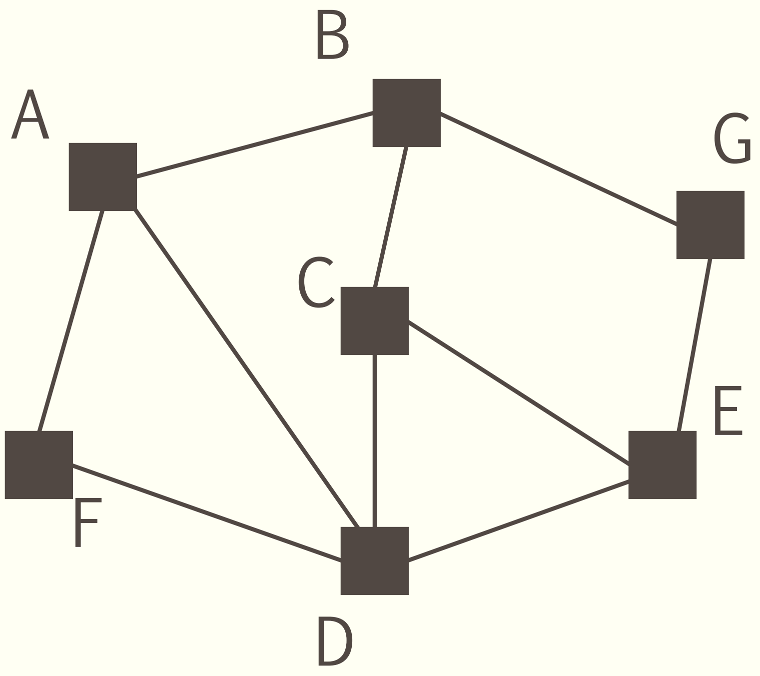
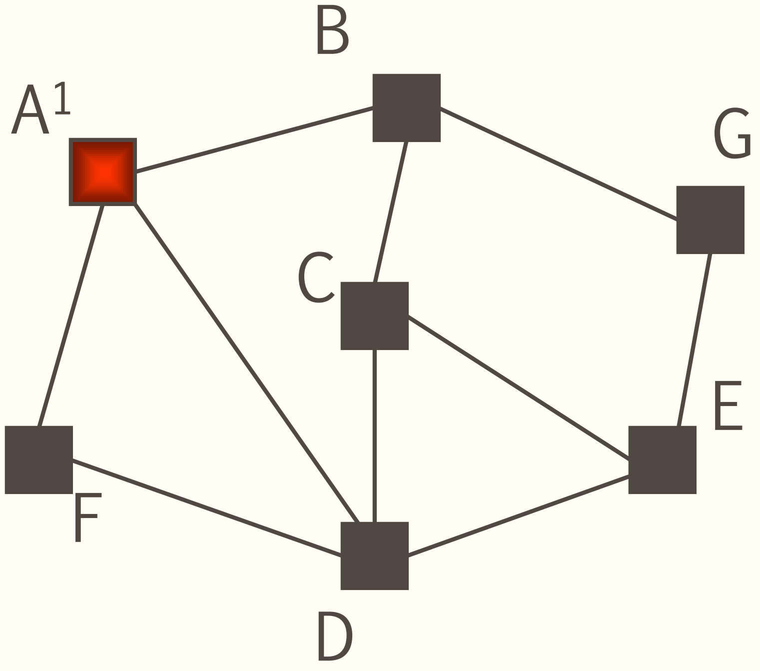
LECTURE 07 – DEPTH FIRST SEARCH

Paul Doliotis (PhD)
Adjunct Assistant Professor
Portland State University

Example of DFS

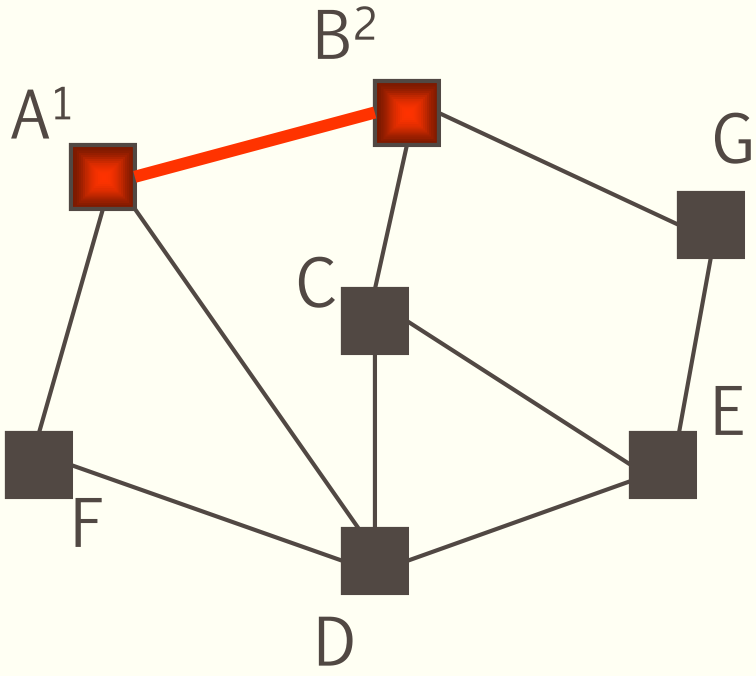


Example of DFS: Step 1



DFS(A)

