Algorithm Design and Analysis (Spring 2022)
Assignment 1
Deadline: Mar 20, 2022

1. (20 Points) Prove the following generalization of the master theorem. Given constants $a \geq 1, b > 1, d \geq 0$, and $w \geq 0$, if $T(n) = 1$ for $n < y$ and $T(n) = aT(n/b) + n^d \log^w n$, we have

$$T(n) = \begin{cases} O(n^d \log^w n) & \text{if } a < b^d \\ O(n^{\log_b a}) & \text{if } a > b^d \\ O(n^d \log^{w+1} n) & \text{if } a = b^d. \end{cases}$$

2. (20 points) Recall that Merge Sort divides the sequence into two subsequences with nearly the same size and sort them recursively. What if we want the two subsequences to have different sizes? Please use the *one third dividing approach* (dividing the sequence $a_1, a_2, \ldots, a_n$ into $a_1, a_2, \ldots, a_{\lceil n/3 \rceil}$ and $a_{\lceil n/3 \rceil+1}, a_{\lceil n/3 \rceil+2}, \ldots, a_n$) to complete the *One Third Merge Sort Algorithm*, and analyze its time complexity.

3. (30 points) Choose **one** of the following two questions to solve.

   (a) Given an array of n integers $x_1, x_2, ..., x_n$, there are queries of the following form: given an integer $1 \leq k \leq n$, you need to return the smallest k integers in the array (in no particular order). Design an $O(n)$ time preprocessing algorithm so that you can answer each query in $O(k)$ time.

   Notice that a trivial $O(n \log n)$ time preprocessing algorithm is to sort the array, and each query is answered by just returning the first k integers, which requires $O(k)$ time.

   (b) Given an array of n integers $x_1, x_2, \ldots, x_n$, design an algorithm that outputs *the second largest integer*. Your algorithm is required to make at most $n + \log n$ comparisons. (Notice that $n + \log n$ is the exact number, and there is no asymptotic notation here.) You can assume $n = 2^k$ for some $k \in \mathbb{Z}^+$.

4. (30 points + 5 points for bonus) For two vectors $\mathbf{a} = (a_1, \ldots, a_d), \mathbf{b} = (b_1, \ldots, b_d) \in \mathbf{R}^d$, we say $\mathbf{a}$ is *greater than* $\mathbf{b}$ if $a_k > b_k$ for each $k = 1, \ldots, d$. You are given two collections of vectors $A, B \subseteq \mathbb{R}^d$. The objective is to count the number of pairs $(\mathbf{a}, \mathbf{b}) \in (A, B)$ such that $\mathbf{a}$ is greater than $\mathbf{b}$. You can assume all the entries in all the vectors are distinct. Let $n = |A| + |B|$ and d be the dimension of the vectors.

   (a) (10 points) Design an $O(n \log n)$ time algorithm for this problem with $d = 1$.

   (b) (20 points) Design an algorithm for this problem with $d = 2$. Your algorithm must run in $o(n^{1.1})$ time.

   (c) (5 points, bonus) Generalize the algorithm in part (b) so that it works for general d. Analyze its running time. The running time must be in terms of n and d.

5. How long does it take you to finish the assignment (including thinking and discussion)? Give a score (1,2,3,4,5) to the difficulty. Do you have any collaborators? Please write down their names here.