CSCI3160: Regular Exercise Set 2

Prepared by Yufei Tao

Problem 1 (Faster Algorithm for Finding the Number of Crossing Inversions). Let \( S_1 \) and \( S_2 \) be two disjoint sets of \( n \) integers. Assume that \( S_1 \) is stored in an array \( A_1 \), and \( S_2 \) in an array \( A_2 \). Both \( A_1 \) and \( A_2 \) are sorted in ascending order. Design an algorithm to find the number of such pairs \((a, b)\) satisfying all of the following conditions: (i) \( a \in S_1 \), (ii) \( b \in S_2 \), and (iii) \( a > b \). Your algorithm must finish in \( O(n) \) time (we gave an \( O(n \log n) \)-time algorithm in the class).

Problem 2 (Faster Algorithm for Finding the Number of Inversions). Given an array \( A \) of \( n \) integers, design an algorithm to find the number of inversions in \( O(n \log n) \) time.

Problem 3. Give an \( O(n \log n) \)-time algorithm to solve the dominance counting problem discussed in the class.

Problem 4 (Section 4.1 of the Textbook). Let \( A \) be an array of \( n \) integers (\( A \) is not necessarily sorted). Each integer in \( A \) may be positive or negative. Given \( i, j \) satisfying \( 1 \leq i \leq j \leq n \), define sub-array \( A[i : j] \) as the sequence \((A[i], A[i + 1], ..., A[j])\), and the weight of \( A[i : j] \) as \( A[i] + A[i + 1] + ... + A[j] \). For example, consider \( A = (13, -3, -25, 20, -3, -16, -23, 18) \); \( A[1 : 4] \) has weight 5, while \( A[2 : 4] \) has weight \(-8\).

1. Give an algorithm to find a sub-array of with the largest weight, among all sub-arrays \( A[i : j] \) with \( j = n \). Your algorithm must finish in \( O(n) \) time.

2. Give an algorithm to find a sub-array with the largest weight in \( O(n \log n) \) time (among all the possible sub-arrays).

Problem 5. In the class, we explained how to multiply two \( n \times n \) matrices in \( O(n^{2.81}) \) time when \( n \) is a power of 2. Explain how to ensure the running time for any value of \( n \).