Algorithms and Data Structures 2023: Assignment 1, due Sept. 12

Please read the assignment policies on the course homepage before starting the assignment. If you are working in a group, each group must turn in a single assignment. Please write with each question who wrote the solution to the question. I expect the load to be roughly balanced, i.e., each group member must write about half the solutions.

---

1. Background reading: Chapter 2 and Section 3.1 from CLRS. This is not part of your assignment, but you should already know the material covered: reading pseudocode, the use of loop invariants in algorithm analysis, RAM model, worst-case analysis, basic sorting algorithms, and asymptotic notation.

2. Reading: (i) Sections 4.2, 4.3, 4.4 of CLRS. (ii) Sections 15.2, 15.3 of CLRS.

3. (15 marks) Solve the recurrences: (i) $T(n) = 2T(n/2) + n \log n$, (ii) $T(n) = 7T(n/3) + n^2$, (iii) $T(n) = \sqrt{n}T(\sqrt{n}) + n$.

4. (5 marks) Give the best upper bounds you can on the nth Fibonacci number F_n, where $F_n = F_{n-1} + F_{n-2}$ and $F_1 = F_2 = 1$.

5. (10 marks) Consider two sets A and B, each having n integers in the range from 0 to $10n$. We wish to compute the *Cartesian sum* of A and B, defined by

   $$C = \{x + y : x \in A, y \in B\}.$$

   Note that the integers in C are in the range 0 to $20n$. We want to find the elements in C and the number of times each element of C is realized as a sum of elements in A and B. Give an algorithm that solves the problem in $O(n \log n)$ time, and prove correctness.

6. (20 marks) Define $[n] := \{1, 2, \ldots, n\}$. You are given n, and oracle access to a function $f : [n] \times [n] \to [n] \times [n]$ that takes as input two positive integers of value at most n, and returns two positive integers of value at most n. Let $f_1(x_1, x_2)$ and $f_2(x_1, x_2)$ be the first and second coordinates of $f(x_1, x_2)$, respectively. You are also told that f_i is monotone nondecreasing in coordinate i when coordinate $3 - i$ is kept fixed, and monotone nonincreasing in coordinate $3 - i$ when coordinate i is kept fixed. That is, given $x_1 \leq x'_1 \in [n]$ and $x_2 \leq x'_2 \in [n]$, $f_1(x_1, x_2) \leq f_1(x'_1, x_2)$, and $f_1(x_1, x_2) \geq f_1(x_1, x'_2)$. Similarly, $f_2(x_1, x_2) \geq f_2(x'_1, x_2)$, and $f_2(x_1, x_2) \leq f_2(x_1, x'_2)$.

   The problem is to find a *fixed point* of the function, i.e., values $x_1, x_2 \in [n]$ so that $f(x_1, x_2) = (x_1, x_2)$. Give an algorithm that given n and oracle access to such a function f, finds a fixed point of f in time $O(\text{poly}(\log n))$. You must also give a proof of correctness, and running time analysis.

7. (15 marks) A *palindrome* is a nonempty string over some alphabet that reads the same forward and backward. Examples of palindromes are all strings of length 1, `civic`, `racecar`, and `aibohphobia`. Give an efficient algorithm, with proof of correctness and run-time analysis, to find the longest palindrome that is a subsequence of a given input string. For example, given the input string `character`, your algorithm should return `carac`.

8. The purpose of this question is to extend the closest-points algorithm seen in the first lecture, to give an $O(n \log^2 n)$ algorithm for finding the closest pair of points in 3 dimensions. All points in this question are in $\mathbb{R}^3$.

(a) (5 marks) Prove that, if all points are at least distance δ apart, a cube with each dimension of size 2δ contains at most a constant (say k) number of points.

(b) (10 marks) You are now given 2 sets of points S_1 and S_2, each containing n points. The distance between any pair of points in S_1 is at least δ, and further, each point in S_1 has z-coordinate in $[0, \delta]$. Similarly, the distance between any pair of points in S_2 is at least δ, and each point in S_2 has z-coordinate in $[-\delta, 0]$.

Extend the algorithm discussed in class to give an $O(n \log n)$-time algorithm for finding the closest pair of points in $S_1 \cup S_2$. Note that, by the first part of the question, any cube with each dimension at most 2δ, contains at most $2k$ points from $S_1 \cup S_2$.

(c) (10 marks) Given a set S of n points in $\mathbb{R}^3$, now give an $O(n \log^2 n)$-time algorithm to find the closest pair of points.