CMPSC 465 — Data Structures & Algorithms

Fall 2025 — Ke Chen and Yana Safonova

HW 4 Solution

Due October 06, 10:00 pm

Instructions: You are encouraged to solve the problem sets on your own, or in groups of three to five people, but you must write your solutions strictly by yourself. You must explicitly acknowledge in your write-up all your collaborators, as well as any books, papers, web pages, etc. you got ideas from.

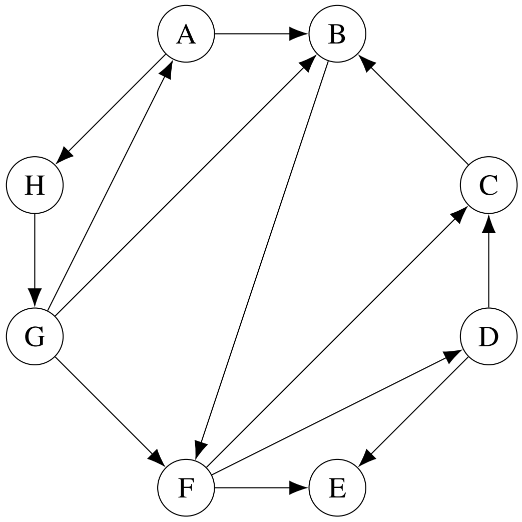
Formatting: Each part of each problem should begin on a new page. Each page should be clearly labeled with the problem number and the problem part. The pages of your homework submissions must be in order. **When submitting in Gradescope, make sure that you assign pages to problems from the rubric. You risk receiving no credit for it if you do not adhere to these guidelines.**

Late homework will not be accepted. Please, do not ask for extensions since we will provide solutions shortly after the due date. Remember that we will drop your lowest three scores.

This homework is due Monday, October 6, at 10:00 pm electronically. You need to submit it via Gradescope. Please ask on Canvas about any details concerning Gradescope.

1. (20 pts.) **Depth-First Search.**

(a). Perform depth-first search on the following graph; whenever there's a choice of vertices, pick the one that is alphabetically first. Classify each edge as a tree edge, forward edge, back edge, or cross edge, and give the pre and post number of each vertex.



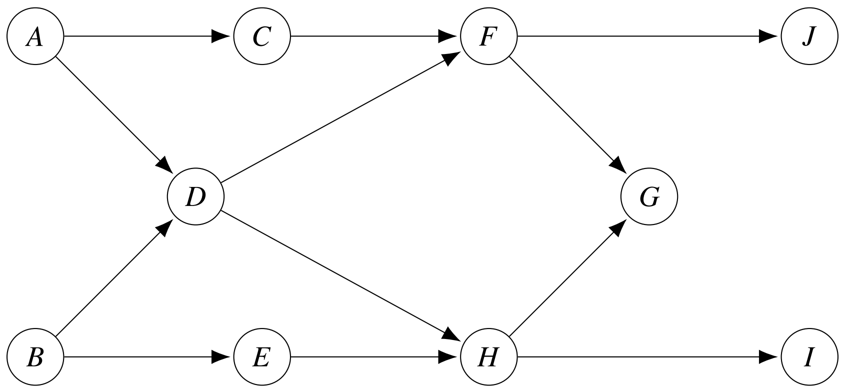
(b). Either prove or give a counterexample: if $\{u,v\}$ is an edge in an undirected graph, and during depth-first search $post(u) < post(v)$, then v is an ancestor of u in the DFS tree.

2. (20 pts.) **Topological Sort** Consider the following directed graph.

(a) What are the sources and sinks of the graph?

(b) Run the topological sort algorithm we learned in class on this graph, use alphabetical order whenever there is a choice

(c) Prove or disprove that the ordering ACBDFJEHIG can be obtained by the algorithm we learned [hint: can you find a valid DFS forest with pre- and post-numbers that yields the given ordering?]



3. (20 pts.) **Ancient Cities.** You are given a set of ancient cities in a desert, along with the pattern of only roads between them, in the form of an undirected graph $G = (V, E)$. Each stretch of the road $e \in E$ connects two of the cities, and you know its length in miles, ℓ_e. You want to get from city s to city t. There's one problem: your bottle can only hold enough water to cover L miles. There are fountains in each city to refill your bottle, but not between cities. Therefore, you can only take a route if every one of its edges has length $\ell_e \leq L$. (**Note:** Only use the algorithms that have been covered in class so far to solve the below problems.)

(a) Given the limitation on your water bottle's capacity, show how to determine in linear time $O(|V| + |E|)$ whether there is a feasible route from s to t. State your algorithm clearly, prove that it is correct and analyze its running time

(b) You are now planning to buy a new bottle, and you want to know the minimum capacity that is needed to travel from s to t. Give an $O((|V| + |E|) \log |V|)$ algorithm to determine this. State your algorithm clearly, prove that it is correct and analyze its running time. (Hint: Consider all the possible values for the minimum bottle capacity. How many are there? What would be the running time of the algorithm that tries them all? Then, try to improve this algorithm.)

4. (20 pts.) **Cycle with an Edge.** Given an undirected graph $G = (V, E)$ and a specific edge $e = \{u, v\} \in E$, the task is to determine if there exists a cycle in G that includes the edge e. Design an algorithm and analyze its runtime.

5. (20 pts.) **Prerequisites.** Suppose a CS curriculum consists of n courses, all of them mandatory. The prerequisite graph G has a node for each course, and an edge from course v to course w if and only if v is a prerequisite for w.

(a) Design an algorithm to check if it is possible to complete all courses.

(b) Design an algorithm to output a valid sequence in which to take all courses (if your algorithm from part (a) determines that this is possible).

(c) Suppose all courses are offered each semester. A student can take any number of courses in a semester, as long as no two are prerequisites of each other. Design an algorithm to find the minimum number of semesters needed to complete the curriculum.

Rubric:

Problem 1, ? pts

?

Problem 2, ? pts

?

Problem 3, ? pts

?

Problem 4, ? pts

?

Problem 5, ? pts

?