Due: 6 pm on Wednesday, June 4

Assignment 13

1 Max 3-Coloring

3-Coloring is a yes/no question, but we can phrase 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 are not 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^* denote this number. Give a polynomial-time randomized algorithm that produces a 3-coloring that satisfies at least $\frac{2}{3}c^*$ edges in expectation.

2 MaxSAT

In class, we designed an approximation algorithm to within a factor of $7/8$ for the Max-3SAT problem, where we assumed that each clause has terms associated with 3 different variables. In this problem we will consider the more general Max-SAT problem: given a set of clauses $C_1, \ldots, C_k$ over a set of variables $X = \{x_1, \ldots, x_n\}$, find a truth assignment satisfying as many of the clauses as possible. Each clause has at least one term in it, but otherwise we do not make any assumptions on the length of the clauses: there may be clauses that have a lot of variables, and others may have just a single variable.

(a) First consider the randomized approximation algorithm we used for Max-3SAT, setting each variable independently to true or false with probability $1/2$ each. Show that the expected number of clauses satisfied by this random assignment is at least $k/2$, i.e., half of the clauses is satisfied in expectation. Give an example to show that there are MaxSAT instances such that no assignment satisfies more than half of the clauses.

(b) If we have a clause that consists just of a single term (e.g. a clause consisting just of x_1, or just of $\overline{x_2}$), then there is only a single way to satisfy it: we need to set the corresponding variable in the appropriate way. If we have two clauses such that one consists of just the term x_i, and the other consists of just the negated term $\overline{x_i}$, then this is a pretty direct contradiction.

Assume that our instance has no such pair of "conflicting clauses", that is, for no variable x_i do we have both a clause $C = \{x_i\}$ and a clause $C' = \{\overline{x_i}\}$. Modify the above randomized procedure to improve the approximation factor from $1/2$ to at least a 0.6 approximation, that is, change the algorithm so that the expected number of clauses satisfied by the process is at least $0.6k$.

(c) Give a randomized polynomial time algorithm for the general MaxSAT problem, so that the expected number of clauses satisfied by the algorithm is at least a 0.6 fraction of the maximum possible.

3 Genome Mapping

One of the (many) hard problems that arises in genome mapping can be formulated in the following abstract way. We are given a set of n markers $\{\mu_1, \ldots, \mu_n\}$ - 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 (μ_i, μ_j, μ_k), requiring that μ_j lie between μ_i and μ_k in the total ordering that we produce (μ_i and μ_k are mutually exchangeable).

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 an NP-complete problem (you don't 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. You should provide a randomized algorithm which has **expected** polynomial running time and **always** produces an approximation within a factor of α.

4 Dense Induced Graph

Let $G = (V, E)$ be an undirected graph with n nodes and m edges. For a subset $X \subseteq V$, we use $G[X]$ to denote the subgraph induced on X, that is, the graph $(X, \{(u, v) \in E : u, v \in X\})$.

We are given a natural number $k \leq n$, and are interested in finding a set of k nodes that induces a "dense" subgraph of G; we'll phrase this concretely as follows. Give a polynomial-time algorithm that produces, for a given natural number $k \leq n$, a set $X \subseteq V$ of k nodes with the property that the induced subgraph $G[X]$ has at least $\frac{mk(k-1)}{n(n-1)}$ edges.

You should give a randomized algorithm that has an **expected** polynomial running time and **always** outputs correct answers.