CS 161 (Stanford, Winter 2024) Homework 4

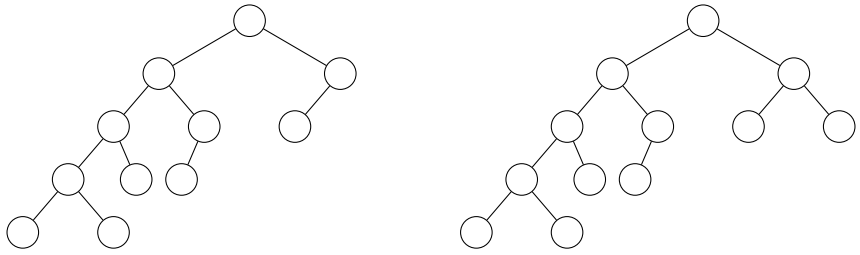
Style guide and expectations: Please see the "Homework" part of the "Resources" section on the webpage for guidance on what we look for in homework solutions. We will grade according to these standards. You should cite all sources you used outside of the course material.
What we expect: Make sure to look at the "**We are expecting**" blocks below each problem to see what we will be grading for in each problem!

Exercises. The following questions are exercises. We suggest you do these on your own. As with any homework question, though, you may ask the course staff for help.

1 Exercise: Red-Black Tree (8 pt.)

For each of the following examples, if the nodes can be colored red or black to make a legitimate red-black tree, then give such a coloring. If not, then say that they cannot.

(You can make use of the latex code in this template file below to color the nodes red or black. You are also welcome to take a screenshot and then color the trees in a graphics tool of your choice, or make a hand-drawn copy and take a photo.)



[We are expecting: For each tree, either an image of a colored-in red-black tree or a statement "No such red-black tree." No justification is required.**]**

2 Exercise: Radix Sort

Recall from lecture that `RadixSort` runs in $O(d(n + r))$ time, where we run the stable `CountingSort` algorithm (which takes $O(n + r)$ time) a total of d times.

2.1 (3 pt.)

Suppose we want to use `RadixSort` to sort a list of lowercase words W in alphabetical order. All words can be padded with '%' characters (assume '%' comes before 'a' alphabetically) to make them the same length. For the following W, describe what the three variables d, n, and r refer to (your explanation should include some reference to this W or its elements) and what their particular values would be for this problem.

$$W = [\texttt{the, quick, brown, fox, jumps, over, the, lazy, dog}]$$

(This isn't the correct answer, but to give you an example of what we want, you might say "n is the length of the first word in the list, and n would have a value of 3 for this list W".)

[We are expecting: Descriptions and concrete values for each of d, n, and r.**]**

2.2 (3 pt.)

You happen to have the date (day, month, year) that each word was first mentioned during a CS 161 lecture. You want to now sort all the words in W first by date, then use alphabetical ordering to break ties. You still want to use `RadixSort`, but you need to augment each of the original words in W with the date information (represented as numbers) by prepending, appending, or somehow inserting these date numbers somewhere in the string. Months are represented using 2 digits ($01, 02, \ldots, 11, 12$), days are also 2 digits ($01, 02, \ldots, 30, 31$), and years are 4 digits (e.g. 2014). Assume the digits 0-9 are alphabetically smaller than the '%' character and a-z. Describe your augmentation procedure and write the string you would use to represent the word "fox", which was first mentioned on October 24, 1989.

[We are expecting: A description of your augmentation procedure, and a string.**]**

Hint: Make sure you remember to appropriately incorporate the '%' character (note that the maximum length of any word here is 5)!

Problems. The following questions are problems. You may talk with your fellow CS 161-ers about the problems. However:

- Try the problems on your own *before* collaborating.
- Write up your answers yourself, in your own words. You should never share your typed-up solutions with your collaborators.
- If you collaborated, list the names of the students you collaborated with at the beginning of each problem.

