Tutorial 2, Design and Analysis of Algorithms, 2021

1. Two sets A and B have n elements each. Assume that each element is an integer in the range $[0, n^{100}]$. These sets are not necessarily sorted. Design an algorithm to check whether these two sets are disjoint in $O(n)$ space and time.

2. Input are the sets $S_1, S_2, \ldots, S_l$, where $l \leq n$. Each of these sets have integers in the range $[0, n^c - 1]$ for a fixed c. Also let $\sum_{j=1}^{l} |S_j| = n$. The goal is to output S_1 in sorted order, then S_2 in sorted order, and so on. Design an $O(n)$-time algorithm for this problem.

3. How would you modify Strassen's algorithm to multiply $n \times n$ matrices in which n is not an exact power of 2? Show that the resulting algorithm runs in time $\Theta(n^{\log_2 7})$.

4. How quickly can you multiply a $kn \times n$ matrix by an $n \times kn$ matrix, using Strassen's algorithm as a subroutine? Answer the same question with the order of the input matrices reversed.

5. (a) Find the number of nodes in the divide and conquer graph for computing FFT of a vector of length n (for simplicity you can assume n to be a power of 2).

   (b) Now find time complexity of the FFT algorithm by only considering the structure of the divide and conquer graph (without solving any recursion).

6. Find 1234×4321 using the FFT algorithm showing its divide and conquer graphs.

7. (a) Describe the generalization of the FFT procedure to the case in which n is a power of 3 (using three subproblems). Give a recurrence for the running time, and solve the recurrence.

   (b) Find 97×68 using the above algorithm showing its divide and conquer graphs.

8. 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
   $C = \{ x + y \mid x \in A, y \in B \}$.
   Note that the integers in C are in the range from 0 to $20n$. We want to find the elements of C and the number of times each element of C is realized as a sum of elements in A and B. Design an $O(n \log n)$-time algorithm to solve this problem.

9. Consider an $n \times n$ grid graph G. (An $n \times n$ grid graph is just the adjacency graph of an $n \times n$ chessboard. To be completely precise, it is a graph whose node set is the set of all ordered pairs of natural numbers (i, j), where $1 \leq i \leq n$ and $1 \leq j \leq n$; the nodes (i, j) and (k, l) are joined by an edge if and only if $|i - k| + |j - l| = 1$.) Each node v of G is labeled with a real number x_v. You may assume that the real numbers labeling the nodes are all distinct. A node v of G is a *local minimum* if the label x_v is less than the label x_w for all nodes w that are joined to v by an edge. You are given such an $n \times n$ grid graph G, but the labeling is only specified in the following *implicit* way: for each node v, you can determine the value x_v by probing the node v. Show how to find a local minimum of G using only $O(n)$ *probes* to the nodes of G. Give a proof of correctness of your algorithm and also prove its time complexity.

10. In the *Josephus Problem*, we start with n people numbered 1 to n around a circle, and we eliminate every *second* remaining person until only one survives. For example, the elimination order for $n = 10$ is $2, 4, 6, 8, 10, 3, 7, 1, 9$, so 5 survives. The problem is to determine the survivor's number, $J(n)$ (in the above example, we have $J(10) = 5$). Design a linear time complexity algorithm for computing $J(n)$.