CS 161: Homework 2

Due by October 14, 2016 at noon

Instructions: Please answer the following questions to the best of your ability. Provide full and rigorous proofs and include all relevant calculations. When writing proofs, please strive for clarity and brevity (in that order). Please see the course website for submission instructions and collaboration policy. Cite any sources that you use.

Question 1 (25 points)

There are n circles of various sizes plotted in the Cartesian plane. For all $i \in \{1, \ldots, n\}$, circle i has a radius $r_i > 0$ and center coordinates (x_i, y_i). The circles may overlap.

Consider a horizontal line of the form $y = a$ or a vertical line of the form $x = b$. These lines can intersect any number of the n circles.

Let a "best" line be a horizontal or vertical line which intersects as many circles as possible. There can be multiple best lines. Your task is to design an algorithm that outputs the number of circles intersected by a best line.

Note that even if a line intersects a circle twice, this counts as only one circle towards the number of intersected circles. Assume that looking up each circle's radius or coordinates by its index takes $O(1)$ time.

(a) (6 points) Design an algorithm to output the number of circles intersected by a best line with worst case running time $\Theta(n^2)$. Prove its correctness and running time bound.

(b) (13 points) Design an algorithm to output the number of circles intersected by a best line with worst case running time $O(n \log n)$. Prove its correctness and running time bound.

(c) (6 points) Now, in addition to considering lines with slope 0 or ∞, also consider lines with slope 1 (i.e., lines of the form $y = x + b$). Modify your algorithm from part (b) to account for this new case while keeping the running time $O(n \log n)$. Prove its correctness and running time bound.

Note: In this question, you may ignore any precision issues when using real numbers.

Question 2 (25 points)

Suppose you have an n-element array A of intervals (x_i, y_i), where x_i, y_i are integers such that $x_i \leq y_i$. The interval (x_i, y_i) represents the set of integers between x_i and y_i. For example, the interval (3,6) represents the set {3, 4, 5, 6}. Define the *overlap* of two intervals I, I' to be $|I \cap I'|$, i.e., the number of integers that are members of both intervals. In this problem, we will design a divide and conquer algorithm that returns the highest overlap among all pairs of intervals in the array. Assume the input array is sorted by the x_i values of the intervals.

(a) (5 points) A naive approach would be to compute the overlap between every pair of intervals. What would be the exact number of comparisons we would need? What would be the runtime of this approach?

(b) (5 points) You are not satisfied with the running time of the last algorithm and realize you can solve this problem faster using a divide-and-conquer approach. Suppose you divide A into two parts L and R

of length $n/2$, such that all the x_i values of intervals in L are smaller than the x_i values of intervals in R. Then you determine the max overlap in these two sub-arrays recursively. Why is this not enough to give the correct answer?

(c) (10 points) Taking advantage of the fact that the array A is sorted, design a divide-and-conquer algorithm that would be faster than the naive approach.

(d) (5 points) What is the recurrence for the running time of your divide-and-conquer algorithm in part (c)? What is the running time of the algorithm?

Question 3 (25 points)

The algorithm Select finds the k^{th} smallest element of an array of n elements in $O(n)$ time. The algorithm *mysteriously* divides n numbers into groups of 5 elements and proceeds. In this problem, we analyze Select with different group sizes other than 5. In parts (a)-(c), we consider groups of size 7, and in part (d), we consider groups of size 3. You may assume that the n elements in the given array are distinct. You may also ignore all the floor and ceiling functions whenever they pop up.

(a) (5 points) Assuming we divide n elements into groups of 7, derive an upper bound on the number of elements greater than the median of medians. Do the same for the number of elements smaller than the median of medians.

(b) (5 points) Derive a recurrence for the worst-case running time $T(n)$ of this variant of Select.

(c) (5 points) Using the substitution method, solve the recurrence from part (b).

(d) (10 points) Assuming we divide the n elements into groups of 3, repeat parts (a)-(c).

Question 4 (25 points)

Solve the recurrences below giving tight upper bounds of the form $T(n) = O(f(n))$ for an appropriate function f. You can use any method from class. Show your work. If you wish, you may assume that n initially has the form $n = a^i$, for an appropriate constant a. Each recurrence is worth 5 points.
Note: Unless stated otherwise, log refers to log base 2, and ln refers to the natural logarithm.

(a) $T(n) = 5T(\frac{n}{4}) + n \log n$

(b) $T(n) = 100T(\sqrt[10]{n}) + 100(\log n)^2$

(c) $T(n) = 7T(\frac{n}{8}) + n \ln n$

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

(e) $T(n) = T(\sqrt[10]{n}) + T(\sqrt[5]{n} + 2) + T(\sqrt[4]{n}) + \log n$

Hint: In some of the recurrences, you can write another recurrence $S(m) = T(f(n))$ for some function f.