3 Can it be done?

For each of the following tasks, either explain clearly how you would accomplish it (pseudocode is optional), or else explain clearly why it cannot be done (e.g., by explicitly reaching a contradiction and/or via counterexample) in the worst case. If you explain how to accomplish it, you do not need to prove that your algorithm works. You may cite any result or algorithm we have seen in class, but don't make any assumptions that are not explicitly stated. All running times refer to deterministic worst-case running time.

3.1 (5 pt.)

Task: Let i and j be integers such that $1 \leq i < j \leq n$. Given an array A of n distinct arbitrary comparable elements, find the elements in A that are strictly larger than the i-th smallest element and strictly smaller than the j-th smallest element and return them in sorted order, in time $O(n)$.

[We are expecting: Can it be done? Either explain clearly how you would accomplish it (pseudocode is optional), or else explain clearly why it cannot be done in the worst case (e.g. by explicitly reaching a contradiction and/or via counterexample).**]**

3.2 (5 pt.)

Design an algorithm that takes as input an array of A of n distinct arbitrary comparable items and outputs a valid BST containing the same n elements in time $O(n)$.

[We are expecting: Can it be done? Either explain clearly how you would accomplish it (pseudocode is optional), or else explain clearly why it cannot be done in the worst case (e.g. by explicitly reaching a contradiction and/or via counterexample).**]**

3.3 (5 pt.)

You are given an array A containing n distinct very large positive integers that are at most d digits long (assume $d = O(n)$). However, the integers are really sparse and have only 2 nonzero digits (we'll call these "2-sparse integers"), so each number is represented as follows:

Suppose x is a 2-sparse integer with (nonzero) x_j as the j-th least significant digit of x, and (nonzero) x_k as the k-th least significant digit of x, and $d \geq j > k \geq 1$. Then the succinct representation of x is $((x_j, j), (x_k, k))$.

Design an algorithm that sorts the array A containing the n succinctly-represented numbers in time $O(n)$. You may assume that all digits are between 0 and 9, inclusive.

Important tip: If you refer to the succinct representation of numbers, please use the same notation above in your answer, so graders can understand your response easily: write x_j when referring to the nonzero j-th least significant digit of x and write x_k to represent the nonzero k-th least significant digit of x. For example, this kind of wording is clear to us: "sort all elements in A while using each element's x_k value as the key."

Examples:

- Inputs: $n = 2$, $A = [((3, 4), (2, 1)), ((8, 2), (7, 1))]$ ($d = 4$ in this case)
- Output: $[((8, 2), (7, 1)), ((3, 4), (2, 1))]$
- Reason: The numbers in A are $[3002, 0087]$. When sorted, we get $[0087, 3002]$.

- Inputs: $n = 2$, $A = [((3, 5), (7, 4)), ((2, 5), (9, 4))]$ ($d = 5$ in this case)
- Output: $[((2, 5), (9, 4)), ((3, 5), (7, 4))]$
- Reason: The numbers in A are $[37000, 29000]$. When sorted, we get $[29000, 37000]$.

[We are expecting: Can it be done? Either explain clearly how you would accomplish it (pseudocode is optional), or else explain clearly why it cannot be done in the worst case (e.g. by explicitly reaching a contradiction and/or via counterexample).**]**

4 Exploration of Ternary Search Trees

In the context of this discussion, our focus is on ternary search trees (TSTs), which are an extension of binary search trees. Instead of having two child nodes (left and right), TSTs include a third pointer, resulting in left, middle, and right child nodes.

A TST, denoted as T, is defined as a ternary tree where each node encapsulates a coordinate pair (x, y), with x, y belonging to the set of integers $\mathbb{Z}$. It is imperative that within T, no two coordinate pairs share the same x value and similarly, no identical y values are found. Consequently, for any node u holding the key (x, y) in a TST, the subsequent conditions are maintained:

- Any point (x_L, y_L) in the left subtree of u will satisfy $x_L < x$ and $y_L < y$.
- Any point (x_M, y_M) in the middle subtree of u will satisfy $x_M > x$ and $y_M < y$.
- Any point (x_R, y_R) in the right subtree of u will satisfy $x_R > x$ and $y_R > y$.

