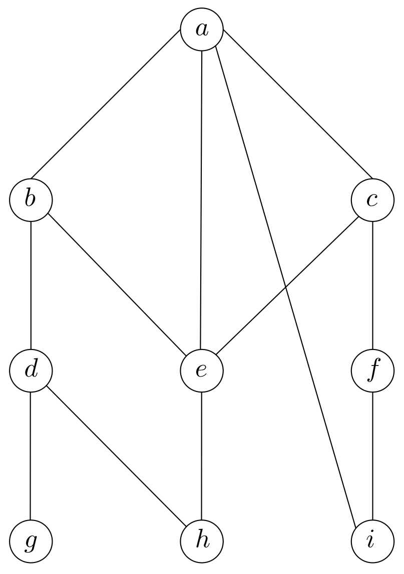
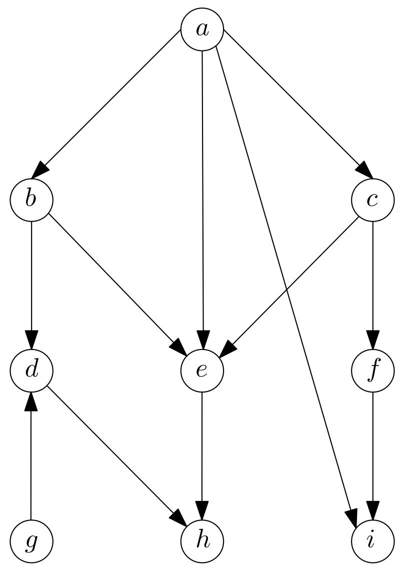
CSCI 3110 Tutorial 2

Reviewed May 24, 2019

1.



G_1



G_2

In this question we will try out some of the graph algorithms we have seen in class. Draw the annotated graphs that result from applying each of the following algorithms. You do not need to show the running states of the algorithms (i.e. the stacks, queues, etc) just the final result. Whenever you have a choice of visiting two vertices, visit them in alphabetic order (i.e. a before c, or u before v). Assume that edges are added to the stack or queue in the correct order so that they will be removed in alphabetic order. This also means that each of the algorithms should begin at a.

(a) Bipartitness Testing on graph G_1 to obtain the BFS tree. Is the graph bipartite? If not then report the odd cycle and remove the edge of any odd cycle with the alphabetically largest vertices until the graph is bipartite and run the algorithm again. Colour the vertices depending on whether they are at an even or odd level (black and white is fine) and draw the non-tree edges as dashed lines. What is the bipartitite partitioning?

(b) Topological Sorting to obtain the DFS tree. Number the vertices and draw the non-tree edges as dashed lines. What is the topological ordering?

2. The transpose of a directed graph $G = (V, E)$ is another directed graph $G^T = (V, E^T)$ on the same vertex set, but with all edges reversed; that is, $E^R = \{(v, u) : (u, v) \in E\}$.

   Give a linear-time algorithm for computing the reverse of a graph in adjacency list format. Then, argue that your algorithm (i) terminates, (ii) is correct, and (iii) has the claimed running time.