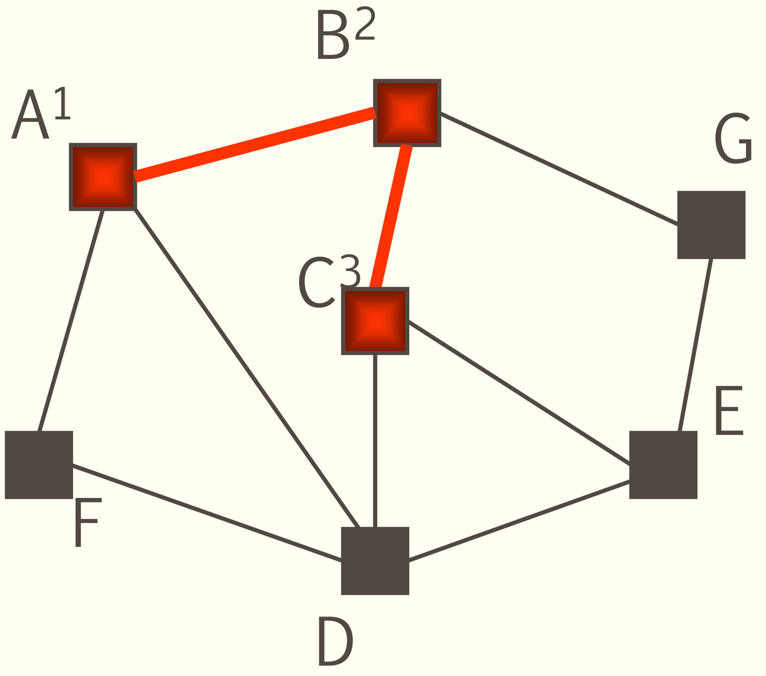
Example of DFS: Step 2



DFS(A)
DFS(B)

Red edges are called **"Tree edges"**, every unvisited vertex is attached to it's parent

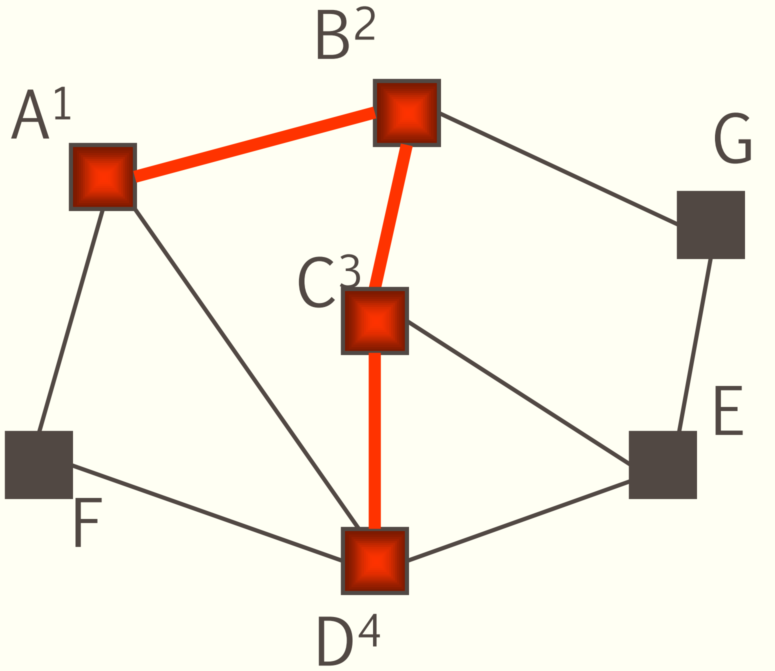
Example of DFS: Step 3



DFS(A)
DFS(B)
DFS(C)

Red edges are called **“Tree edges”**, every unvisited vertex is attached to it’s parent

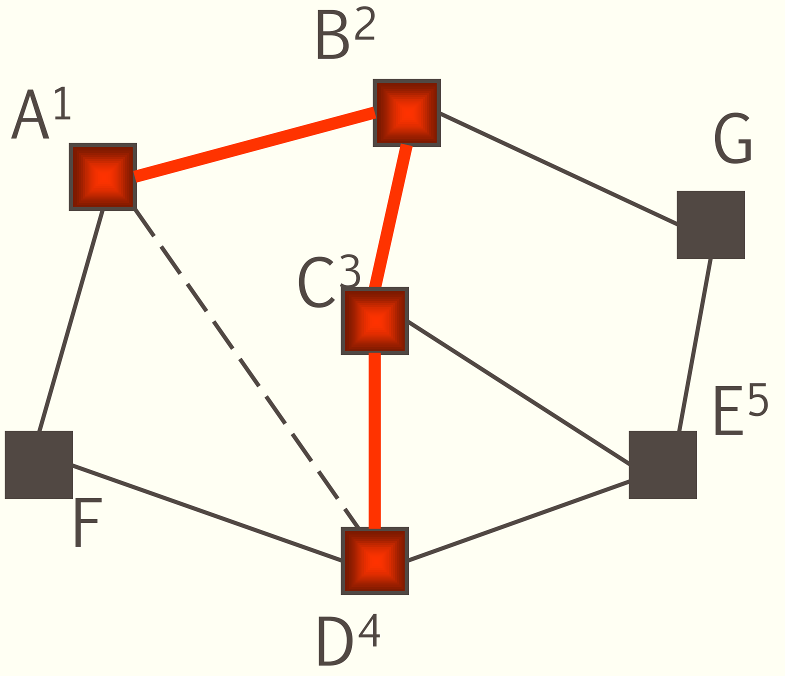
Example of DFS: Step 4



DFS(A)
DFS(B)
DFS(C)
DFS(D)

Red edges are called **"Tree edges"**, every unvisited vertex is attached to it's parent

