Problem Set 5: Divide and Conquer Algorithm

1. (EASY) Given an array A of integers, find indices i, j such that $A[i] + A[i+1] + A[i+2] + \ldots + A[j]$ is maximized. Design a divide-and-conquer algorithm for finding the indices i, j.

2. (EASY) Given an array A of n numbers, find the maximum difference between two elements in the array. Desgin a divide and conquer algorithm for this problem with a running time $O(n)$.

3. (EASY) Consider the following divide and conquer to find the minimum spanning tree in the graph. Divide the graph into $S, V - S$ where the number the of vertices in S and $V - S$ differ by at most 1. Recursively find the minimum spanning tree M_1 and M_2 in S and $V - S$ respectively. Let e be the minimum weight edge between S and $V - S$. Return $M_1 \cup M_2 \cup \{e\}$ as the answer. Show that the above algorithm is incorrect.

4. (MEDIUM) Given an array A of non-zero numbers, find indices i, j that minimizes

$$\prod_{k=i}^{j} A[k]$$

   Design a divide and conquer algorithm for this problem.

5. (HARD) Given an array of n non-negative integers A, find indices i and j such that $i \leq j$ that maximizes:

$$\left(\min_{k=i,\ldots,j} A[k]\right) \times \prod_{k=i}^{j} A[k]$$

   (a) Assume that we know that the index ℓ is that part of the answer, that is $i \leq \ell \leq j$. Design a greedy algorithm to find the indices i, j in $O(n)$ time.

   (b) Design a divide and conquer algorithm that finds i, j in $O(n \log n)$ time without the assumption mentioned in the previous part.

6. (HARD) You want to find the maximum and the minimum number in the array. There is a simple algorithm that performs $n - 1$ comparisons and finds the maximum number in the array. Symmetrically, finding the minimum also takes $n - 1$ comparisons. Design a divide and algorithm that performs $\leq 3n/2$ comparisons to find both the maximum and the minimum.

7. (HARD) Consider a set P containing n points in the plane, and let k be a parameter. Develop a divide-and-conquer algorithm that efficiently computes the smallest (axis-parallel) square containing exactly k points from P. Your algorithm should have a time complexity of $O(n \operatorname{poly} \log n)$ when k is a constant.

8. (HARD) You are given a sequence $P = \{p_1 < p_2 < \ldots < p_n\}$ of n distinct real numbers. You are also given a set $L = \{f_i(x) = \alpha_i x + \beta_i \mid i = 1, \ldots, n\}$ of n linear functions. Consider the max function $f(x) = \max_i f_i(x)$. Describe a direct (i.e., without computing other structures first) divide-and-conquer algorithm, as fast as possible, that computes the values $y_i = f(p_i)$, for $i = 1, \ldots, n$. [Hint: (I) Draw an example of how the functions f_i and f look like in the xy plane. (II) Compute the function f_i realizing $f(p_{n/2})$.]

9. (MEDIUM) In the cricket world cup in the year 2099, there are n team. The first phase of the world cup is a round robin tournament where each team will play every other team. The first phase must end in $n-1$ days. For example, if there are four team: India, Pakistan, Australia and England, then the first phase can be designed as:

   Day 1: India vs Pakistan, England vs Australia

Day 2: India vs England, Pakistan vs Australia

Day 3: India vs Australia, Pakistan vs England

Your job is to design the above schedule when there are n teams. Design an $O(n^2)$ time divide and conquer algorithm that output the schedule when n is a power of 2. Note that, since you have to output the schedule, the running time of your algorithm must be $\Omega(n^2)$.

10. (HARD) Let P be the set of n points on a plane. A point $p = (a, b)$ is called unodominated if there is no point in first quadrant if the origin is shifted to (a, b). Or in other words, there is no point $q = (c, d)$ such that $c \geq a$ and $d \geq b$.

    Design and analyze a divide and conquer algorithm that finds all undominated points in $O(n \log n)$ time.

11. (MEDIUM) Given an array A of n numbers, $A[i]$ and $A[j]$ are said be in an *inversion* if $A[i] > A[j]$ and $i < j$. Design an algorithm that counts the number of inversion in an array in $O(n \log n)$ time.

12. (HARD) You are given a complete binary tree in which each leaf represents a unique number. Let us assume that there are n leaves in the tree where n is a power of 2. The leaves at the bottom can be viewed as an array. There is an inversion between two leaves a and b if a lies to the left of b and $a > b$. You can decrease the number of inversions by selecting an internal node and swapping its left subtree with the right subtree. Design an algorithm which finds the vertices whose subtree need to be swapped so that the number of inversions is minimized. The running time of your algorithm should be $O(n \log n)$.

13. (EASY) You are given n coins in which exactly one coin is bad. All the good coins have same weight. The bad coin weighs less than the good coin. You are given a balance. Each weighing on the balance tell you whether all the coins on the left side of the balance have (1) more weight, (2) same weight or (3) less weight than all the coins on the right side of the balance. Design an algorithm that uses the balance $O(\log n)$ times and finds the bad coin.

14. (MEDIUM) Redo the above question but now the weight of the bad coin may be more or less than weight of a good coin. Again, use only $O(\log n)$ weighings to find the bad coin.

15. (HARD) You are given a network of communication towers, where each tower is connected to some neighboring towers via bidirectional cables. The cables have varying signal strengths, and the maximum signal strength along any cable between two neighboring towers is provided on a network map.

    Given two towers s and t, your task is to find a path from s to t that minimizes the maximum signal strength along the path. The input is provided in adjacency-list format, where each tower has a list of all cables connecting it to its neighboring towers.

    (a) Design an algorithm that finds the optimal path in $O((n + m) \log n)$ time, where n is the number of towers and m is the number of cables.

    (b) Can you do it in $O(m + n)$ time?

16. (HARD) You are given an $n \times n$ forest, represented as a grid, where each cell in the grid represents a tree, and each tree has a height given by a function $H(i, j)$. The function $H(i, j)$ provides the height of the tree at position (i, j) where $1 \leq i, j \leq n$. Your goal is to find a tree that is the lowest compared to all its immediate neighbors, i.e., a local minimum. A local minimum is a tree at position (i^*, j^*) such that its height is less than or equal to the height of its neighboring trees in the grid. The neighbors of a tree at (i, j) are the trees at positions $(i - 1, j)$, $(i + 1, j)$, $(i, j + 1)$, and $(i, j - 1)$. Trees along the diagonals are not considered neighbors.

    Design an algorithm that finds this local minimum using only $O(n)$ queries to the function $H(i, j)$.

17. (HARD) You are given a $n \times n$ matrix A containing integers having the following property: every row and column in the matrix have numbers in strictly increasing order. Given such a matrix A and a number x, find if $x \in A$ in $O(n)$ time.