Tutorial 2

Instructor: Meng-Fen Chiang

COMPSCI: WEEK 12.3

Lab

- Given a 3×3 grid filled with non-negative numbers [[1,3,1],[1,5,1],[4,2,1]], design a single source the shortest path algorithm of your choice to find a path from the top left to bottom right, which minimizes the sum of all numbers along its path.

- Grid:
  - [1,3,1]
  - [1,5,1]
  - [4,2,1]

OUTLINE

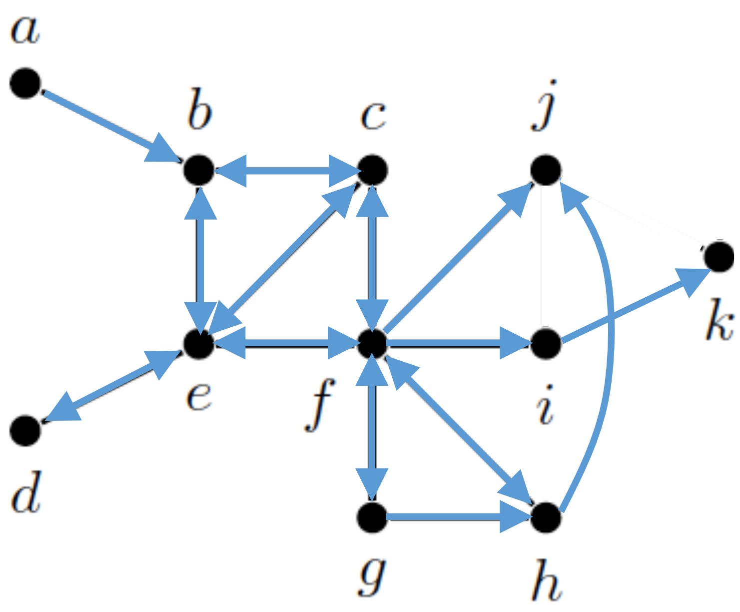
- Question 1: Graph Definition
- Question 2: DFS
- Question 3: BFS
- Question 4: Topological Ordering
- Question 5: Cycles
- Question 6: Girths
- Question 7: Mixed
- Question 8: Girths

Question 1 (A): Graph Definition

- Consider the adjacency list of a digraph G below:

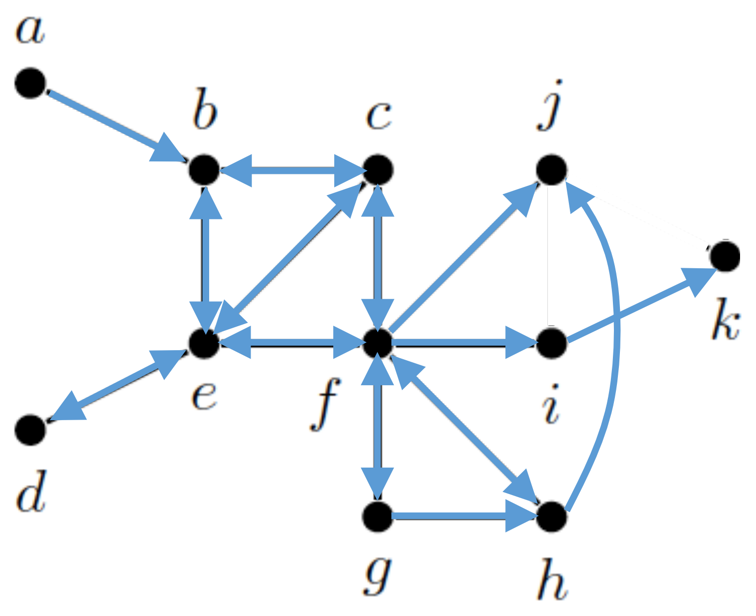
$$
\begin{array}{ll}
a: & b \\
b: & c, e \\
c: & b, e, f \\
d: & e \\
e: & b, c, d, f \\
f: & c, e, g, h, i, j \\
g: & f, h \\
h: & f, j \\
i: & k
\end{array}
$$

Draw the Digraph G



Question 1 (B): Graph Definition

- What is the source node and sink node of G?

