TTIC 31010 and CMSC 37000 Algorithms

Name: ______________________________

Sample Exam

- The exam contains 4 questions. You need to solve all of them to get full credit.
- You can use any results that were proved in class (no need to re-prove them), as long as you **state them precisely**.
- Make sure that your proofs are formal and complete.

Question 1 (25%) Suppose we are given a connected directed graph $G = (V, E)$, with positive integral capacities $c(e)$ on each edge e, and a pair of vertices s and t. We are also given a maximum s-t flow in G, defined by a flow value $f(e)$ on each edge e, where all values $f(e)$ are integral. The flow f is also acyclic, that is, there is no cycle in G on which all edges carry positive flow.

a. Suppose we pick some edge $e^* \in E$ and increase its capacity by 1 unit. Show how to find a maximum flow in the resulting flow network in time $O(m)$, where $m = |E|$.

b. Suppose we pick some edge $e^* \in E$ with non-zero capacity and reduce its capacity by 1 unit. Show how to find a maximum flow in the resulting flow network in time $O(m)$.

Question 2 (25%) In the Minimum Dominating Set problem, we are given an undirected graph $G = (V, E)$ with non-negative weights $w(v)$ on vertices. We say that a subset $S \subseteq V$ of vertices is a *dominating set* iff for every vertex $u \notin S$, there is some edge $(u, v) \in E$, such that $v \in S$. The goal in the Minimum Dominating Set Problem is to find a dominating set S, minimizing $\sum_{v \in S} w(v)$.

a. Give an efficient algorithm for solving Minimum Dominating Set on trees. Prove the algorithm's correctness.

b. Prove that Minimum Dominating Set is NP-complete in general graphs.

Question 3 (25%) Suppose you are given an $n \times n$ grid graph, as in the figure below. Associated with each node v of the grid is a non-negative integer weight $w(v)$. You may assume that the weights of all vertices are distinct. Your goal is to choose an independent set S of vertices of the grid, so that the sum of the total weight of the vertices in S, $\sum_{v \in S} w(v)$ is maximized.

Consider the following greedy algorithm.

- Start with $S = \emptyset$.
- While some node remains in G:

  a. Pick a node $v \in G$ of maximum weight.

b. Add v to S.

c. Delete v and all its neighbors, together with their adjacent edges, from G.

- Return S.

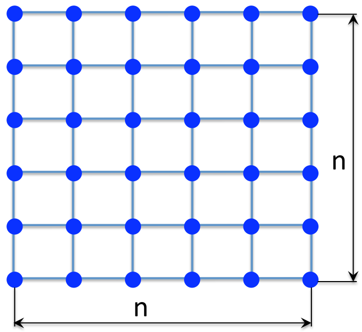

Figure 1: An $n \times n$ grid graph

a. Let S be the solution returned by the above algorithm, and let T be any other independent set in G. Show that for every node $v \in T$, either $v \in S$, or there is a node $v' \in S$, so that $w(v) \leq w(v')$, and v' is a neighbor of v.

b. Show that the above greedy algorithm returns an independent set of weight at least $\mathsf{OPT}/4$, where OPT is the weight of the maximum-weight independent set.

c. Show an example where the weight of the solution produced by the algorithm is at most $\frac{\mathsf{OPT}}{4} + \epsilon$, where $\epsilon = 0.001$. (You are free to choose the value n that works best for your example).

Question 4 (25%) In the k-Not-All-Equal problem, we are given a set $x_1, \ldots, x_n$ of variables that can be assigned values 0 or 1. Additionally, we are given a collection Σ of m constraints. Each constraint $C_i \in \Sigma$ is specified by a subset $x_{i_1}, x_{i_2}, \ldots x_{i_k}$ of k variables. Constraint C_i is satisfied iff not all variables are assigned the same value. In other words, the only assignments that **do not** satisfy C_i are the ones where all variables $x_{i_1}, x_{i_2}, \ldots x_{i_k}$ are assigned 0, or all these variables are assigned 1. The goal is to find an assignment that satisfies as many constraints as possible.

a. Consider an algorithm that chooses, for every variable x_i, an assignment 0 or 1 independently at random, with probability $\frac{1}{2}$ each. What is the expected number of constraints satisfied by the solution the algorithm produces?

b. Assume now that the variables are allowed to take values in set $\{1, \ldots, r\}$. Extend the above randomized algorithm to this case. What is the expected number of constraints satisfied by the solution produced by the algorithm?

c. Prove that any instance of k-Not-All-Equal problem on $m = 5$ constraints, where $k = 4$ and $r = 3$, always has a solution satisfying all constraints.