CS2040S: Data Structures and Algorithms

Discussion Group Problems for Week 4

For: Feb. 10–Feb. 4

1 Check in and PS3

Discuss questions, if you have any, with the tutor and the rest of the class, about the material and content so far.

2 Problems

Problem 1. Review AVL tree insertion and deletion algorithms in class. Work through an example that also covers the cases for rotations.

Problem 2. (A few puzzles with duplicates and missing elements.)
Assume you have an *unsorted* array containing $n-1$ unique elements in the range from $[1, n]$. How can you find the missing element? What if there are k missing elements? e.g. Given $[5, 2, 1, 4, 7, 3]$, we should return 6 here. (Hint: The partitioning algorithm would be quite useful here.)

Problem 3. (Chicken Rice) Imagine you are the judge of a chicken rice competition. You have in front of you n plates of chicken rice. Your goal is to identify which plate of chicken rice is best, but it's only possible for you to compare when you eat the two plates in close succession.

Problem 3.a. A simple algorithm:

- Put the first plate on your table.
- Go through all the remaining plates. For each plate, taste the chicken rice on the plate, taste the chicken rice on the table, decide which is better. If the new plate is better than the one on your table, replace the plate on your table with the new plate.
- When you are done, the plate on your table is the winner!

Assume each plate begins containing $n-1$ bites of chicken rice. When you are done, in the worst-case, how much chicken rice is left on the winning plate?

Problem 3.b. Oh no! We want to make sure that there is as much chicken rice left on the winning plate as possible (so you can take it home and give it to all your friends). Design an algorithm to maximize the amount of remaining chicken rice on the winning plate, once you have

completed the testing/tasting process. How much chicken rice is left on the winning plate? How much chicken rice have you had to consume in total? Here we assume deliciousness is transitive.

Problem 3.c. Now I do not want to find the best chicken rice, but (for some perverse reason) I want to find the median chicken rice, i.e. the chicken rice that is the $\frac{n}{2}^{th}$ best, amongst the n. Again, design an algorithm to maximize the amount of remaining chicken rice on the median plate, once you have completed the testing/tasting process. How much chicken rice is left on the median plate? How much chicken rice have you had to consume in total? (If your algorithm is randomized, give your answers in expectation.)

Problem 4. How much space does MergeSort as presented in class take? How can you improve this?

Problem 5. (**Order Maintenance**)

The goal of the order maintenance problem is to maintain a total order over some (unspecified) objects. The data structure supports two operations:

- `InsertAfter(A, B)`: insert B immediately after A;
- `InsertBefore(A, B)`: insert B immediately before A;
- `isAfter(A, B)`: is B after A in the total order?

Notice that the insert operation adds B directly after A, while the query operation `isAfter(A, B)` asks whether B is anywhere after A in the total order. The expected complexity of each operation is $O(\log n)$, where n is the number of items in the data structure. Also, for this question, you're allowed to assume that when `isAfter(A, B)` is called, you're given the nodes corresponding to A and B.

Problem 6. (**Ancestor Queries**)

Our job now *simulate* a binary tree (not necessarily balanced). Each node has zero, one, or two children, and the tree is of height h. Unfortunately, it is not a balanced tree. We want to support the following operations:

- `InsertLeft(x, y)`: insert y as a left child of x in the binary tree.
- `InsertRight(x, y)`: insert y as a right child of x in the binary tree.
- `isAncestor(x, y)`: is x an ancestor of y in the binary search that contains them?

Problem 7. **(If you have time)** What solutions did you find for Contest 1 (Catch the Spies)?