

Study Questions

ICS 163 Graph Algorithms

Not Due (But will be discussed Wednesday, January 29, 2003, in class)

1. Let G be a simple connected graph with n vertices and m edges. Explain why $O(\log m)$ is $O(\log n)$.
2. Draw a simple connected directed graph with 8 vertices and 16 edges such that the in-degree and out-degree of each vertex is 2. Show that there is a single non-simple cycle that includes all the edges of your graph, that is, you can trace all the edges in their respective directions without ever lifting your pencil. (Such a cycle is called a *directed Euler tour*.)
3. Draw the directed graph induced by the courses required for the ICS major. (Where vertices are courses and edges are prerequisite requirements.) Compute a topological ordering on the vertices of this graph.
4. Let G be a graph whose vertices are the integers 1 through 8, and let the adjacent vertices of each vertex be given by the table below:

vertex	adjacent vertices
1	(2, 3, 4)
2	(1, 3, 4)
3	(1, 2, 4)
4	(1, 2, 3, 6)
5	(6, 7, 8)
6	(4, 5, 7)
7	(5, 6, 8)
8	(5, 7)

Assume that, in a traversal of G , the adjacent vertices of a given vertex are returned in the same order as they are listed in the above table.

- (a) Draw G .
 - (b) Order the vertices as they are visited in a DFS traversal starting at vertex 1.
 - (c) Order the vertices as they are visited in a BFS traversal starting at vertex 1.
5. Would you use the adjacency list structure or the adjacency matrix structure in each of the following cases? Justify your choice.
 - (a) The graph has 10,000 vertices and 20,000 edges, and it is important to use as little space as possible.
 - (b) The graph has 10,000 vertices and 20,000,000 edges, and it is important to use as little space as possible.
 - (c) You need to answer the query `areAdjacent(v, w)` as fast as possible, no matter how much space you use.
 6. Let T be the spanning tree rooted at the start vertex produced by the depth-first search of a connected, undirected graph G . Argue why every edge of G goes from a vertex in T to one of its ancestors, that is, it is a *back edge*.

7. Show that if a graph G has at least three vertices, then it has a separation edge only if it has a separation vertex.
8. Show that, if T is a BFS tree produced for a connected graph G , then, for each vertex v at level i , the path of T between s and v has i edges, and any other path of G between s and v has at least i edges.
9. A company named RT&T has a network of n switching stations connected by m high-speed communication links. Each customer's phone is directly connected to one station in his or her area. The engineers of RT&T have developed a prototype video-phone system that allows two customers to see each other during a phone call. In order to have acceptable image quality, however, the number of links used to transmit video signals between the two parties cannot exceed 4. Suppose that RT&T's network is represented by a graph. Design an efficient algorithm that computes, for each station, the set of stations it can reach using no more than 4 links.
10. Explain why there are no forward non-tree edges with respect to a BFS tree constructed for a directed graph.
11. An independent set of an undirected graph $G = (V, E)$ is a subset I of V such that no two vertices in I are adjacent. That is, if $u, v \in I$, then $(u, v) \notin E$. A *maximal independent set* M is an independent set such that, if we were to add any additional vertex to M , then it would not be independent any longer. Every graph has a maximal independent set. (Can you see this? This question is not part of the exercise, but it is worth thinking about.) Give an efficient algorithm that computes a maximal independent set for a graph G . What is this method's running time?