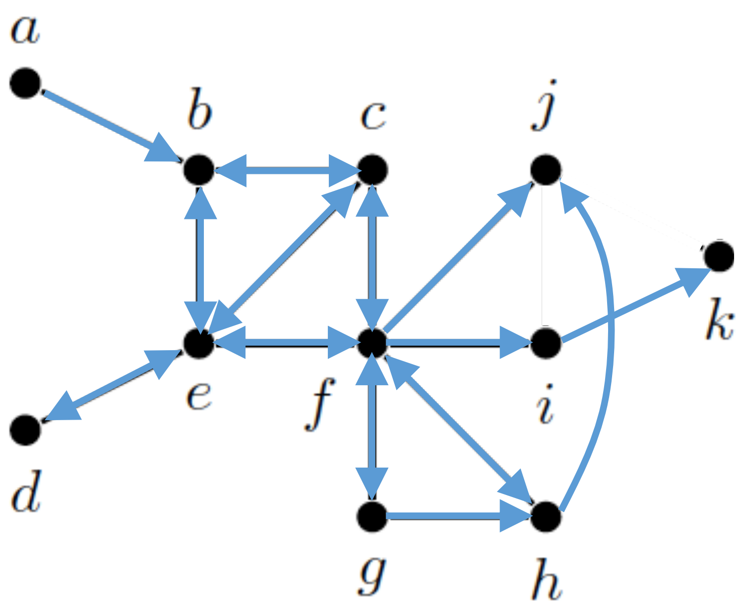


- Solution
  - Source nodes: {a}
  - Sink nodes: {j, k}

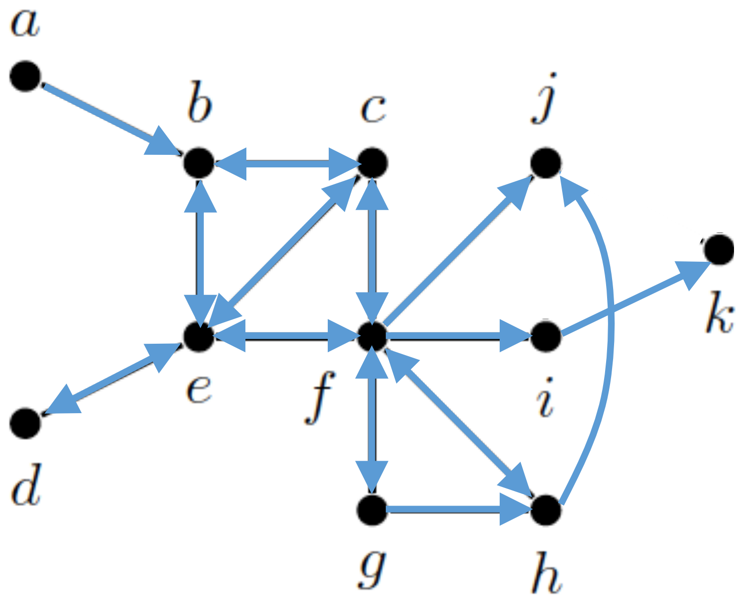
Question 1 (C): Graph Definition

- What is the adjacency matrix of this digraph G?

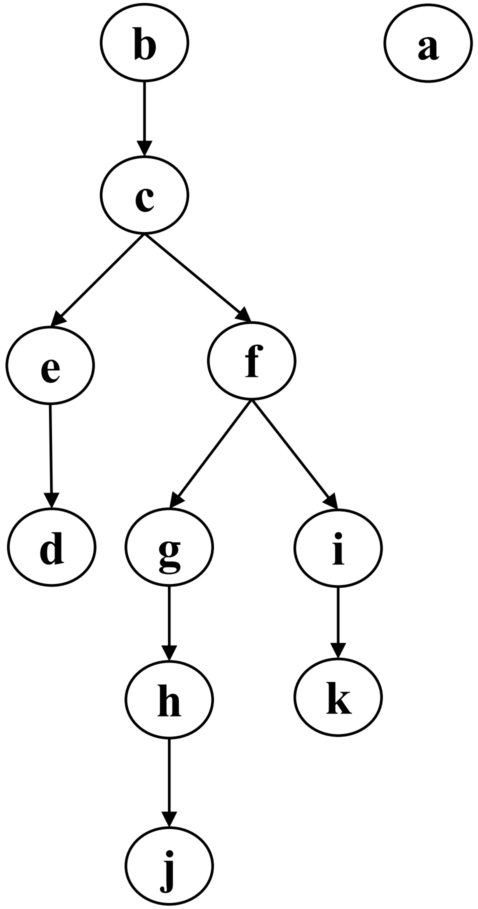
$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



