CSE431: Complexity Theory November 14, 2018

Homework 3

Anup Rao Due: November 27, 2018

Read the fine print[1]. Each problem is worth 10 points:

1. Recall that in the 3-coloring problem we want to color the vertices of a graph with 3-colors such that every edge is colored with two colors. We showed that it is **NP**-hard to give algorithms that can 3-color a graph. Give and (analyze) a randomized algorithm that colors 2/3 fraction of the edges correctly in expectation. HINT: The algorithm is very similar to the randomized algorithm for MAX-Cut that we saw in class.

2. In class we showed that the expected time for the randomized 2SAT algorithm to find a satisfying assignment, if one exists, is $O(n^2)$. Here we study the performance of the same algorithm on 3SAT. In each step, the algorithm picks a uniformly random variable from an unsatisfied clause and flips the value of the variable. The Chernoff bound states that $X_1, \dots, X_n$ are independent random variables such that

$$X_i = \begin{cases} 1 & \text{with probability } p, \\ 0 & \text{with probability } 1-p, \end{cases}$$

then for every ϵ, $\Pr[\sum_{i=1}^{n} X_i > pn(1+\epsilon)] < 2^{-\epsilon^2 pn/4}$.

   (a) Assume that the 3SAT formula has exactly one satisfying assignment, and this assignment has distance at least $n/2$ from the initial assignment that the algorithm starts out with (this is the hard case for the algorithm). Argue that the only way that the algorithm can succeed is if there is some contiguous interval of $n/2$ steps, where in at least $n/4$ of those $n/2$ steps, the algorithm moves towards the satisfying solution. To do this, consider the final $n/2$ steps in a sequence of steps that leads the algorithm towards a satisfying solution.

   (b) Use the Chernoff boud to argue that the probability that such an interval occurs is exponentially small in n. Use the union bound to conclude that the probability that the algorithm finds the solution in time $2^{o(n)}$ is at most $2^{-\Omega(n)}$.

   (c) Finally, argue that the expected time for the algorithm to find a solution is at most $2^{O(n)}$. To do this, break up the run of the algorithm into consecutive blocks of length n and observe that if the algorithm makes n moves towards the satisfying solution in

[1] In solving the problem sets, you are allowed to collaborate with fellow students taking the class, but **each submission can have at most one author**. If you do collaborate in any way, you must acknowledge, for each problem, the people you worked with on that problem. The problems have been carefully chosen for their pedagogical value, and hence might be similar to those given in past offerings of this course at UW, or similar to other courses at other schools. Using any pre-existing solutions from these sources, for from the web, constitutes a violation of the academic integrity you are expected to exemplify, and is strictly prohibited. Most of the problems only require one or two key ideas for their solution. It will help you a lot to spell out these main ideas so that you can get most of the credit for a problem even if you err on the finer details. Please justify all answers. Some other guidelines for writing good solutions are here: `http://www.cs.washington.edu/education/courses/cse421/08wi/guidelines.pdf`.

any one of these blocks, then it will find the satisfying solution. Compute the expected number of blocks that the algorithm has to execute before finding the satisfying solution.

3. Recall the following function:

$$\mathbf{2COL}(G) = \begin{cases} 1, & \text{if graph } G \text{ has a coloring with two colors} \\ 0, & \text{otherwise,} \end{cases}$$

   where a coloring of G with c colors is an assignment of a number in $[c]$ to each vertex such that no adjacent vertices get the same number.

   Prove that $2\mathbf{COL} \in \mathbf{NL}$. You can use the following fact: A graph G can be colored with two colors if and only if it contains no cycle of odd length.

4. Suppose **TQBF** is also **PSPACE**-complete under log-space reductions—meaning that for every $f \in \mathbf{PSPACE}$, there is a logspace computable function h such that $f(x) = \mathbf{TQBF}(h(x))$. Prove that this implies that $\mathbf{TQBF} \notin \mathbf{NL}$. [Hint: Use Savitch's theorem and one of the hierarchy theorems.]