University of Zurich[UZH]

Department of Informatics
Database Technology Group
Prof. Dr. M. Böhlen

Informatics II
Exercise 4

March 09, 2020

Divide and Conquer

Task 1. Given an array $A[0...n-1]$ of integers, every element has an even number of occurrences except one element with odd number of occurrences which we call odd element. An element with even occurrences appears in pairs in the array. Find an odd occuring element with an asymptotic complexity of $O(\log n)$ using divide and conquer approach.

a) Draw a tree to illustrate the process of finding the odd occuring element in the array A = [2, 2, 1, 1, 3, 3, 2, 2, 4, 4, 3, 1, 1], according to your divide and conquer algorithm.

b) Provide a C code for divide and conquer odd occuring element finder algorithm.

Task 2. The maximum-subarray algorithm finds the contiguous subarray that has the largest sum within an **unsorted** array `A[0...n-1]` of integers. For example, for array `A = [-2, -5, 6, -2, -3, 1, 5]`, the maximum subarray is `[6, -2, -3, 1, 5]`.
The algorithm works as follows:
Firstly, it divides the input array into two equal partitions: **I** (A[0]...A[*mid*]) and **II** (A[*mid*+1]...A[n-1]). Afterwards, it calls itself recursively on both partitions to find the maximum subarray of each partition. The combination step decides the maximum-subarray by comparing three arrays: the maximum-subarray from the left part, the maximum-subarray from the right part, and the maximum-subarray that overlaps the middle. The maximum-subarray that overlaps the middle is determined by considering all elements to the left and all elements to the right of the middle.

a) Based on the above algorithm description, draw a tree that illustrates the process of determining the maximum subarray in array `A = [-1, 2, -3, 4, 3, -5, 1, 5]`.

b) Provide a C code for maximum-subarray algorithm.

c) Calculate its asymptotic tight bound.

Recurrences

Task 3. Consider the following recurrence:

$$T(n) = \begin{cases} 1 & \text{, if } n = 1 \\ T(n/2) + T(n/4) + T(n/8) + n & \text{, if } n > 1 \end{cases}$$

a) Draw a recursion tree and use it to estimate the asymptotic upper bound of $T(n)$. Demonstrate the tree-based computations that led to your estimate.

b) Use the substitution method to prove that your estimate in (a) is correct.

Task 4. Calculate the asymptotic tight bound of the following recurrences. If the Master Theorem can be used, write down a, b, $f(n)$ and the case (1-3).

1. $T(n) = 3T(\frac{n}{9}) + 32\sqrt{n}$
2. $T(n) = T(\sqrt[3]{n}) + \log n$
3. $T(n) = 15T(n/4) + n^2$
4. $T(n) = 2T(n-2) + n$
5. $T(n) = \log n + T(\sqrt{n})$