Theory Assignment 2

CS 452/652/752 Advanced Algorithms and Applications

Full points: 100 + Extra points: 20

Question 1 - 10 points

Suppose `merge sort` routine is applied on the following input arrays: 1, 0, 2, 6, 4, 5, 3, 8, 9, 7. Fast forward to the moment after the two outermost recursive calls complete, but before the final Merge step. Thinking of the two 5-element output arrays of the recursive calls as a glued-together 10-element array, which number is at the 7-th position?

Question 2 - 10 points

Consider the following modification to the `merge sort` algorithm: divide the input array into thirds (rather than halves), recursively sort each third, and finally combine the results using a three-way Merge subroutine. What is the running time of this algorithm as a function of the length n of the input array, ignoring the constant factors and lowest order terms ? (provide explanation and working details)

Question 3 - 15 points

Suppose you are given k sorted arrays, each with n elements, and you want to combine them into a single array of kn elements. Our approach is to use the linear time `merge` subroutine [O(n) runtime] repeatedly, first merging the first two arrays, then merging the result with the third array, then with the fourth array, and so on until you merge in the k-th and the final input array. What is the running time taken by this successive merging algorithm, as a function of k and n, ignoring constant factors and lower-order terms. (provide explanation and working details)

Question 4 - 15 points

Consider again the problem if merging k sorted length-n arrays into a single sorted length-kn array. Consider the algorithm that first divides the k arrays into $k/2$ pairs of arrays, and use the merge subroutine to combine each pair, resulting in $k/2$ sorted length-$2n$ arrays, The algorithm repeats this step until there is only one length-kn sorted array. What is the running time taken by this successive merging algorithm, as a function of k and n, ignoring constant factors and lower-order terms. (provide explanation and working details)

Question 5 - 10 points

Insertion sort can be expressed as a recursive procedure as follows. In order to sort A[1...n], we recursively sort A[1...n-1] and then insert A[n] into the sorted array A[1.. n-1]. Write a recurrence (T(n) as a function of input size n) for the running time of this recursive version of insertion sort.

Question 6 - 20 points

Given an array A = [23, -10, 14, -20, 16, -8, 12, -2] and we want to find the sum of maximum subarray (1) Explain how the divide and conquer algorithm solve this problem. Show level by level results and final result.

(2) Explain how the dynamic programming algorithm solve this problem. Show how you fill the memorization table and get the final result.

Question 7 - 10 points

Given an array A = [4, 0, 6, 7, 2, 5, 9, 3] and want to use `quick sort` to sort the array. Assume you always pick the last element at partition pivot. Show results after each step and final output.

Question 8 - 20 points

In the algorithm D-SELECT, the input elements are divided into groups of 5. Will the algorithm work in linear time if they are divided into groups of 7? Explain why it works or does not work in linear time.

Question 9 - 10 points

Given the market price of rod of different lengths

Length (inch)	1	2	3	4	5	6	7	8	9	10
Price ($)	1	2	4	5	8	9	15	16	19	20

Table 1: Rod price table

Explain how to use dynamic programming to solve the rod cutting problem that maximize the revenue when given a rod of length 10. Show how you fill the memorization table and get the final result.