Minor II (COL 106)

Total marks: 30 21 August 2020, 11AM - 12 Noon.
Pages: 7

Answer all questions

Guidelines

- Read the complete guidelines posted on Piazza carefully, and follow them.
- All answers have to be handwritten (by yourself), using either blue or black pen.
- Begin each answer in a new page.
- Before you start answering the questions, write the honor-code (given in Piazza and reproduced below for convenience) by hand, sign to declare that you understand and follow it.
- Make appropriate and meaningful assumptions that may be necessary to answer questions, and mention them explicitly in your answer.
- We are aware of some websites which may provide you the final solution for a given question. But their use is prohibited. So if you just write the final solution, or replicate the answer from those websites, no marks will be awarded. Further, this will be considered as plagiarism, attracting disciplinary action.

Honor-code

> I, <your name>, acknowledge the IITD Honor Code, and confirm that the submission below is entirely my own except where I have cited the source clearly. I have not:
>
> 1. used the services of any person, organization or discussions during the examination in preparation of this open-book exam answers.
> 2. given or received assistance in either accessing the question paper, or providing specimen answer/hint/diagram/model/code to or from other candidates submitting this open-book exam.
>
> I am fully aware of the fact that if found to have used unfair and disallowed practices in the examination, I am liable for strict action which could include receiving a failure grade in this course, as well as other actions as deemed fit by the institute disciplinary committee.
> Entry number:
> Mobile number:
> Signature:
> Date:

1. (6 points) Raman is a physicist. He has just started taking an interest in quantum computing. For that he started learning programming. He studied the randomized *quicksort* algorithm and started coding his first sorting algorithm. While coding the algorithm, he came up with a brilliant idea. He designed an algorithm that could find the $\lceil n/t \rceil^{th}$ largest element in an n element array (for any constant t). Since he is good at mathematics, he found the running time of his algorithm to be $\Theta(n)$. So instead of randomly choosing the pivot element, he used this algorithm to choose the pivot element in the quicksort. He claims that the worst-case time complexity of his *modified quicksort* algorithm is better than $O(n^2)$. Raman has challenged you to prove him wrong. Suppose t is defined as: 2 + last two digits of your entry number (for example if your entry number is: 2017CSZ7015, then $t = 2 + 15 = 17$).

   (a) (3 points) Derive the worst-case running time of the *modified quicksort* algorithm. Show all the steps.

   (b) (3 points) Consider a hypothetical scenario where the time complexity of finding the pivot element is $\mathcal{T}$ (independent of n) instead of $\Theta(n)$. Then state the running time of the *modified Quicksort* algorithm in terms of n and $\mathcal{T}$.

   Give proper reasoning for both parts.

2. (5 points) We have seen AVL trees in the class which are balanced by using various kinds of rotations. In this question you need to "dry-run" various insertions in the AVL tree. Consider the left to right sequence of characters in: **P18CMK47**, i.e., the sequence of characters 'P', '1', '8', 'C', 'M', 'K', '4', '7'.

   For the next two sub-questions, assume that the collation order of characters is as per their ASCII code (that is, the digits come before the alphabets and both are in their natural order. e.g. '1' < '9' < 'A' < 'P'.).

   (a) (3 points) Demonstrate the sequence of AVL trees generated after inserting each character of the above given sequence. Do show separately, the rotations required after each insertion to balance the tree. At each step, point out which node(s) are unbalanced, and how the rotation has restored the balance invariant. Marks will be awarded only if the steps as well as the explanation are correct.

   (b) (2 points) Suppose we select characters of your entry number in random order, and insert them in an AVL tree. Would the AVL tree be always the same? If yes, prove it. Otherwise give a counterexample (just show the orderings and final AVL trees).

3. (5 points) The *magic number* of an array X is defined as the number of ordered pairs $(X[a], X[b])$ s.t. index $a < b$ but $X[a] > X[b]$. Given an array $X = [2, 1, 5, 4, 7, 3, 9, 10]$, compute the *magic number* of array X **using a modified version of Merge Sort algorithm only**. Clearly show all the steps in your computation and give a complexity of your proposed algorithm in terms of array size n (for a general array X of size n). You don't need to write pseudo-code but your explanation of computation steps should make use of the modified version of Merge Sort algorithm.

   Hint: You are only allowed to modify either the divide-step or the combine-step of the Merge Sort.

4. (5 points) Consider the **distinct digits** of your 10-digit mobile number in the order they appear as list X. For e.g. if your mobile number is 9941131411, then X = [9, 4, 1, 3]. This list has d elements. If $d < 5$ then extend this list to length 5 by choosing the first $5 - d$ elements from the list Y = [21, 11, 22, 12, 23].

   Consider the first 5 elements of X. Count the total number of distinct Binary Search Trees (BSTs) that can be constructed using these 5 elements. You are **not allowed to use any direct formula** for this, but you need to analytically find the answer and show your calculations clearly.

5. (4 points) You were taught the **linked list ADT** in class. Consider 4 sorting algorithms for sorting a list - (i) selection sort, (ii) insertion sort, (iii) merge sort and (iv) quick sort. Which algorithm would you prefer to **sort a singly linked list**? Why? Write the pseudocode of proposed method mentioning the time complexity of each step of your algorithm. You need to clearly give a comparison (in terms of time and space complexity) of your choice with other algorithms while answering why those may not be efficient for sorting a singly linked list.

6. (5 points) A binary search tree is sub-class of binary trees in which the left child is smaller and the right child is larger than the parent. Given a binary search tree give an algorithm (pseudo-code) to find the Kth smallest element.

END OF THE QUESTION PAPER