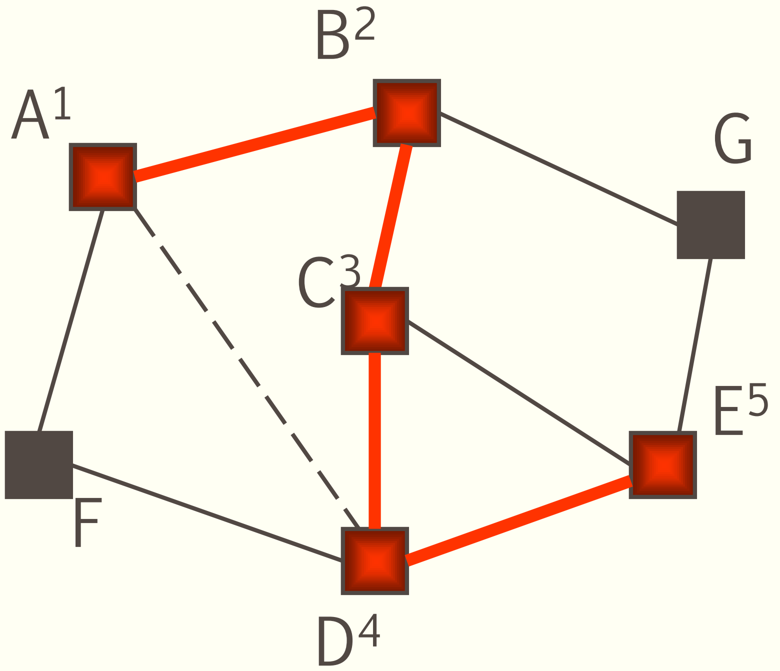
Example of DFS: Step 5



DFS(A)
DFS(B)
DFS(C)
DFS(D)
~~DFS(A)~~

Red edges are called **“Tree edges”**, every unvisited vertex is attached to it’s parent

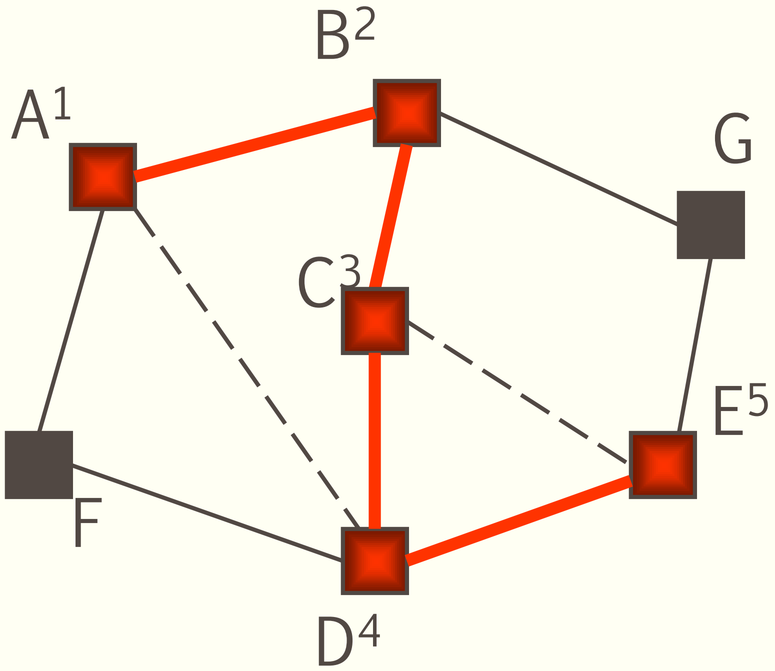
Example of DFS: Step 6



DFS(A)
DFS(B)
DFS(C)
DFS(D)
DFS(E)

Red edges are called **"Tree edges"**, every unvisited vertex is attached to it's parent

