CMPSCI 311: Introduction to Algorithms **Spring 2017**

Homework 5

Your Name: ____________________

Collaborators and sources: ____

You make work in groups, but you must write solutions yourself. List collaborators on your submission.

If you are asked to design an algorithm, please provide: (a) the pseudocode for the algorithm, (b) an explanation of the intuition for the algorithm, (c) a proof of correctness, (d) the running time of your algorithm and (e) justification for your running time analysis.

Submission instructions. This assignment is due by 8:00pm on Friday, Apr 7 in Gradescope (as a pdf file). Please review the course policy about Gradescope submissions on the course website.

1. **(10 points) Independent Set.** K&T, Chapter 6, Exercise 1

2. **(10 points) Maximum Subsequence Sum.** Recall the maximum subsequence sum (MSS) problem, for which we gave a $\Theta(n \log n)$ divide-and-conquer algorithm. In this problem you will develop a dynamic programming algorithm with running time $\Theta(n)$ to solve the problem.

   The input is an array A containing n numbers, and the goal is to find a starting index s and ending index t (where $s \leq t$) so the following sum is as large as possible:

   $$A[s] + A[s+1] + \ldots + A[t-1] + A[t]$$

   For example, in the array

   $$A = \{31, -41, 59, 26, -53, 58, 97, -93, -23, 84\}$$

   the maximum is achieved by summing the third through seventh elements, to get $59 + 26 + (-53) + 58 + 97 = 187$, so the optimal solution is $s = 3$ and $t = 7$. When all entries are positive, the entire array is the answer ($s = 1$ and $t = n$). In general, we will allow the case $s = t$ (a sequence of only one element) but not allow empty subsequences.

   (a) (3 points) Define $\text{OPT}(j)$ to be the maximum sum of any subsequence that *ends at index* j. Write a recurrence and base case for $\text{OPT}(j)$.

   Hint 1: In this case you know the ending index t must be equal to j. Consider two different possibilities for the starting index s.

   Hint 2: It may help to first write a recursive function `MSS(j)` to compute the maximum sum of any subsequence ending at j and then translate this into a recurrence.

   (b) (2 points) Describe in one sentence how we can find the solution to the maximum subsequence sum problem if we know the value of $\text{OPT}(j)$ for all j.

   (c) (5 points) Write the pseudo-code for a bottom-up dynamic programming algorithm for MSS that runs in $\Theta(n)$ time.

3. **(10 points) Texting.** You are competing with your friends to type text messages on your smartphone as quickly as possible. Here are the rules: you use two thumbs for texting and they start out on the bottom left and bottom right keys of the keyboard. To type a character, you move either thumb from its current key to the key you need to press, and it takes time equal to the distance between the keys. You can assume the following:

- The keyboard has keys labeled $\{1, 2, \ldots, k\}$ and there is a function `dist`(i,j) to calculate the distance between two keys i and j. (To visualize this, you may want imagine the digits 1 though 9 arranged on a standard numeric keypad).
- Your left thumb starts on key a, and your right thumb starts on key b. (For example, on the 9-digit numeric keypad, $a = 7$ is the bottom left key, and $b = 9$ is the bottom right key.)
- You can press any key with either thumb
- Both thumbs can rest on the same key if necessary
- The characters to by typed are $c_1c_2 \cdots c_n$, where $c_i \in \{1, 2, \ldots, k\}$ is the ith key to push

Design an algorithm that finds the fastest way to type the message. In other words, your algorithm needs to decide which thumb to use to type each character, and it should minimize the total distance moved by your two thumbs. Try to use $O(nk^3)$ time.

Example. Imagine the 9-digit numeric keypad where your thumbs start at $a = 7$ and $b = 9$, with input message $c_1c_2c_3 = 589$. The solution "left, right, left" would look like this:

0 Left/right thumbs start at 7/9

(a) Left thumb moves from 7 to $c_1 = 5$. Time = `dist`(7, 5). Thumbs end at 5/9.

(b) Right thumb moves from 9 to $c_2 = 8$. Time = `dist`(9, 8). Thumbs end at 5/8.

(c) Left thumb moves from 5 to $c_3 = 9$. Time = `dist`(5, 9). Thumbs end at 9/8.

Total time = `dist`(7, 5) + `dist`(9, 8) + `dist`(5, 9).

4. **(0 points).** How long did it take you to complete this assignment?