COMPSCI 311: Introduction to Algorithms **Fall 2023**

Challenge Problems 5

due 11/15/2023 at 11:59pm in Gradescope

Instructions. Limited collaboration is allowed while solving problems, but you must write solutions yourself. List collaborators on your submission.

You can choose which problems to complete, but must submit at least one problem per assignment. See the course page for information about how challenge problems are graded and contribute to your homework grade. Since you don't need to complete every problem, you are encouraged to focus your efforts on producing high-quality solutions to the problems you feel confident about. There is no benefit to guessing or writing vague answers.

If you are asked to design an algorithm, please (a) give a precise description of your algorithm using either pseudocode or language, (b) explain the intuition of the algorithm, (c) justify the correctness of the algorithm; give a proof if needed, (d) state the running time of your algorithm, (e) justify the running-time analysis.

Submissions. Please submit a PDF file. You may submit a scanned handwritten document, but a typed submission is preferred. Please assign pages to questions in Gradescope.

1 Challenge Problems

Problem 1. Pebbles Game. Two players play a game, starting with three piles of n pebbles each. Players alternate turns. On each turn, a player may do one of the following:

- take one pebble from any pile
- take two pebbles each from two different piles

The player who takes the last pebble(s) wins the game.

Write an algorithm that, given n, determines whether the first player can "force" a win. That is, can the first player choose moves such that, regardless of what the other player does, they will always win.
In addition, write a procedure that tells a player which move (if any) they should make to force a win given the current triple of pebbles in each pile.
What is the complexity of your solution?

Problem 2. 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 to imagine the digits 1 through 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 $c_1 c_2 \cdots c_n$, 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 $O(nk^2)$ time.

For this problem, you only need to return the value of the optimal solution, not the solution itself.

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

1. Left thumb moves from 7 to $c_1 = 5$. Time = `dist`(7, 5). Thumbs end at 5/9.
2. Right thumb moves from 9 to $c_2 = 8$. Time = `dist`(9, 8). Thumbs end at 5/8.
3. 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).

Problem 3. Polling Locations. Suppose n voters live in a town and there are k polling locations. We want to assign each voter to a polling location within r miles of their home (and we know the address of each voter and each polling location, so we can compute the distance from any voter's home to any polling location). However, each polling location can accept at most C voters.

Your goal is to design a polynomial-time algorithm for the following problem. Given the addresses of each voter and polling location as well as the range parameter r and capacity C of polling locations, decide whether every it is possible to match every voter to a polling location within r miles of their home such that each polling location is assigned at most C voters.

Problem 4. Network Flows (K&T Ch 7 Ex 12). You are given a flow network with unit-capacity edges: It consists of a directed graph $G = (V, E)$, a source $s \in V$, and a sink $t \in V$; and $c_e = 1$ for every $e \in E$. You are also given a parameter k. The goal is to delete k edges so as to reduce the maximum $s - t$ flow in G by as much as possible. In other words, you should find a set of edges $F \subset E$ so that $|F| = k$ and the maximum $s - t$ flow in $G = (V, E \setminus F)$ is as small as possible subject to this. Give an efficient algorithm to solve this problem, argue it is correct, and state its running time.