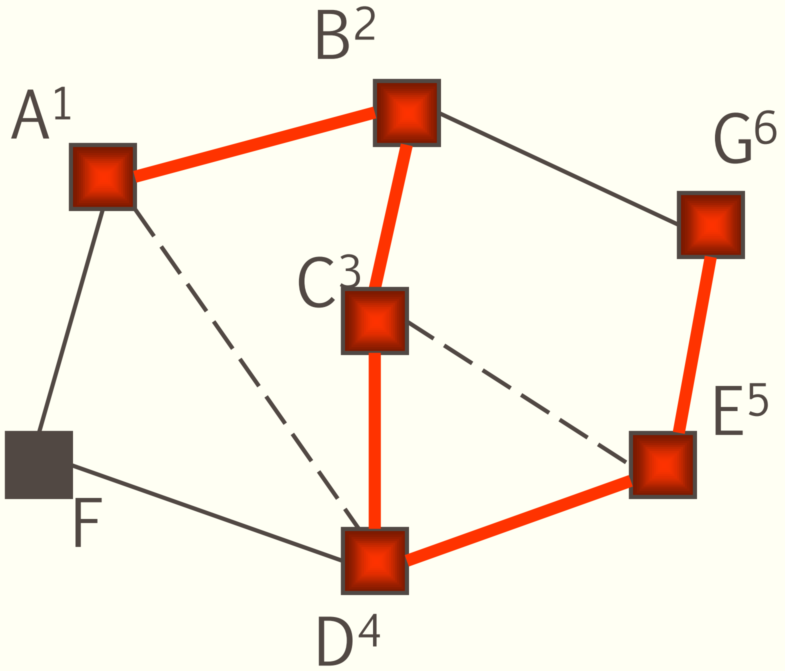
Example of DFS: Step 7



DFS(A)
DFS(B)
DFS(C)
DFS(D)
DFS(E)
~~DFS(C)~~

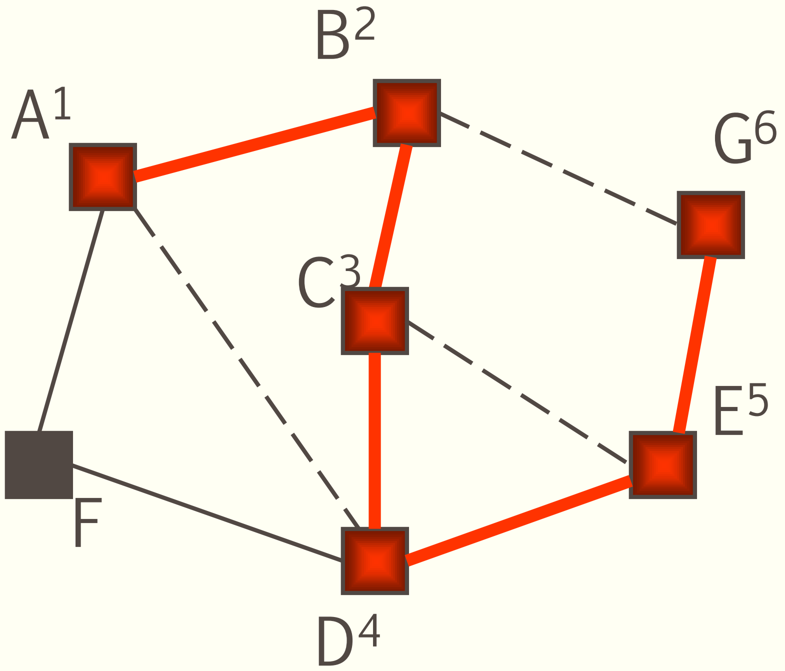
Black edges are called **“Back edges”**, because it connects vertex to an anchestor

Example of DFS: Step 8



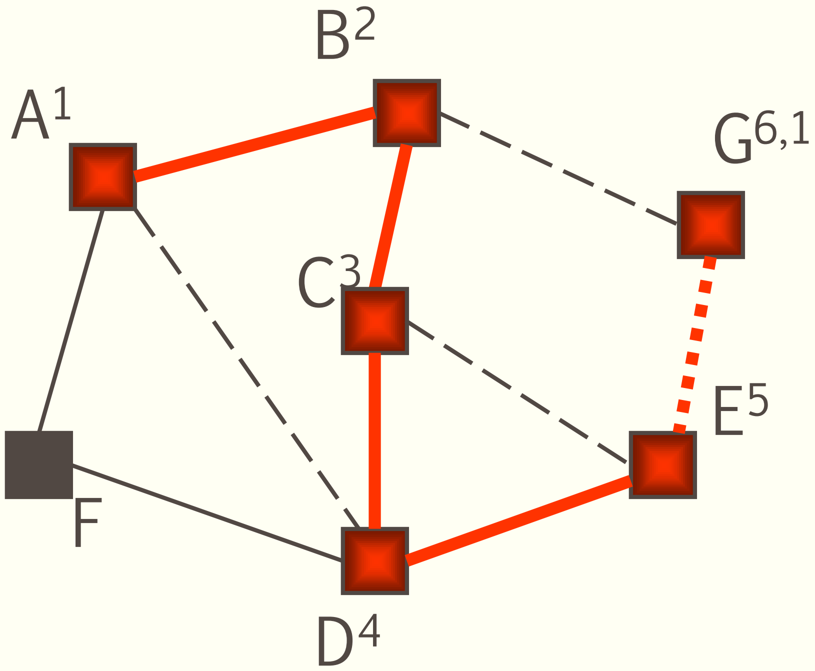
DFS(A)
DFS(B)
DFS(C)
DFS(D)
DFS(E)
DFS(G)

Example of DFS: Step 9



DFS(A)
DFS(B)
DFS(C)
DFS(D)
DFS(E)
DFS(G)
~~DFS(B)~~

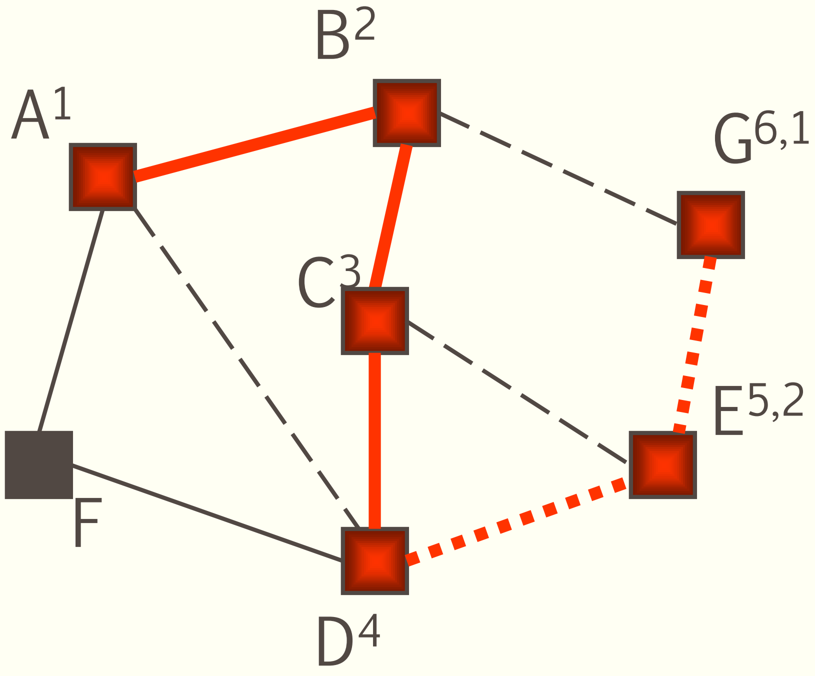
Example of DFS: Step 10



DFS(A)
DFS(B)
DFS(C)
DFS(D)
DFS(E)

Left superscript denotes order by which the element has been pushed into the stack. Right superscript denotes order by which element has been popped out of the stack.