Question 2: DFS

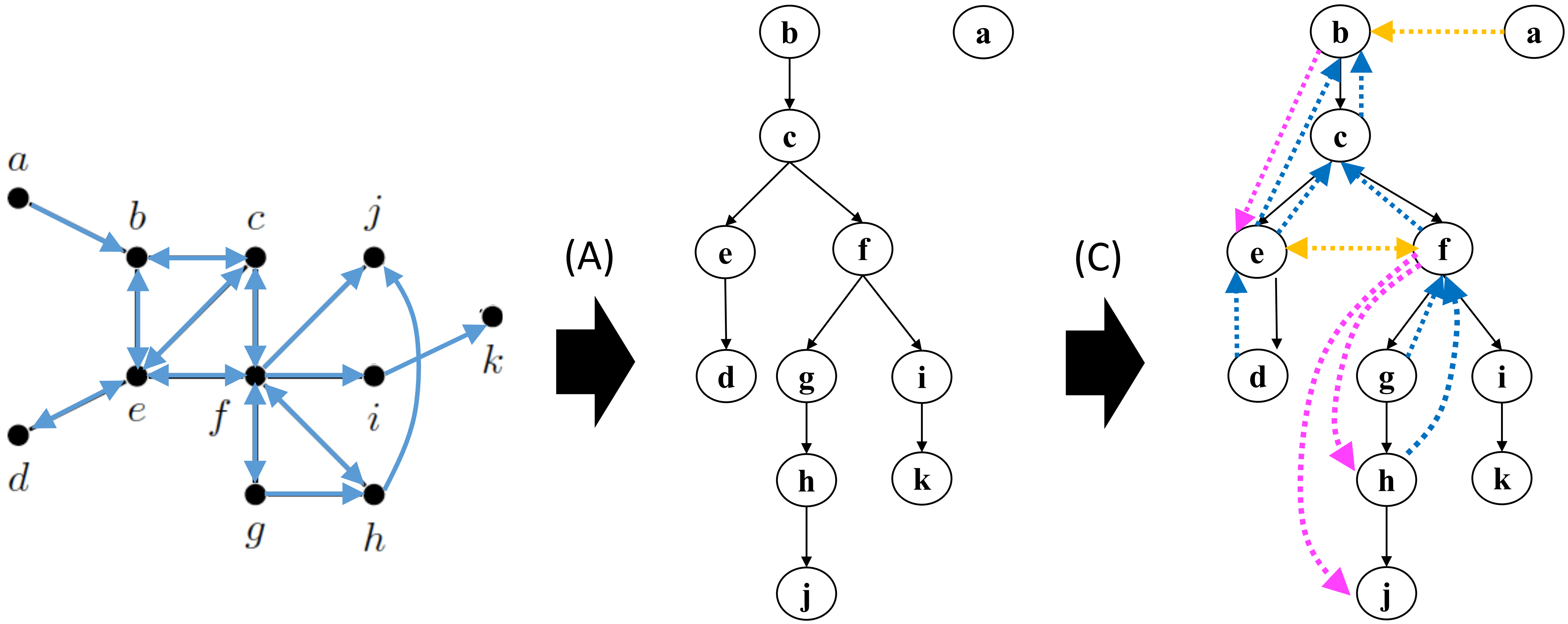


(A)