Clarification: It is assumed for any node v within the TST T, the count of nodes within its subtree can be determined in constant time $O(1)$ (for instance, this data is stored within the TST structure).

4.1 (3 pt.)

Given n unique coordinate pairs where no two pairs share the same x or y values, prove that it is possible to construct a TST T encapsulating these n pairs.

[We are expecting: A logical argument or construction method that demonstrates the existence of such a TST, assuming the uniqueness of the x and y values of the coordinate pairs.**]**

4.2 (3 pt.)

Prove or disprove whether the TSTs always exist with a height of at most $O(\log(n))$.

(Hint: Consider if there exists a configuration of points resulting solely in a tree height of $\Omega(n)$?)

[We are expecting: An argument with equivalent depth and rigor as seen in our Red-Black Tree height discussion in class.**]**

4.3 (8 pt.)

Given a coordinate pair (x', y') and a TST T containing n points with height h, the objective is to ascertain the presence of (x', y') within T. Elaborate an efficient methodology for this inquiry. Substantiate the algorithm's validity and appraise its computational efficiency with respect to n and h.

[We are expecting: The pseudocode AND a concise description in English of your approach. Provide a solid proof of accuracy along with a succinct runtime analysis that convincingly elucidates your argument to the evaluator.**]**

5 The Enigmatic Axolotl

Andy, the enigmatic axolotl, possesses knowledge of an array A with a length of n, such that $A[i] \in \{1, \ldots, k\}$ for all i (the elements of A may not be unique). While you cannot directly see the list, Andy allows you to ask any yes/no questions about it. For instance, you could inquire, "If I remove $A[5]$ and exchange $A[7]$ with $A[8]$, would the array then be sorted?" or "Do axolotls enjoy the same food as koalas?"

Equipped with paper and pencil, your task is to deduce all elements of A in their sorted arrangement.[1] Andy requires one berry for each question you pose, but you may take your time to perform any calculations on paper based on Andy's responses.

5.1 Sorting algorithm (4 pt.)

Propose an algorithm that sorts array A utilizing $O(k \log n)$ berries, given that you are aware of n and k, although this knowledge is not essential.

[We are expecting: The pseudocode and a straightforward English description of its function. An explanation of why the algorithm expends $O(k \log n)$ berries is required. You do not need to demonstrate the correctness of your algorithm.**]**

5.2 Comparison Bounds (6 pt.)

Demonstrate that any method to resolve this challenge necessitates at least $\Omega(k \log \frac{n}{k})$ berries.

[We are expecting: A compelling and mathematically sound argument (which does NOT imply a need for verbosity). The expected rigor should be akin to that of the sorting lower bounds discussed in class.**]**

[1] It's important to note that you cannot alter array A itself; you can only question Andy about it.

6 The Hiring Committee

The highly prestigious University of West Palo Alto is hiring a new computer science professor, and you're on the hiring committee. The committee is committed to hiring the best researcher and teacher they can find, someone who will have world-class research output and be an inspiration to their students.

The committee has received hundreds of applications, and the chair of the committee wants you to rank the applicants from best to worst so the committee can decide who to interview. You have been given a list of applicants that includes, for each applicant, the number of research papers they have published and their average score on teaching evaluations.

6.1 Imperfect proxy (4 pt.)

Pick one of the measures you have been given (number of research papers or average teaching evaluation score) and give at least one reason why that measure might not be a perfect proxy for the value it is intended to track.

[We are expecting: A few sentences in which you clearly state which measure you have chosen, the value that measure is intended to track, and at least one reason why that measure might not perfectly track that value.**]**

6.2 Insufficient information (4 pt.)

Using terminology introduced in class, explain why the list you have been given is insufficient to rank the applicants from best to worst *even if we assume* that number of research papers is a perfect proxy for research promise and average teaching evaluation score is a perfect proxy for teaching skill.

[We are expecting: A few sentences that answer the question using terminology introduced in class. Your answer should define the relevant terminology and explain how it applies to this case.**]**