CSOR W4231: Midterm 1 Practice Problems
Fall 2019

These problems are ungraded, and are intended as a study aid. Solutions will also be posted on canvas. They are of similar flavor to the problems that will appear on the midterm. The midterm is closed book, closed notes; you are allowed to have 1 page of notes (double-sided). It will cover up material up to and including lecture 8. There are 6 problems.

Problem 1 Solve the following recurrences. Give your final answer in Θ notation. You may use any method you like and show your work.

- $T(n) = 2T(n/2) + 5$
- $T(n) = 2T(n-1)$
- $T(n) = 2T(n/3) + \log(n)$

Problem 2 Describe how to modify the QuickSort algorithm to achieve $O(n \log n)$ runtime *deterministically* in the worst-case. You may assume all elements are distinct. Briefly justify this running time. You may use the Partition subroutine and its running time bound as described in class.

Problem 3 Suppose you are given n integers between 0 and n^d. Give the running times for sorting them using each of the following algorithms (as functions of n and d in asymptotic notation). You do not need to justify your answers.

- Mergesort
- Counting sort
- Radix sorting using expressions of each number as d digits in base n

Problem 4 Consider a uniformly random permutation of 1 through n. Compute the expected number of pairs i, j $(i < j)$ such that i still appears before j in the permutation. For example, if we permute 1 through 4 as 3214, the pairs $(1, 4)$, $(2, 4)$, and $(3, 4)$ are the pairs that satisfy this property. Justify your answer.

Problem 5 Prove that, in order to find the *median* of an unsorted array A, one needs to perform at least $n - 1$ comparisons (for any comparisons-based algorithm).[1]

[1] For fun, you can try to prove a higher lower bound, $\geq n$.

Problem 6 Consider a *min*-heap that stores $n \geq 100$ distinct integers. What are the nodes of the min-heap that may contain the 3^{rd} smallest integer (i.e., the rank 4 element in increasing order of the elements in the min-heap)? Deduce an exact number of such possible nodes. How much time would it take to find this rank-4 integer ($O(\cdot)$ answer is ok)? No need to justify your answer.