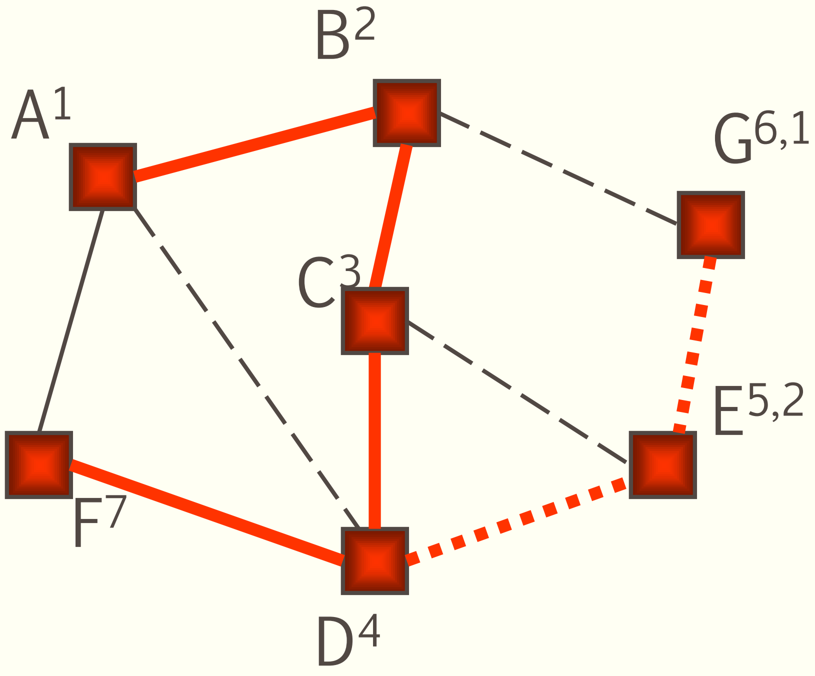
Example of DFS: Step 11



DFS(A)
DFS(B)
DFS(C)
DFS(D)

Left superscript denotes order by which the element has been pushed into the stack. Right superscript denotes order by which element has been popped out of the stack.

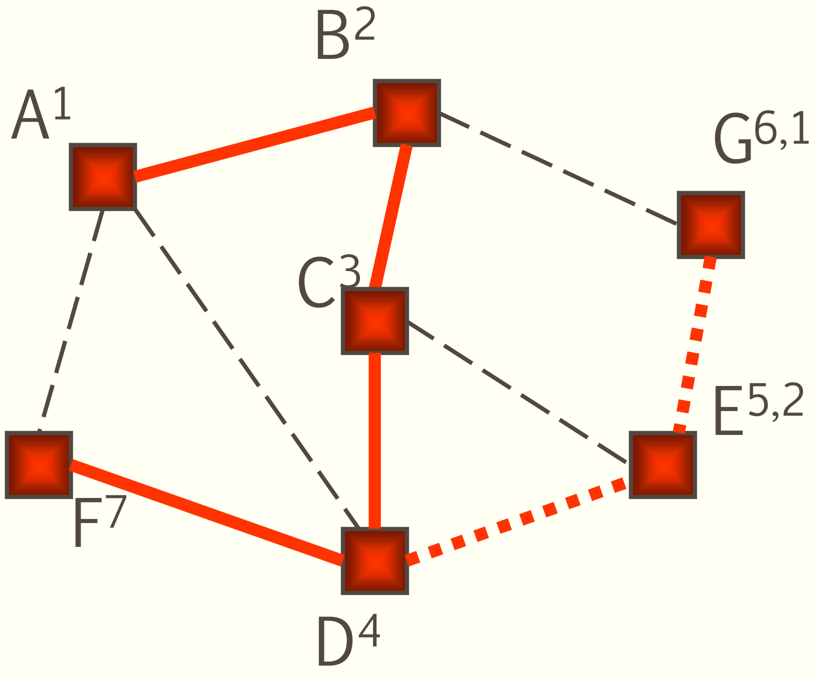
Example of DFS: Step 12



DFS(A)
DFS(B)
DFS(C)
DFS(D)
DFS(F)

Left superscript denotes order by which the element has been pushed into the stack. Right superscript denotes order by which element has been popped out of the stack.

