Advanced Algorithms: Homework 2

Due on Jan. 31, 2024 at 2:29pm EST

Professor Dana Randall Spring 2024

As stated in the syllabus, unauthorized use of previous semester course materials is strictly prohibited in this course.

Exercise 1

(KT Chapter 13, Exercise 1) The 3-coloring problem is a decision problem, but we can word it as an optimization problem as follows. Suppose we are given a graph $G = (V, E)$ and we want to color each node with one of three colors, even if we aren't necessarily able to give different colors to every pair of adjacent nodes. Rather, we say that an edge (u, v) is *satisfied* if the colors assigned to u and v are different.

Consider a 3-coloring that maximizes the number of satisfied edges, and let c^* be the number of edges this satisfies. Give a polynomial-time algorithm that produces a 3-coloring that satisfies at least $\frac{2}{3}c^*$ edges. If you want, your algorithm can be randomized; in this cases the *expected* number of edges it satisfies should be at least $\frac{2}{3}c^*$.

Exercise 2

(KT Chapter 13, Exercise 2) Consider a country in which 100,000 people vote in an election. There are only two candidates on the ballot: a Democratic candidate (denoted D) and a Republican candidate (denoted R). As it happens, this country is heavily Democratic, so 80,000 people go to the polls intending to vote for D and 20,000 go intending to vote for R.

However, the layout of the ballot is a little confusing, so each voter, independently and with probability $\frac{1}{100}$, votes for the wrong candidate — that is, the one that he or she did not intend to vote for.

Let X denote the random variable equal to the number of votes received by the Democratic candidate D, when the voting is conducted with this process of error. Determine the expected value of X, and give an explanation of your derivation of this value.

Exercise 3

(KT Chapter 13, Exercise 12) Consider the following analogue of Karger's algorithm for finding minimum $s-t$ cuts. We will contract edges iteratively using the following randomized procedure. In a given iteration, let s and t denote the possibly contracted nodes that contain the original nodes s and t, respectively. To make sure that s and t do not get contracted, at each iteration we delete any edges connecting s and t and select a random edge to contract among the remaining edges. Give an example to show that the probability that this method finds a minimum $s-t$ cut can be exponentially small.

Exercise 4

(KT Chapter 13, Exercise 6) One of the (many) hard problems that arises in genome mapping can be formulated in the following way. We are given a set of n *markers* $\{\mu_1, \mu_2, \ldots \mu_n\}$ — these are positions on a chromosome that we are trying to map — and our goal is to output a linear ordering of these markers. The output should be consistent with a set of k *constraints*, each specified by a triple $\{\mu_i, \mu_j, \mu_k\}$, requiring that μ_j lie between μ_i and μ_k somewhere in the total ordering that we produce. Note that this constraint does not specify which of μ_i or μ_i should come first in the ordering, only that μ_j should come between them. It also is not saying that they need to be adjacent to each other in this order.

Now it is not always possible to satisfy all constraints simultaneously, so we wish to produce an ordering that satisfies as many as possible. Unfortunately, deciding whether there is an ordering that satisfies at least k' of the k constraints is NP-complete. (You do not have to prove this!)

Give a constant $\alpha > 0$ (independent of n) and an algorithm with the following property. If it is possible to satisfy k^* of the constraints, then the algorithm produces an ordering of markers satisfying at least αk^* of the constraints. Your algorithm may be randomized; in this case it should produce an ordering for which the *expected* number of satisfied constraints is at least αk^*.