

CSE 3101: Assignment 2

Worth 15%. Due June 20, 2006 at the beginning of class.

The work you submit must be your own. You may discuss problems with each others; however, you should prepare written solutions alone. Copying assignments is a serious academic offence, and will be dealt with accordingly.

Question 1 Throughout this Question, the distance between two points on the two-dimensional plane is taken to be their Mahattan distance, i.e., the distance between two points $A = \langle x_1, y_1 \rangle$ and $B = \langle x_2, y_2 \rangle$ is

$$d(A, B) = |x_1 - x_2| + |y_1 - y_2|$$

- a) Give a divide-and-conquer algorithm that, given a set P of points represented by their x - and y -coordinates, outputs the smallest (Manhattan) distance of all pairs of points in the P . Your algorithm must run in time $n \log n$. You may assume that the points have distinct x -coordinates.
- b) Prove the correctness of your algorithm. (It suffices to show that the combining step in your algorithm is correct.)

Question 2 Consider the “Counting Significant Inversion Problem”. The input is an array of n natural numbers $A[1 \dots n]$. A “significant inversion” in this array is a pair $\langle i, j \rangle$ where $i < j$ and $A[i] > 2A[j]$. For example, in the following array

A	3	9	5	2
	1	2	3	4

there are 2 significant inversions: $\langle 2, 4 \rangle$ and $\langle 3, 4 \rangle$.

- a) Give a Divide-and-Conquer algorithm for counting the number of significant inversion. Your algorithm should run in time $\mathcal{O}(n \log(n))$.
- b) Verify that the running time of your algorithm is $\mathcal{O}(n \log(n))$.

Question 3 The Making Change Problem is the problem of, given a set of coin denominations (such as $\{1, 5, 10, 25, 100, 200\}$ for the Canadian currency), and a value n cents, finding the smallest number of coins needed that add up to n . For example, one way of producing 395 cents in Canadian coins is

1 tooney, 1 looney, 2 quarters, 4 dimes, 1 nickel

which requires 9 coins. A better way is

1 tooney, 1 looney, 3 quarters, 2 dimes

which requires only 7 coins.

In this question you are asked to design a greedy algorithm which, although may not work for all currencies, must work for the Canadian currency. Formally, the denominations are given in an array:

$$C[1], \dots, C[k]$$

(So for the Canadian currency, the input might be

$$C[1] = 25, C[2] = 10, C[3] = 1, C[4] = 200, C[5] = 100, C[6] = 5$$

Here $k = 6$.)

- a) Give a greedy algorithm which, given input the array C of denominations and a value n , output the number of coins of each type needed to add up to n .
- b) Show that your algorithm produces an optimal solution for the Canadian currency.
- c) What is the running time of your algorithm, as a function of k and *the length of the binary representation of n* ?

Question 4 Show that an array of n natural numbers, each is in the range $\{0, 1, 2, \dots, n^3 - 1\}$, can be sorted in time $\mathcal{O}(n)$. (Hint: A number in $\{0, 1, \dots, n^3 - 1\}$ can be expressed in the form

$$an^2 + bn + c$$

where $0 \leq a, b, c \leq n - 1$.)