C of nodes in T with the property that C contains exactly one node on any root to leaf path. Thus, if a node x is in C, no ancestor or descendant of x can be in C.

The size of a pruning C is the number of nodes in C, and the score of a pruning is the sum of the scores of the nodes in it. For example, the following tree has 2 prunings of size 3: $C_1 = \{a, b, c\}$ with score 8 and $C_2 = \{d, e, f\}$ with score 5, and one pruning of size 2: $C_3 = \{a, d\}$, with score 7.



Given a binary tree T, with root r, and the scores $s(x)$ for each node x of T, design a dynamic programming algorithm to find the score of a pruning C of T such that (a) the size of C is exactly k (b) the score of C is as large as possible. To get any credit, the running time of your algorithm should be polynomial in n and k. (Hint: The exhaustive search algorithm that checks all possible prunings of size k has $n^{O(k)}$ running time.)