Assignment 6

Algorithms, Spring 2023

Honor code: *Work on this assignment alone or with one partner. Between different teams, collaboration is at level 1 [verbal collaboration only]. There are lots of resources online, such as animations, visualizations, practice problems, videos, and solutions— which you are encouraged to explore to deepen your understanding. However, you must be careful not to search for the specific problems in the assignment with the intent of getting hints for the solution. Searching for the assignment problems on the internet violates academic honesty for this class.*

---

1. **Majority element via divide-and-conquer:** Suppose we are given an array A of length n with the promise that there exists a majority element (i.e. an element that appears $> \frac{n}{2}$ times). Additionally, we are only allowed to check whether two elements are equal (no $>$ or $<$ comparisons between the elements). Design an $On \lg n)$ algorithm to find the majority element, using divide-and-conquer. Explain why your algorithm works correctly (if a majority element exists, it will find it; otherwise, it will report than none exists); and the runtime of your algorithm.

   [*We expect: pseudocode, an English description of the main idea of the algorithm, an (informal) explanation of why it's correct; and running time analysis.*]

2. **The inversion problem:** Let $A[1..n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an inversion of A. Give an algorihm that determines the number of inversions in an array in $O(n \lg n)$ time worst-case (Hint: modify merge sort).

   We expect: pseudocode and a clear English description of what the algorithm is doing; A brief informal justification of why the algorithm is correct, and its runtime analysis.

   Helper exercises (do not turn in).

   (a) List the inversions of the array $< 2, 3, 8, 6, 1 >$.

   (b) What array with elements from the set $\{1, 2, ..., n\}$ has the most inversions? How many does it have?

(c) Give an algorithm that determines the number of inversions in an array in $O(n^2)$ time (this is the straightforward solution).

3. You're interviewing for your dream job at an echological ethical tech company with healthy snacks. You already passed 25 stages of interviews, and your final interviewer asks you to design a binary search tree data structure that performs INSERT operations in $O(\sqrt{\lg n})$ time using a comparison-based algorithms. Design such a structure, or prove that it is impossible.

   [**We expect:** If possible: An English description of the algorithm and a running time analysis. If impossible: Justification that this is impossible.]

Evaluation

The assignment will be evaluated along several criteria:

1. **Correctness**: Is your solution correct?

2. **Justification**: Is your answer justified?

3. **Style**: Does it look professional and neat? Is the explanation written carefully in complete sentences, and well-organized logic? Is it easily human-readable? Is it easy to understand?

   - Assignments should be typed. Feel free to annotate the pdf to add figures and formulas which are too time-consuming to type.
   - Write each problem on a separate page or leave plenty of space between problems so that we can write comments.
   - Try to put yourself in the position of the reader. If you hadn't been thinking of this problem for 3 hours, would your answers make sense to you?
   - Try to finish the assignment early, then step away for a day or two, and then come back to it and read it again. Chances are you'll find something you can write more clearly.
   - Look at posted solutions for style advice (if solutions are not posted, ask).