College of Computer & Information Science | CS5800
Northeastern University | MS Algorithms
Fall 2017 | 22 September 2017

Problem set 2 (Due Wednesday, October 4, 5:59 pm)

- The assignment is due at the time and date specified. Late assignments will not be accepted.
- We encourage you to attempt and work out all of the problems on your own. You are permitted to study with friends and discuss the problems; however, *you must write up your own solutions, in your own words.*
- If you do collaborate with any of the other students on any problem, please do list all your collaborators in your submission for each problem.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class (or the class staff) is strictly prohibited.
- We require that all homework submissions be neat, organized, and *typeset*. You may use plain text or a word processor like Microsoft Word or LaTeX for your submissions. If you need to draw any diagrams, however, you may draw them with your hand.

1. ($3 \times 2 = 6$ points) Recurrences

Solve the following recurrences and obtain tight asymptotic bounds on $T(n)$. You may ignore the floor operation while solving these recurrences.

(a) $T(n) = 9T(\lfloor n/3 \rfloor) + 7n$, $T(1) = 1$.

(b) $T(n) = 9T(\lfloor n/3 \rfloor) + 5n^2$, $T(1) = 1$.

(c) $T(n) = T(\lfloor n/3 \rfloor) + T(\lfloor 2n/3 \rfloor) + 6n$, $T(1) = 1$.

2. ($2 + 3 = 5$ points) More on recurrences

Let $T(n)$ be defined by the following recurrence:

$$T(n) = \begin{cases} 1, & \text{n=1.} \\ 3T\left(\lfloor \frac{n}{2} \rfloor\right) + n^2, & n > 1; \end{cases}$$

(a) Show that $T(n)$ is a monotonically nondecreasing function.

(b) Find a simple function $f(n)$ such that $T(n) = \Theta(f(n))$. Prove your claim without using the Master theorem. For full credit, prove your claim without ignoring the floor operation. Verify that your result agrees with the Master theorem.

3. ($2 + 2 = 4$ points) Computing a sorted prefix of a list

(a) Prove or disprove: There exists a comparison algorithm that gets as input an array A of length n, and returns a sorted array with the $\lfloor \frac{n}{4} \rfloor$ smallest elements of A, using $O(n)$ comparison operations.

(b) Prove or disprove: There exists a comparison algorithm that gets as input an array A of length n, and returns a sorted array with the $\lfloor\sqrt{n}\rfloor$ smallest elements of A, using $O(n)$ comparison operations.

4. (5 points) Selection from two sorted lists

Describe a comparison algorithm with time complexity $\Theta(\log n)$, that selects an element of a given rank k in the union of two given sorted arrays.

The input for the algorithm consists of an integer k, $1 \leq k \leq 2n$ and two sorted arrays of length n:

$$a = (a_1, a_2, ..., a_n) \text{ such that } a_1 < a_2 < ... < a_n$$

$$b = (b_1, b_2, ..., b_n) \text{ such that } b_1 < b_2 < ... < b_n$$

The output of the algorithm is the element of rank k in the union of the sets of elements of the arrays - $a \cup b$. You may assume that no two elements are equal.

5. (5 points) Hidden surface removal in computer graphics

Chapter 5, Exercise 5, page 248.

6. (1 + 1 + 3 = 5 points) A fault-tolerant AND-gate

Assume we are given an infinite supply of two-input, one-output gates, most of which are AND gates and some of which are OR gates. Unfortunately the OR and AND gates have been mixed together and we can't tell them apart. For a given integer $k \geq 0$, we would like to construct a two-input, one-output combinational "k-AND" circuit from our supply of two-input, one output gates such that the following property holds: If at most k of the gates are OR gates then the circuit correctly implements AND. Assume for simplicity that k is a power of two.

For a given integer $k \geq 0$, we would like to design a k-AND circuit that uses the smallest number of gates.

(a) Design a 1-AND circuit with the smallest number of gates. Argue the correctness of your circuit.

(b) Using a 1-AND circuit as a black box, design a 2-AND circuit. Argue the correctness of your circuit.

(c) Generalizing the above approach, or using a different approach, design the best possible k-AND circuit you can and derive a Θ-bound (in terms of the parameter k) for the number of gates in your k-AND circuit. Argue the correctness of your circuit.