Problem Set 4, CS 5800 Spring 2017

Due: Wed, 3/1, 2PM

Problem 1. (Difficulty 3) Bob has to plan his working life for the next T years. There are m jobs. Every year Bob must do exactly one job. Job j pays $W(j,t)$ during year t. Bob can switch job a maximum of S times. Bob starts doing job 0. Give an algorithm that produces the plan for Bob that maximizes the sum of his earnings for the next T years. The running time of the algorithm should be polynomial in T, m, and S.

Problem 2. (Difficulty 3) A special rod-cutting machine enables a worker to break a rod into two pieces. This operation costs n time units to break a rod of length n into two pieces. Suppose a worker wants to break a rod into many pieces by cutting it at different break points. The order in which the breaks occur can affect the total amount of time used. For example, suppose that the worker wants to break a 20 units rod after 2 units, 8 units, and 10 units. If he performs the breaks in left-to-right order, then the first break costs 20 time units, the second break costs 18 time units (breaking the rod from 3 units to 20 units), and the third break costs 12 time units, totaling 50 time units. If he performs the breaks to occur in right-to-left order, however, then the first break costs 20 time units, the second break costs 10 time units, and the third break costs 8 time units, totaling 38 time units.

Design an algorithm that, given a rod R of length n and an array $L[1 \ldots m]$ containing the break points, computes the lowest cost for a sequence of breaks, as well as the sequence of breaks. The running time should be polynomial in m.

Problem 3. (Difficulty 3) Use dynamic programming to count the number of sequences of length n from the alphabet $\{A, B, C\}$ such that no two adjacent symbols are equal. Make your algorithm as fast as you can. For example for $n = 2$ the sequences are AB, AC, BA, BC, CA, and CB, and so the answer is 6.

Problem 4. (Difficulty 3) You are given n subsets $S_1, S_2, \ldots, S_n$ of {1,. . . ,k}, each with an integer cost $c(i)$. Find the minimum-cost collection of subsets that covers all $\{1, \ldots, k\}$ in time $O(2^k \cdot n)$.