CSCI 433 Midterm Exam 2

5 hours

March 22, 2021

Problem 1 (10 pts)

Use Master Theorem to solve the following recurrence equations.

1. $T(n) = 2T(\frac{n}{2}) + 1$, $T(1) = 1$
2. $T(n) = 2T(\frac{n}{2}) + n$, $T(1) = 1$
3. $T(n) = 7T(\frac{n}{2}) + n^2$, $T(1) = 1$

Problem 2 (10 pts)

Given an integer array of size n, find the maximum value of the sum of elements of a subarray where no two elements of the subarray are adjacent to each other. Design a $\Theta(n)$ algorithm using a decrease and conquer approach. Your algorithm just needs to find the maximum value of the sum. It does not need to identify the elements of the subarray.

Problem 3 (10 pts)

Suppose we want to find the minimum and maximum value in an array of size n. A brute force approach scans the array twice: in the first pass, it finds the minimum using $n-1$ comparisons; in the second pass, it finds the maximum using $n-1$ comparisons. It total, it takes $2n-2$ comparisons. Now, consider a divide and conquer approach. Assume that the size n is a power of 2. Split the array into two subarrays of equal size. Find the minimum and maximum of subarrays. Two more comparisons are suffice to find the maximum and minimum of the array.

- Write a pseudo code for the above divide and conquer algorithm. Let $T(n)$ be the number of comparisons. Derive the recurrence equations for the above algorithm and show that

  $$T(n) = \frac{3n}{2} - 2$$

  for n a power of 2. (Hints: Prove it by mathematical induction. Pay attention to the base case $T(2)$.)

- (2 Bonus Points) Describe a bottom-up implementation of the above divide and conquer algorithm (i.e., without recursion) that takes $T(n) = \frac{3n}{2} - 2$ comparisons.