CMPS 4610 Introduction to Algorithms – Fall 16

9/22/16

3. Homework

Due **10/4/16** at the beginning of class

1. **Recurrences (6 points)**
   For each recurrence below, find an asymptotic solution for it using the Master theorem if possible. If the Master theorem does not apply, generate a good guess using the recursion tree method for example (no induction required). Assume that $T(n)$ is constant for sufficiently small n. Justify your answers.

   (a) $T(n) = 125T(\frac{n}{5}) + 1$

   (b) $T(n) = 9T(\frac{n}{3}) + n^2 \log n$

   (c) $T(n) = T(\sqrt{n}) + 1$

2. **Quicksort (4 points)**
   Consider the following types of input of n distinct numbers: (1) Sorted input, (2) random input.

   Determine the runtime of quicksort with the following pivot choices, for both input types:

   (a) The pivot is chosen as the first element.

   (b) The pivot is chosen as a random element.

3. **Quicksort with duplicate keys (8 points)**
   This question is concerned with quicksort on arrays that contain duplicate keys.

   (a) (4 points) How does deterministic quicksort behave on an array with n equal keys? What is its runtime? What is the behavior and the runtime of randomized quicksort in this case? Justify your answer.

   (b) (2 points) If you change $A[j] \leq x$ to $A[j] < x$ in the pseudocode for partition, how does quicksort behave on an array with n equal keys? What is its runtime?

   (c) (2 points) How does deterministic quicksort behave on an array with just two distinct keys (the total number of keys is still n)?

Flip over to back page $\Longrightarrow$

4. **Matrix search (5 points)**
   Let A be an $n \times n$ matrix of integers that is sorted in the following sense: Each row is sorted in non-decreasing order and each column is sorted in non-decreasing order. The task is, for a given integer x to decide whether A contains x or not.

   (a) (1 point) Since each row is sorted, one approach is to perform binary search in each row. What is the (worst-case) running time of this algorithm to search for x in A?

   (b) (4 points) Develop a more efficient divide-and-conquer algorithm for searching x in the sorted matrix A. You can describe your algorithm in pseudo-code or in words. Analyze your runtime.