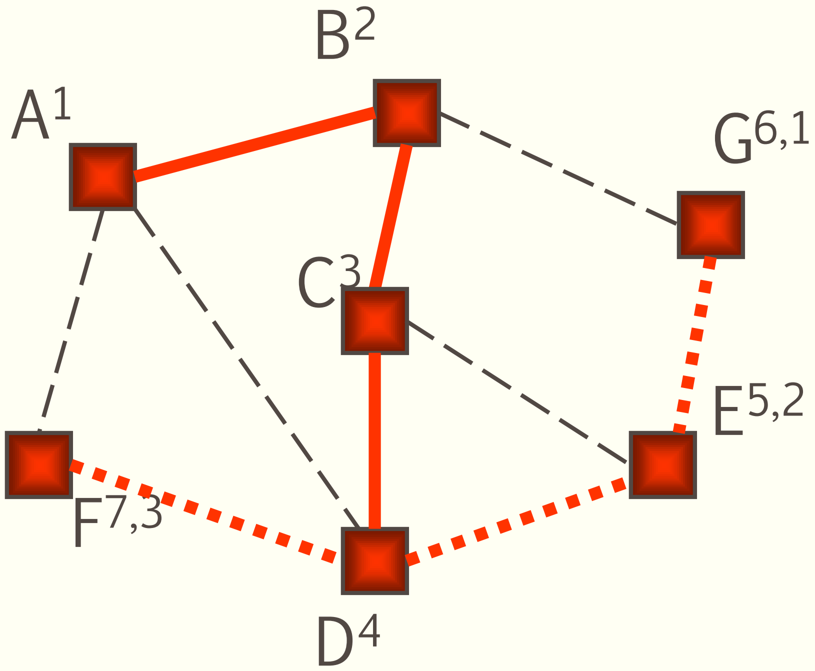
Example of DFS: Step 13



DFS(A)
DFS(B)
DFS(C)
DFS(D)
DFS(F)

Left superscript denotes order by which the element has been pushed into the stack. Right superscript denotes order by which element has been popped out of the stack.

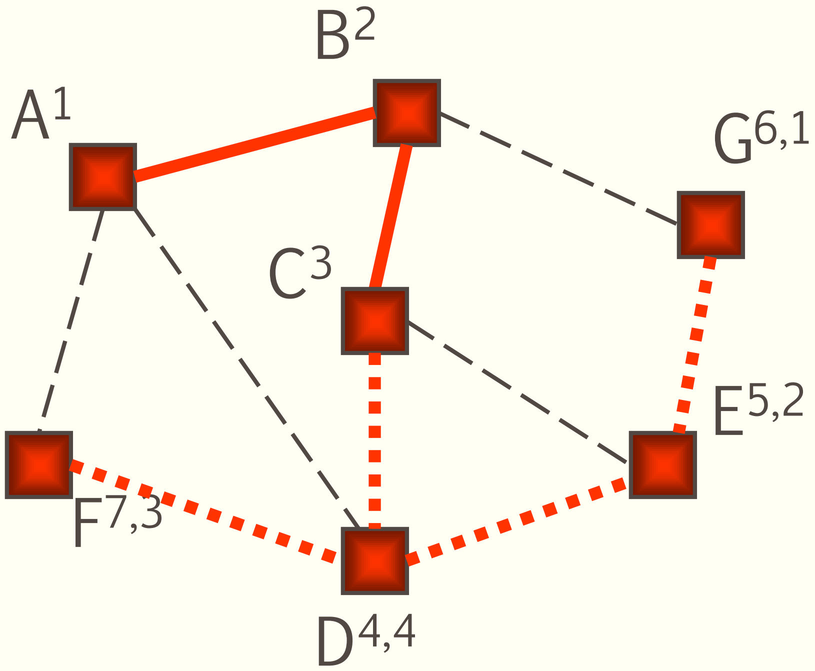
Example of DFS: Step 14



DFS(A)
DFS(B)
DFS(C)
DFS(D)

Left superscript denotes order by which the element has been pushed into the stack. Right superscript denotes order by which element has been popped out of the stack.

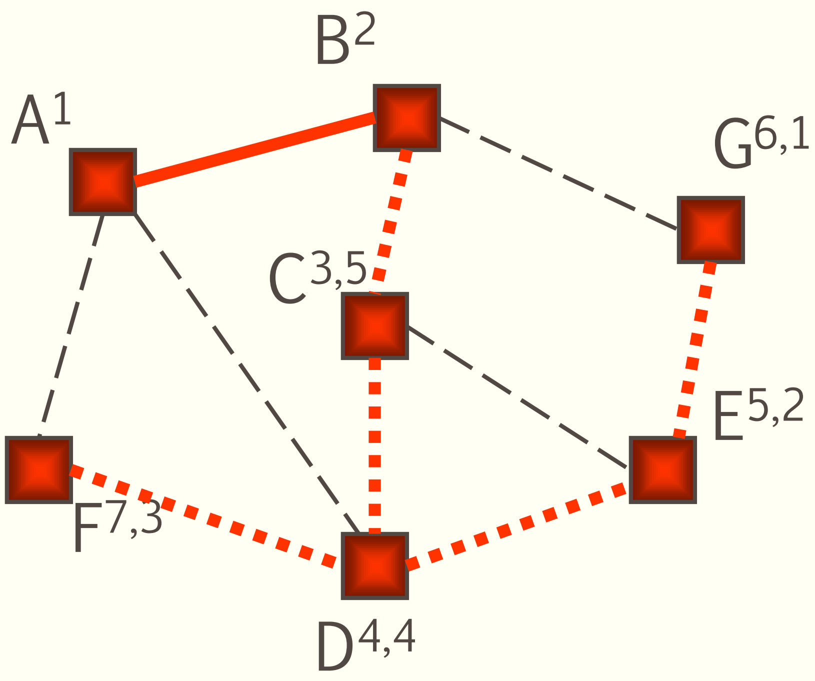
Example of DFS: Step 15



DFS(A)
DFS(B)
DFS(C)

Left superscript denotes order by which the element has been pushed into the stack. Right superscript denotes order by which element has been popped out of the stack.