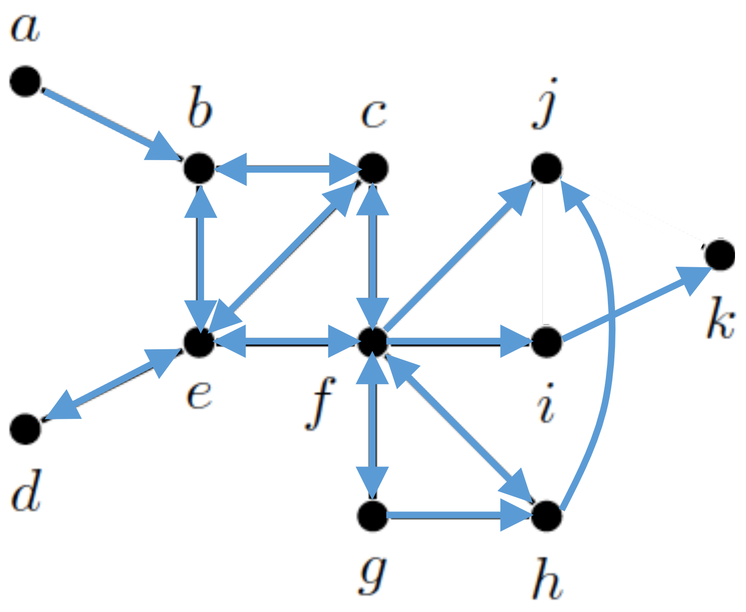


(B)

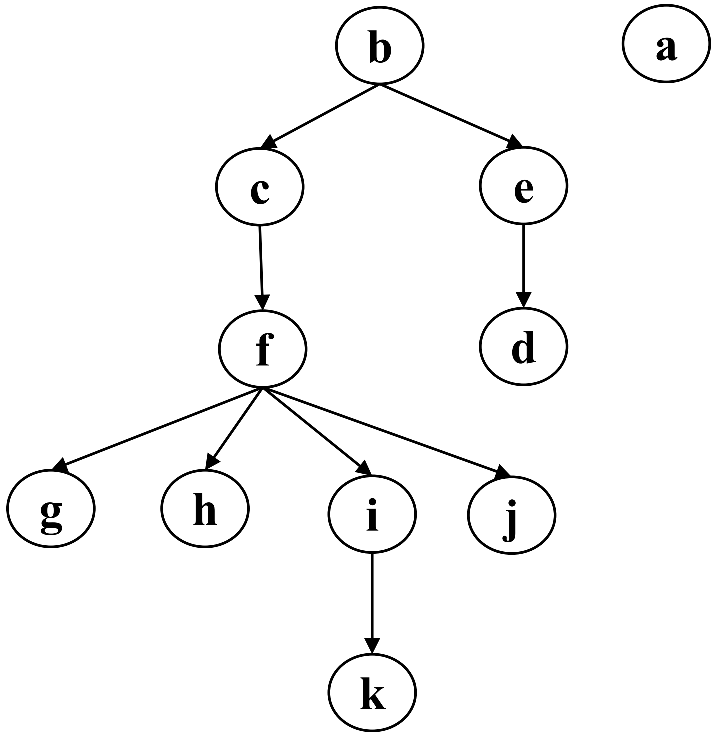
Stack: []
Stack: [b
Stack: [b,c]
Stack: [b,c,e]
Stack: [b,c,e,d]
Stack: [b,c,e]
Stack: [b,c]
Stack: [b,c,f]
Stack: [b,c,f,g]
Stack: [b,c,f,g,h]
Stack: [b,c,f,g,h,j]
Stack: [b,c,f,g,h]
Stack: [b,c,f,g]
Stack: [b,c,f,i]
Stack: [b,c,f,i,k]
Stack: [b,c,f,i]
Stack: [b,c,f]
Stack: [b,c]
Stack: [b]
Stack: []
Stack: [a]
Stack: []


