CMPSC 465 Data Structures & Algorithms
Fall 2024 Mingfu Shao and Debarati Das

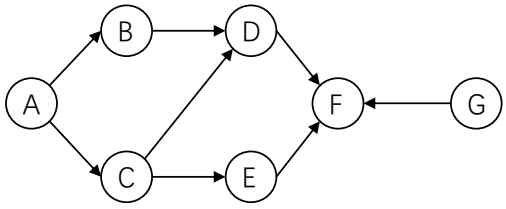
HW 4

Due September 27th (Friday), 11:59 pm

Formatting: Each problem should begin on a new page. When submitting in Gradescope, try to assign pages to problems from the rubric as much as you can. Make sure you write all your group members' names. For the full policy on assignments, consult the syllabus.

0. (0 pts.) Acknowledgements. List any resources besides the course material that you consulted in order to solve the assignment problems. If you did not consult anything, write "I did not consult any non-class materials." The assignment will receive a 0 if this question is not answered.

1. (13 pts.) Consider the following directed graph:



   a. (2 pts) What are the sources and sinks of the graph?
   b. (4 pts) Perform DFS with timing on the graph. When there's a choice of vertices, pick the alphabetically first one. Provide the pre and post numbers for each vertex, and the postlist.
   c. (4 pts) How many linearizations does this graph have?
   d. (3 pts) Draw the meta-graph of this graph and list the vertices in each connected component.

2. (10 + 5 (bonus) pts.) A directed graph is said to have the SC-property if for every pair of vertices (u, v) there is atmost one simple path from u to v.

   a. (10 pts) Let $G = (V, E)$ be a given directed graph. Given $u \in V$, design an algorithm to decide if there is atmost one simple path from u to every other $v \in V \setminus \{u\}$.
   b. (Bonus question, 2 pts) Analyze the time complexity of your algorithm for part (a). You will get full points if you can show your algorithm runs in $O(|V|)$ time.
   c. (Bonus question, 3 pts) Give an $O(|V|^2)$-time algorithm to check if a given directed graph G has the SC-property.

3. (15 pts.) You are given an undirected graph $G = (V, E)$, where V is the set of vertices and E is the set of edges. Your task is to count the total number of simple cycles of length n in the graph. A simple cycle of length n is defined as a cycle that contains exactly n vertices and n edges. Each cycle should only be counted once, regardless of its starting vertex or direction.

Input: an undirected graph $G = (V, E)$, represented as an adjacency list; an integer n, which represents the desired length of the cycle.

Output: return the total number of simple cycles of length n in graph G.

Design an algorithm for above problem and analyze its time complexity. (Hints: consider a DFS-based algorithm; ensure that each cycle is counted only once, despite the undirected nature of the graph; optimize the DFS to avoid redundant searches and prune unnecessary paths that do not lead to valid cycles.)

4. (5 bonus pts.) In Lecture 11 we showed that in order to obtain a special order to run DFS to identify all connected components in a directed graph G, we need to build the reverse graph G_R of G and then run DFS-with-timing on G_R to get the resulting postlist (termed $postlist_R$), which is the special order. You might wonder if running DFS on G to obtain the postlist followed by reversing it would also result in a correct special order. Show that this approach does not work by giving an example. You need to draw a directed graph G (the example), show all connected components of G, show the postlist after running DFS-with-timing on G, show the visited array after runing DFS following the reverse of the postlist, and verify that it does not give the correct connected components.