ECE 374 B - Fall 2023

Homework 7

- **Submit your solutions electronically on the course Gradescope site as PDF files.** If you plan to typeset your solutions, please use the LaTeX solution template on the course web site. If you must submit scanned handwritten solutions, please use a black pen on blank white paper and a high-quality scanner app (or an actual scanner, not just a phone camera). We will mark difficult to read solutions as incorrect and move on.
- **Every homework problem must be done *individually*.** Each problem needs to be submitted to Gradescope before 6AM of the due data which can be found on the course website: https://ecealgo.com/homeworks.html.
- For nearly every problem, **we have covered all the requisite knowledge required to complete a homework assignment prior to the "assigned" date.** This means that there is no reason not to begin a homework assignment as soon as it is assigned. Starting a problem the night before it is due a recipe for failure.

Policies to keep in mind

- **You may use any source at your disposal**—paper, electronic, or human—but you ***must*** cite ***every*** source that you use, and you ***must*** write everything yourself in your own words. See the academic integrity policies on the course web site for more details.
- **Being able to clearly and concisely explain your solution is a part of the grade you will receive.** Before submitting a solution ask yourself, if you were reading the solution without having seen it before, would you be able to understand it within two minutes? If not, you need to edit. Images and flow-charts are very useful for concisely explain difficult concepts.

See the course web site (https://ecealgo.com) for more information.

If you have any questions about these policies,
please don't hesitate to ask in class, in office hours, or on Piazza.

1. You are given a list $D[n]$ of n words each of length k over an alphabet Σ in a language you don't know, although you are told that words are sorted in lexicographic order. Using $D[n]$, describe an algorithm to efficiently identify the order of the symbols in Σ. For example, given the alphabet $\Sigma = \{Q, X, Z\}$ and the list $D = \{QQZ, QZZ, XQZ, XQX, XXX\}$, your algorithm should return QZX. You may assume D always contains enough information to completely determine the order of the symbols. (Hint: use a graph structure, where each node represents one letter.)

2. Given a directed-acyclic-graph ($G = (V, E)$) with integer (positive or negative) edge weights:

   (a) Give an algorithm to find the **shortest** path from a node s to a node t.

   (b) Give an algorithm to find the **longest** path from a node s to a node t.

3. Suppose you are given two DAGs $G(V_G, E_G)$ and $H(V_H, E_H)$ in which every node has a *label* from some finite alphabet; different nodes may have the same label. The label of a *path* in either dag is the string obtained by concatenating the labels of its vertices.

   (a) Describe and analyze an algorithm to compute the length of a longest string that is both the label of a path in G and the label of a path in H.

   (b) Describe and analyze an algorithm to compute the length of a longest string that is both a subsequence of the label of a path in G and a subsequence of the label of a path in H.

   (c) Describe and analyze an algorithm to compute the length of a shortest string that is both a supersequence of the label of a path in G and a supersequence of the label of a path in H.

4. (a) Define the *width* of a walk in a graph to be the maximum weight of any vertex in the walk. Let $G(V, E)$ be a *directed* graph with weighted edges, $w(v)$ be the weight of each vertex v, s be a starting vertex, t be a target vertex and L be a maximum length. Describe and analyze an algorithm that returns the *maximum width* of any walk from s to t in G whose total length is at most L. You may assume G has no negative cycles but *not* that it has no edges of negative weight.

   (b) Let $G(V, E)$ be a *directed* graph with *nonnegative* edge weights, let s and t be vertices of G and let $H(V, E')$ be a subgraph of G where $E' \neq E$. Suppose we want to reinsert *exactly one* edge from G back into H, so that a shortest path from s to t in the resulting graph is as short as possible. Describe and analyze an algorithm that chooses a best edge to reinsert.

   If for any reinserted edge the shortest-path distance from s to t doesn't change, your algorithm should return NONE. If there are multiple edges that improve the shortest-path distance from s to t the same largest amount, your algorithm should return one of those edges.