a
b
c
j
k
e
f
i
d
g
h
(A)
b
a
c
e
f
d
g
i
h
k
j
(C)
b
a
c
e
f
d
g
i
h
k
j

Question 3: BFS

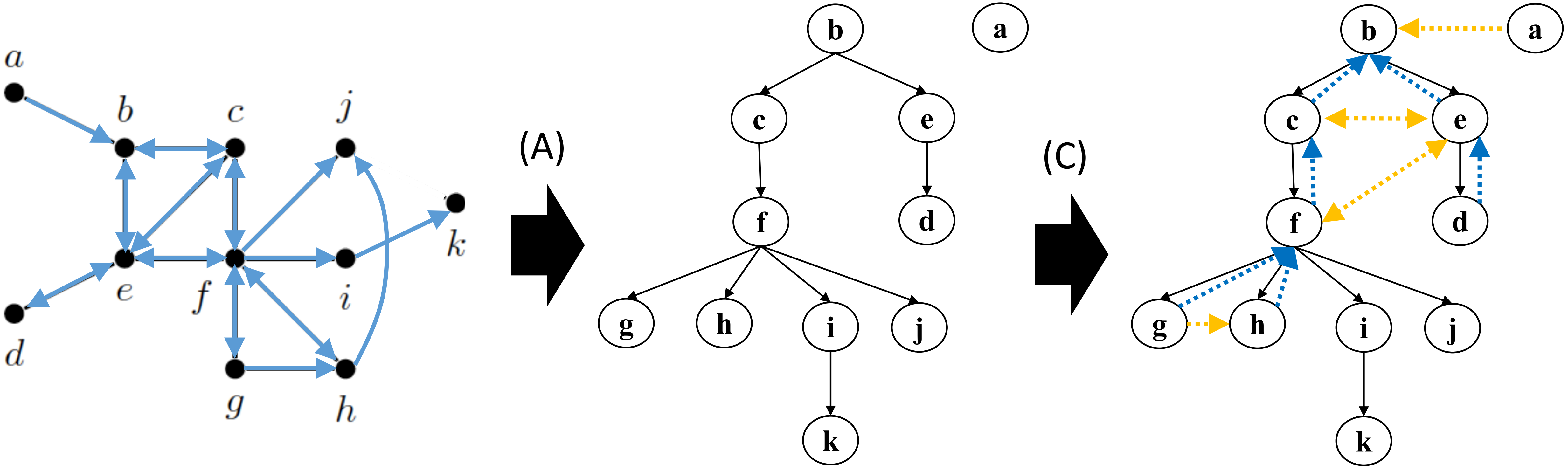


(A)



(B)

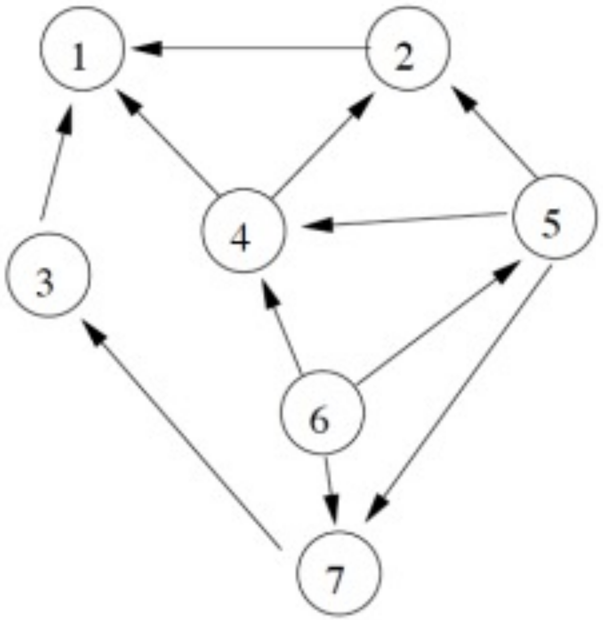
Queue: []
Queue: [b]
Queue: [c,e]
Queue: [f,e]
Queue: [g,h,i,j,e]
Queue: [h,i,j,e]
Queue: [i,j,e]
Queue: [k,j,e]
Queue: [e]
Queue: [d]
Queue: []
Queue: [a]
Queue: []


a
b
c
j
k
e
f
i
d
g
h
(A)
b
a
c
e
f
d
g
h
i
j
k
(C)
b
a
c
e
f
d
g
h
i
j
k

Question 4: Topological Ordering

- Consider the following directed graph G. Does the graph G have a topological sorting?
- If so give one with your working, if not why not?

