CS345: Design and Analysis of Algorithms

Assignment 1

Due Date: 23rd August

Total Number of Pages: 2 | Total Points 25

Instructions-

1. For submission typeset the solution to each problem and compile them in a single pdf file. Hand-written solutions will not be accepted. You can use LaTeX or Word for typesetting.
2. Start each problem from a new page. Write down your Name, Roll number and problem number clearly for each problem.
3. For each question, give the pseudo-code of the algorithm with a clear description of the algorithm. Unclear description will receive less marks. Less optimal solutions will receive only partial marks.
4. Assume that sorting would have $O(n \log n)$ complexity.

1 Problems that need to be submitted

Question 1. (10 points) Given two sets $\{p_1, p_2, \ldots, p_n\}$ and $\{q_1, q_2, \ldots, q_n\}$ of n points on the unit circle, connect each point p_i to the corresponding point q_i. Describe and analyze a divide-and-conquer algorithm to determine how many pairs of these line segments intersect in $O(n \log^2 n)$ time.

Question 2. The concept of non-dominated points can be extended to three dimensions. However, applying a divide-and-conquer strategy to compute non-dominated points in 3D is not straightforward. One might consider reducing the problem to three instances of a 2-dimensional problem by projecting the points onto the xy, yz, and xz planes. However, this approach is incorrect (take some time to think about why). Interestingly, there exists a simple and elegant algorithm that uses a basic data structure to compute non-dominated points in 3D. The purpose of this exercise is to help you realize this fact.

(a) (7 points) **Online Algorithm for 2D Non-Dominated Points:** Design an algorithm that receives n points in the xy-plane one by one and maintains the non-dominated points in an online manner. Upon insertion of the i^{th} point, the algorithm should update the set of non-dominated points in $O(\log i)$ time. Note that it is acceptable if your algorithm guarantees a bound of $O(i \log i)$ on the total time for the insertion of i points. It is not necessary for your algorithm to achieve an $O(\log i)$ bound on the processing time for the insertion of the i^{th} point.

(b) (8 points) **Algorithm for 3D Non-Dominated Points:** Design an $O(n \log n)$ time algorithm to compute the non-dominated points of a set of n points in 3D. You must carefully use the result from the previous part in your design.

2 Ungraded Problems (Do not submit them)

Question 1. Given two positive numbers n and m, your task is to design a $O(n \log m)$ time complexity algorithm to compute the nth root of m (i.e. $\sqrt[n]{m}$) using a **Divide and Conquer** approach. Alternatively, your algorithm should find the value of x such that x^n is equal to m. Assume that nth root of m is a natural number.

Question 2. Consider a fence F made up of small brick walls of length n, where each unit of the fence corresponds to a small wall in the fence. Some of these walls are reinforced with concrete. We have a k sized array R containing the locations (indices) of the reinforced walls.

1. If the current length of the fence is at least 2, you may divide the fence into two halves and compute the power required to destroy each half separately.
2. Alternatively, you may destroy the entire fence at once. The power required for this action is calculated as follows:
   - If the fence contains no reinforced walls, the power required is α.
   - If the array contains reinforced walls, the power required is $\beta \times l \times s$, where s is the total number of reinforced walls in the fence, and l is the length of the fence.

Your task is to design an algorithm that computes the minimum power required to destroy the entire fence in $O(k \log k \log n)$ time.

Assume that n is always a power of 2 and α, β are constants.

Question 3. Given a polygon (not necessarily convex) P on n points, and a point α, design an $O(n)$ time algorithm to check if α is contained in P or not.

Question 4. You are given two sets A and B of integers. Each has size n and the integers are in the range 0 to $10n$. Let $A + B$ denote the cartesian-sum multiset

$$\{a + b \mid a \in A, b \in B\}.$$

Give an algorithm to compute the multiset $A + B$ in $O(n \log n)$ time.