CSE 222 (ADA) Mid Semester Examination

March 9, 2022

Duration of Examination : 2 hours.

General instructions

Each question is worth 10 points. For each question we need you to provide the following:

1. This examination will be proctored online. Your videos have to be turned on throughout the duration of the exam.
2. The exam starts at 10:00 through GC assignment and ends at 12:00. You will be allowed 10 extra minutes for uploading. Any upload done after 12.10 according to GC will receive zero in the entire exam. So your attempt to upload needs to start at 12:00 sharp.
3. You can either write your solutions cleanly on paper/ type in a word document and upload a single pdf on GC.
4. Cellphones and tablets are strictly disallowed. You should leave your devices by your side and confirm this with your proctor. If any student is found using a cell phone, browsing the internet, talking to someone, or using any other unfair means of cheating, his/her name will be immediately reported as a cheating case.
5. In case you have any doubts, ask your proctor by using raise hand option. Any important announcement will be made verbally by your proctor.
6. You can leave the exam room only once during the entire duration after securing permission from your proctor/TA. You cannot turn off your camera during this time and you must leave your electronic devices in front of the camera.

Problem 1. Solve the following recurrence relation *using recursion trees*

$$T(n) = \begin{cases} 3T(n/9) + \sqrt{n} & \text{if } n > 1 \\ 1 & \text{otherwise} \end{cases}$$

Problem 2. Suppose that there are k sorted arrays, $A_1, A_2, \ldots, A_k$ each of size n. Give an order $O(nk \log k)$-time algorithm to count the size of $\{(x, y) \mid x \in A_i, y \in A_j \text{ such that } i < j \text{ but } x > y\}$. Give the pseudocode and argue the running time. Nothing else is required. But the pseudocode should be clear enough. Use English sentences withing the pseudocode if something is too complicated. You can assume k is a power of 2.

Problem 3. Suppose you are at the ground floor of a building with n floors inside an elevator. Your friends are hiding at different floors of the building (more than one of your friends can hide on the same floor). Now, let us say a game starts exactly at time $t = 0$. At time $t = i, i = 1, 2, \cdots n-1$, your friend i will reveal himself/herself momentarily. If the elevator is on that particular floor at time $t = i$, then your friend has to get in to the elevator - essentially you catch him/her !. Otherwise, you have missed him/her for good and they will not reveal themselves again. The final goal is to catch as many of your friends as possible and finally take them to floor n by time $t = n$ (this is very important !). Throughout the game you can take the elevator wherever you want. You can also wait at a floor if you want. Assume

- The elevator moves one floor (in either direction) in 1 unit of time.
- Nobody gets out of the elevator before $t = n$

You are given an array $floor[]$ of size n where $floor[i]$ contains the floor number of friend i. Design a polynomial time algorithm to catch as many of your friends as possible and take them to the top floor by $t = n$.

Problem 4. For the following two problems, you need to find *examples which show that the given greedy algorithms* fail to find the optimal solution.

a) Consider the interval coloring problem. You are given a set of n intervals $I_1, I_2, \cdots I_n$ with s_i, e_i being the start and end time of interval $i = 1, 2, \cdots n$. The task is to assign a color (or a number, for simplicity) to each interval so that two overlapping intervals do not get the same color. The objective is to use the minimum possible number of colors.

   Suppose the list is sorted according to **non-decreasing end-time**. We maintain a palette of colors. We start by giving color 1 to the first interval and adding it to the palette. At any iteration $1 < i \leq n$, we assign the **lowest index** color admissible from the current palette. If none of the colors in the palette is admissible for interval i, then we add a new color to the palette and assign this color to i. Find an example where this algorithm will not give you the optimal solution.

b) Suppose you are auctioning a set of n indivisible items and you have a set of n buyers. Every buyer $i = 1, 2, \cdots n$ bids a price p_{ij} for every item $j = 1, 2, \cdots n$. Your goal is to sell exactly one item to each buyer so as to maximize the total money you earn. A natural greedy approach is the following - assign item j to buyer i such that p_{ij} is the maximum among all possible pairings i, j. Remove item j and buyer i from the list and continue. Find an example where this does not give you the optimal solution.

Problem 5. Describe an implementation of the Interval Partitioning (also known as Interval Coloring) algorithm done in the lecture in $O(n \log_2 n)$-time. More precisely, there are two parts to this algorithm - first you need to do some sorting and then you run a greedy loop. Design your algorithm in a way that the main loop takes only $\mathcal{O}(n)$ time. You can use additional data structures.

(First write the pseudocode for the naive algorithm done in lecture. This will fetch you 20%. Next describe how you modify the naive implementation to achieve the required running time).