CS 170 HW 6

Due on 2018-03-04, at 11:59 pm

1 (★) Study Group

List the names and SIDs of the members in your study group.

2 (★) Longest Increasing Subsequence

Given an array A of n integers, here is a linear time algorithm to find the longest increasing subsequence, where $M[j]$ represents the longest increasing subsequence of the first j integers of A.

$$M[1] = 1$$
$$M[i] = \begin{cases} M[i-1] & \text{if } A[i-1] > A[i] \\ M[i-1]+1 & \text{otherwise} \end{cases}$$

Verify that this algorithm takes linear time. Is it correct? Prove it is or give a counterexample and explain.

3 (★★) Copper Pipes

Bubbles has a copper pipe of length n inches and an array of integers that contains prices of all pieces of size smaller than n. He wants to find the maximum value he can make by cutting up the pipe and selling the pieces. For example, if length of the pipe is 8 and the values of different pieces are given as following, then the maximum obtainable value is 22 (by cutting in two pieces of lengths 2 and 6).

length	1	2	3	4	5	6	7	8
price	1	5	8	9	10	17	17	20

Give a dynamic programming algorithm so Bubbles can find the maximum obtainable value given any pipe length and set of prices. Give your answer in a four-part solution, but give only a brief justification of your algorithm in the proof section.

4 (★★★) Road Trip

Suppose you want to drive from San Francisco to New York City on I-80. Your car holds C gallons of gas and gets m miles to the gallon. You are handed a list of the n gas stations that are on I-80 and the price that they sell gas. Let d_i be the distance of the i^{th} gas station from SF, and let c_i be the cost of gasoline at the i^{th} station. Furthermore, you can assume that for any two stations i and j, the distance $|d_i - d_j|$ between them is divisible by m. You start out with an empty tank at station 1. Your final destination is gas station n. You may not run out of gas between stations but you need not fill up when you stop at a station, for example, you might to decide to purchase only 1 gallon at a given station.

Find a polynomial-time dynamic programming algorithm to output the minimum gas bill to cross the country. Give only a brief justification of your algorithm. Analyze the running time of your algorithm in terms of n and C. You do not need to find the most efficient algorithm, as long as your solution's running time is polynomial in n and C.

5 (★★★★) Propositional Parentheses

Say you are given a propositional logic formula using $\wedge, \vee, T$, and F that does not include parentheses. You want to find out how many different ways there are to draw parentheses so that the resulting formula evaluates to true. For example, the formula $T \vee F \vee T \wedge F$ has 3 solutions: $(T \vee F) \vee (T \wedge F)$, $T \vee ((F \vee T) \wedge F)$, and $T \vee (F \vee (T \wedge F))$.

(a) Give a four-part solution that solves this problem.

(b) Say you were to randomly draw parentheses so that the sentence is now syntactically correct. For example, the formula $T \vee T \wedge F$ might yield $(T \vee T) \wedge F$ and $T \vee (T \wedge F)$ with equal probability, but not $T(\vee T) \wedge F$ or $(T) \vee T)(\wedge F$, since they are not syntactically correct. Briefly explain how you could use your original algorithm to find the probability that randomly drawing syntactically-correct parentheses causes the sentence to become true.

6 (★★) Jeweler

You're a jeweler. You can make necklaces or engagement rings. Necklace takes 4 oz of gold. Engagement ring takes 1 oz of gold + 1 diamond. You have a limited supply of gold at only 100 ounces, but unlimited diamonds. You make 30 profit per necklace and 100 per ring. There are only so many people getting married each month, so you can sell at most 40 engagement rings. You want to figure out how many of each type of jewelry you should make each month to maximize your profits. Formulate this problem as a linear programming problem and find the solution (state the cost-function, linear constraints, and vertices).

7 (★★★★★) Largest Substring Sum (Extra-Credit)

Let's say we are given a sequence of positive and negative integers $x_1, x_2, \ldots, x_n$. We use $sum(i, j)$ represent the sum of the *substring* $x_i, x_{i+1}, \ldots, x_j$. Give a four-part solution describing a linear-time algorithm to find the substring of $x_1, x_2, \ldots, x_n$ with the largest sum.

For example, the largest-summing substring of $[-1, 5, -2, -1, 4, -2, 1]$ is $[5, -2, -1, 4]$, which sums to 6.