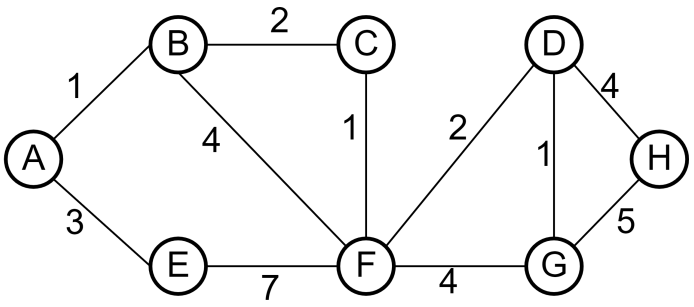
CS 61B Fall 2020

DFS, BFS, Shortest Paths

Exam Prep Discussion 10: October 26, 2020

1 DFS, BFS, Dijkstra's, A*

For the following questions, use the graph below and assume that we break ties by visiting lexicographically earlier nodes first.



(a) Give the depth first search preorder traversal starting from vertex A.

(b) Give the depth first search postorder traversal starting from vertex A.

(c) Give the breadth first search traversal starting from vertex A.

(d) Give the order in which Dijkstra's Algorithm would visit each vertex, starting from vertex A. Sketch the resulting shortest paths tree.

(e) Give the path A* search would return, starting from A and with G as a goal. Let $h(u, v)$ be the valued returned by the heuristic for nodes u and v.

u	v	$h(u, v)$
A	G	9
B	G	7
C	G	4
D	G	1
E	G	10
F	G	3
H	G	5

2 Graph Conceptuals

Answer the following questions as either **True** or **False** and provide a brief explanation:

1. If a graph with n vertices has $n - 1$ edges, it **must** be a tree.

2. The adjacency matrix representation is **typically** better than the adjacency list representation when the graph is very connected.

3. Every edge is looked at exactly twice in **every** iteration of DFS on a connected, undirected graph.

4. In BFS, let $d(v)$ be the minimum number of edges between a vertex v and the start vertex. For any two vertices u, v in the fringe, $|d(u) - d(v)|$ is **always less than** 2.

5. Given a fully connected, directed graph (a directed edge exists between every pair of vertices), a topological sort can never exist.

3 Cycle Detection

Given an undirected graph, provide an algorithm that returns true if a cycle exists in the graph, and false otherwise. Also, provide a Θ bound for the worst case runtime of your algorithm. You may use either an adjacency list or an adjacency matrix to represent your graph. (We are looking for an answer in plain English, not code).