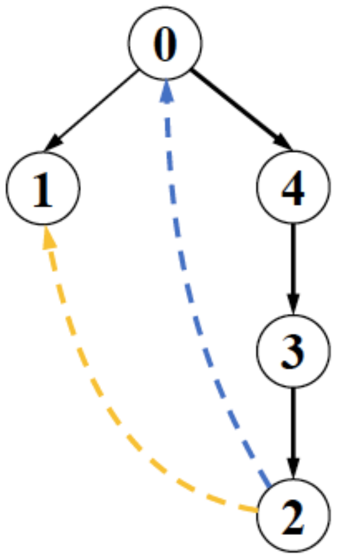
- Solution: Yes
  - [6,5,4,2,7,3,1] or
  - [6,5,4,7,3,2,1] or
  - [6,5,4,7,2,3,1] or
  - [6,5,7,3,4,2,1] or
  - [6,5,7,4,2,3,1] or
  - [6,5,7,4,2,2,1]



Question 5: Cycles

- Let v be a node of G. Which of the following lists the main steps in finding the length of a shortest directed cycle in a digraph G that contains v?

A. Run BFS starting from v and stop when a back arc (x, v) is found.
B. Run DFS starting from v and stop when a cross arc (x, v) is found.
C. Run BFS starting from v and stop when a cross arc (x, v) is found.
D. Run DFS starting from v and stop when a back arc (x, v) is found.
E. Run DFS and count the number of sink nodes in G.



Question 6: Girths

- Can we use Dijkstra's algorithm to calculate the girth?
- If yes, how can we achieve that?

- Solution
  - For each edge (u, v), if we remove the edge, and calculate the shortest path between u and v.
    - If there is a path between (u, v), then we have a cycle passing through u and v by adding back the edge, and the girth is the distance between u and v plus 1, i.e., d(u, v) + 1.
  - We run Dijkstra on one end node of each edge and get the shortest cycle

Question 7: Mixed

- Which of the following statements is FALSE?

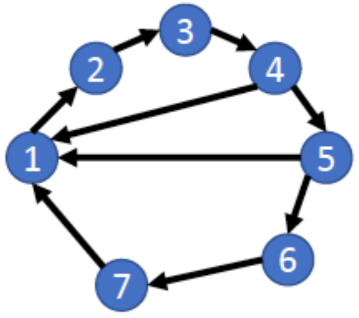
A. Zero-indegree sorting can be used to find a topological order of a graph.

B. Suppose DFS is run on a digraph G. Then G is acyclic if and only if there are no cross arcs.

C. BFS can be used to find the connected components of a graph.

D. The girth of a directed graph can be strictly larger than its directed girth.

E. A digraph has a topological ordering if and only if it is acyclic.



Girth: 3
Directed girth: 4

Question 8 (A): Girth

(A) Why is there no need to continue to the end of the level before halting the traversal?

1. For all nodes $v \in V(G)$ do:
   (a) Run BFSVISIT from node v.
   (b) As soon as the algorithm finds a back arc of the form (x, v), terminate, recording the length of such a cycle c, which will be $h + 1$, where h is the depth of node x in the given search tree.
2. Return smallest c.

Solution:

- In the algorithm for finding girth, we NEED to check all nodes at the same level because there could be shorter cycle found by the algorithm at the same level.
- However, in the above algorithm, all cycles found at the same level has the same length, h+1.

Question 8 (B): Girth

(B) What is the runtime?

1. For all nodes $v \in V(G)$ do:
   (a) Run BFSVISIT from node v.
   (b) As soon as the algorithm finds a back arc of the form (x, v), terminate, recording the length of such a cycle c, which will be $h + 1$, where h is the depth of node x in the given search tree.
2. Return smallest c.

Solution:

- The loop executes n times. For each loop, we may need to traverse the whole graph in the worst case:
  - O$(n + m)$ for adjacency list,
  - O(n^2) for adjacency matrix.
- Thus, the overall worst-case running time complexity is O$(n(n + m))$ for adjacency list and O(n^3) for adjacency matrix.