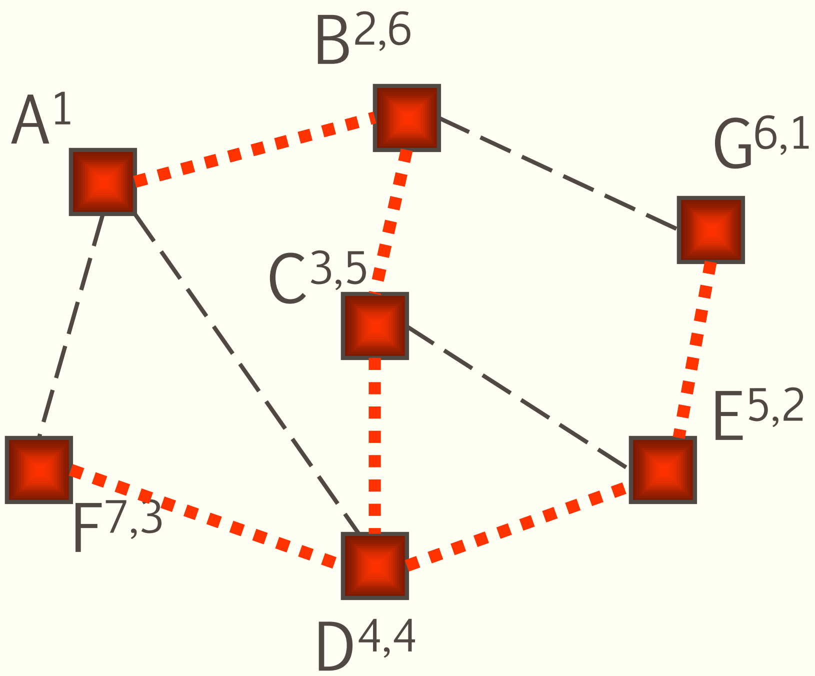
Example of DFS: Step 16



DFS(A)
DFS(B)

Left superscript denotes order by which the element has been pushed into the stack. Right superscript denotes order by which element has been popped out of the stack.

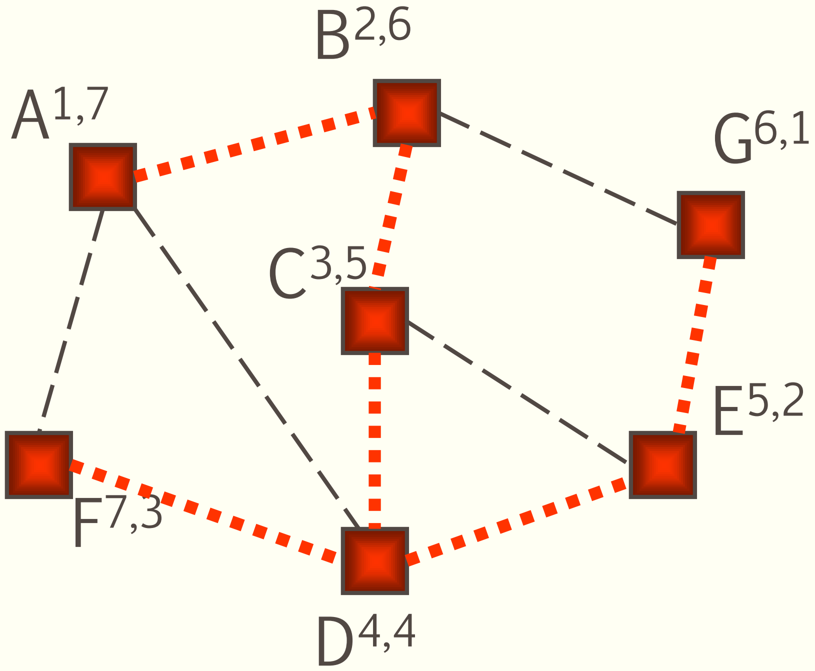
Example of DFS: Step 17



DFS(A)

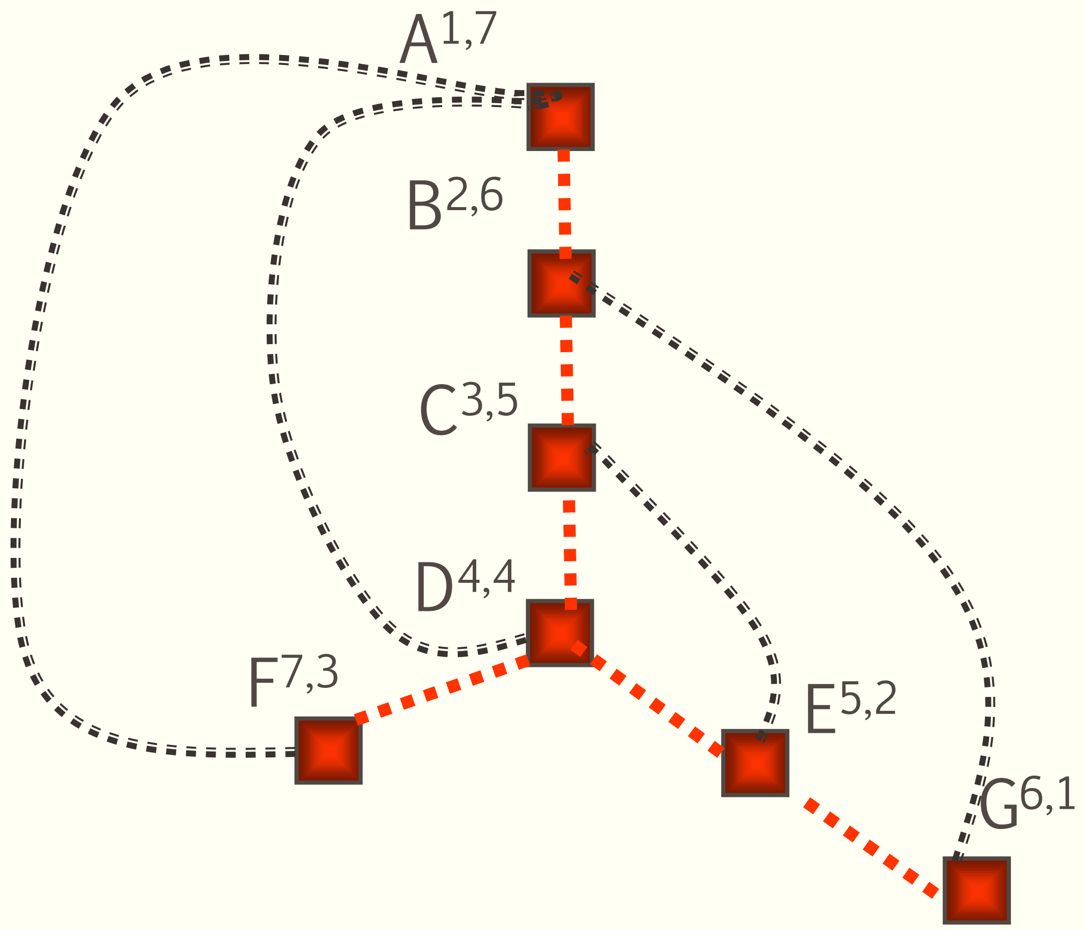
Left superscript denotes order by which the element has been pushed into the stack. Right superscript denotes order by which element has been popped out of the stack.

