CSCI-GA 3520: Honors Analysis of Algorithms

Final Exam: Fri, Dec 19 2014, Room WWH- 312, 11:00-3:00pm.

• This is a four hour exam. There are six questions, worth 10 points each. Answer all questions and all their subparts.

• This is a closed book exam. No books, notes, reference material, either hard-copy, soft-copy or online, is allowed.

• Please print your name and SID on the front of the envelope only (not on the exam booklets). Please answer each question in a separate booklet, and number each booklet according to the question.

• Read the questions carefully. Keep your answers legible, and brief but precise. Assume standard results.

• You must prove correctness of your algorithm and prove its time bound unless stated otherwise. The algorithm can be written in plain English (preferred) or as a pseudo-code.

• The graphs are undirected in Problems 2, 5 and directed in Problems 1, 4.

Best of luck!
Problem 1 (Graphs are directed)

Suppose a CS curriculum consists of \( n \) courses, all of them mandatory. The pre-requisite graph \( G(V, E) \), \( |V| = n \) has a vertex for each course, and a directed edge from course \( v \) to course \( w \) if and only if \( v \) is a pre-requisite for \( w \). Give an algorithm that computes the minimum number of semesters necessary to complete the curriculum. You may assume that a student can take any number of courses in one semester. The running time of your algorithm should be \( O(|V| + |E|) \).

Assume adjacency list representation of the graph. Assume that \( G(V, E) \) does not have a directed cycle.

Problem 2 (Graphs are undirected)

This problem requires the creation of a data structure that maintains connected components of a graph on vertex set \( \{1, \ldots, n\} \). The graph initially has no edges and then \( m \) edges are added to it one at a time, at time steps \( t = 1, 2, \ldots, m \). The edge added at step \( t \) is \((x_t, y_t)\) where \( x_t, y_t \in \{1, \ldots, n\} \) and \( x_t \neq y_t \). Let \( G_t \) be the graph after adding the first \( t \) edges. Let \( C_t \) denote the size of the largest component of \( G_t \).

1. Give an efficient algorithm that determines for each \( t \) whether or not \( x_t, y_t \) lie in the same connected component of \( G_{t-1} \). The additional time, given the data structure at the end of step \( t - 1 \), should be \( O(\log n) \).

2. Extend your algorithm above to determine \( C_t \) in \( O(1) \) additional time.

Problem 3

You are given a string of \( n \) characters \( s[1, \ldots, n] \), which you believe to be a corrupted text document in which all punctuation has vanished (so that it looks like “itwasthebestoftimes”). You wish to reconstruct the document using a dictionary, which is available in the form of a Boolean function \( \text{Dict}(\cdot) \): for any string \( w \), \( \text{Dict}(w) \) is true if \( w \) is a valid word, and false otherwise.

Give an algorithm that determines whether the string \( s[1, \ldots, n] \) can be reconstituted as a sequence of valid words. The running time should be at most \( O(n^2) \), assuming calls to \( \text{Dict}(\cdot) \) take unit time.
Problem 4 (Graphs are directed)

You are given a directed graph $G(V, E)$ with a non-negative weight function $\text{wt} : E \rightarrow \mathbb{R}^+$ on its edges. The graph is presented in the adjacency list representation. A source vertex $s \in V$ is specified. The goal is to design an algorithm that finds the smallest weight of a path from $s$ to every vertex in the graph.

1. One can of course use Dijkstra’s algorithm. What is the data structure used and the running time (in terms of $|V|$ and $|E|$) for an efficient implementation of Dijkstra’s algorithm?
   Just state the answer. No explanation necessary.

2. If all edges have unit weight, do you know a faster algorithm?
   Just state the answer. No explanation necessary.

3. Now suppose the weight function has the form $\text{wt} : E \rightarrow \{1, 2\}$, i.e. every edge has weight either 1 or 2. Design $O(|V| + |E|)$-time algorithm for the problem.

Problem 5 (Graphs are undirected)

Construct a graph $G(V, E)$ at random as follows. Let $V$ be a set of $n$ vertices. For each pair of distinct vertices $u, v \in V$, let $(u, v) \in E$ with probability $p$, independently for all vertex pairs. That is, for each vertex pair $(u, v)$, the pair is included as an edge with probability $p$ and left out with probability $1 − p$, independently for all vertex pairs. We intend to analyze the probability that the resulting graph happens to be connected.

For a subset $S \subseteq V$, $1 \leq |S| \leq \frac{n}{2}$, let $E_S$ be the event that there is no edge in $G$ between the sets $S$ and $V \setminus S$.

1. If $|S| = j$, what is $\Pr[E_S]$?

2. Let $\mathcal{D}$ be the event that $G$ is disconnected. Can you express $\mathcal{D}$ in terms of events $E_S$?

3. Can you provide an upper bound on $\Pr[\mathcal{D}]$ as a function $U(p)$ of parameter $p$? Hint: Union bound.

4. What is the smallest value of $p$ for which $U(p) \leq \frac{1}{1000}$?
   The desired value of $p$ should be in terms of number of vertices $n$. Give the smallest value you can. It is enough to be correct up to a constant factor. A rough calculation suffices and the proof need not be completely formal.
Problem 6

Let \( H(V, E) \) be a 3-uniform hyper-graph, i.e. \( V \) is a set of vertices and each hyper-edge \( e \in E \) is a 3-element subset of \( V \). A vertex cover in a hyper-graph is a subset \( S \subseteq V \) such that \( e \cap S \neq \emptyset \) for every \( e \in E \). Define the language HYPERGRAPH VERTEX COVER as follows:

\[
\text{HYPERGRAPH VERTEX COVER} = \{(H(V, E), k) \mid H(V, E) \text{ is a 3-uniform hyper-graph and } \exists S \subseteq V, |S| \leq k, \text{ such that } S \text{ is a vertex cover in } H(V, E)\}.
\]

1. Show that HYPERGRAPH VERTEX COVER is in NP.

2. Show that HYPERGRAPH VERTEX COVER is NP-complete.

   \textit{Hint: reduce from a closely related problem on graphs.}