CS 161 Winter 2021 Section 3

January 28, 2021

Exercise 1

Suppose you're given n distinct ordered pairs of integers $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$, where for all i, j, $x_i \neq x_j$ and $y_i \neq y_j$. Recall that two points uniquely define a line, $y = mx + b$, with slope m and intercept b. (Note that choosing m and b also uniquely defines a line.) We say that a set of points S is *collinear* if they all fall on the same line; that is, for all $(x_i, y_i) \in S$, $y_i = mx_i + b$ for fixed m and b. In this question, we want to find the maximum cardinality subset of the given points which are collinear. Assume that given two points, you can compute the corresponding m and b for the line passing through them in constant time, and you can compare two slopes or two intercepts in constant time.

a. Design an algorithm to find a maximum cardinality set of collinear points in $O(n^2 \log n)$ time. If there are several maximal sets, your algorithm can output any such set. Since we haven't covered hashing yet, your algorithm should not use any form of hash table.

b. It is not known whether we can solve the collinear points problem in better than $O(n^2)$ time. But suppose we know that our maximum-cardinality set of collinear points consists of exactly n/k points for some constant k. Design a randomzed algorithm that reports the points in some maximum-cardinality set in expected time $O(n)$. *(Hint: Your running time may also be expressed as $O(k^2 n)$.)* Prove the correctness and runtime of your algorithms.

c. Is your algorithm from part b guaranteed to terminate?

Exercise 2

In lecture, we saw the quicksort algorithm, which is an example of a Las Vegas algorithm: a randomized algorithm that is always correct, but that its run time is a random variable. In this question, we consider another family of randomized algorithms: Monte Carlo algorithms.

A Monte Carlo algorithm can return a incorrect answer with some probability. In this question we will see how to amplify the success probability of such algorithms. We assume that the problem we are trying to solve is a *decision problem*, that is, the two possible outputs are Yes or No.

a. Assume that you are given access to an algorithm A that has one-sided error: if the correct answer is No, A always returns No, and if the correct answer is Yes, A returns Yes with probability $1/2$ and No with probability $1/2$. For a given $p < 1/2$, design a randomized algorithm that fails (that is, reports the wrong answer) with probability at most p. What is the run time of your algorithm?

b. Assume that you are given access to an algorithm with two-sided error: given any input, it returns the correct answer with probability $2/3$ and the wrong answer with probability $1/3$. For a given $p < 1/3$, design a randomized algorithm that fails with probability at most p. What is the run time of your algorithm?

Exercise 3

You are given a collection of n differently-sized light bulbs that have to be fit into n flashlights in a dark room. You are guaranteed that there is exactly one appropriately-sized light bulb for each flashlight and vice versa; however, there is no way to compare two bulbs together or two flashlights together as you are in the dark and can barely see! (You are, however, able to see where the flashlights and light bulbs are.) You can try to fit a light bulb into a chosen flashlight, from which you can determine whether the light bulb's base is too large, too small, or is an exact fit for the flashlight. If the bulb fits exactly, it will flash once, in which case you have a correct match. (Note that the flashing light does not allow you to visually compare bulbs/flashlights to other bulbs/flashlights.)

Suggest a (possibly randomized) algorithm to match each light bulb to its matching flashlight. Your algorithm should run strictly faster than quadratic time in expectation. Give an upper bound on the worst-case runtime, then prove your algorithm's correctness and expected runtime.