CSCI3610: Special Exercise Set 2

Problem 1. Given an array A of size n, design an algorithm to output all the inversions in A using $O(n \log^2 n + k)$ time, where k is the number of inversions reported.

Problem 2. Prove: if you can solve the dominance counting on n points in $f(n)$ time, then you can count the number of inversions in an integer array of length n in $f(n) + O(n)$ time. (Hint: you can convert the inversion counting problem to an instance of dominance counting.)

Problem 3. Assuming $m \geq n$, give an algorithm to multiply an $m \times n$ matrix with an $n \times m$ matrix in $O(m^2 \cdot n^{0.81})$ time. (Hint: apply Strassen's algorithm to multiply $\lceil m/n \rceil^2$ pairs of order-n matrices.)

Problem 4. Assuming $m \geq n \geq t$, give an algorithm to multiply an $m \times n$ matrix with an $n \times t$ matrix in $O(m \cdot n \cdot t^{0.81})$ time. (Hint: apply Strassen's algorithm to multiply pairs of $t \times t$ matrices.)

Problem 5. Let $A_1, A_2, ..., A_k$ be k arrays, each of which has been sorted. These arrays are mutually disjoint, namely, no integer can appear in more than one array. Design an algorithm to merge the k arrays into one sorted array in $O(n \log k)$ time, where n is the total length of the k arrays. Note: these arrays may have different lengths.