Example of DFS: Step 18

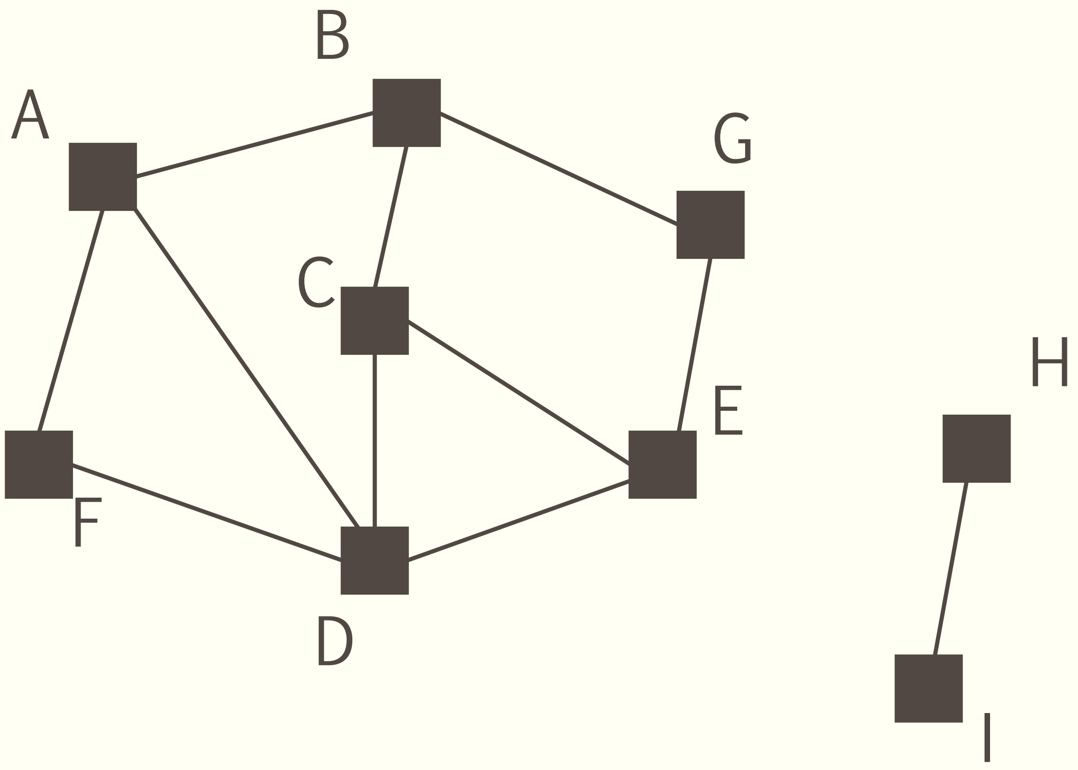


All nodes are marked so graph is connected

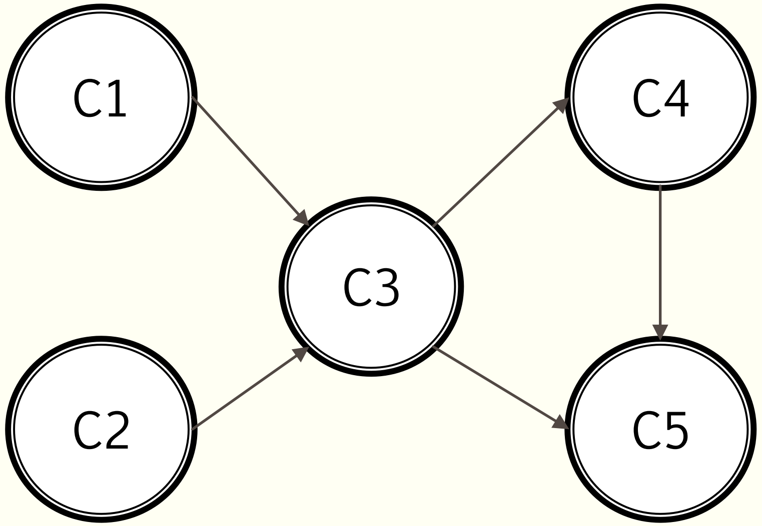
Example of DFS



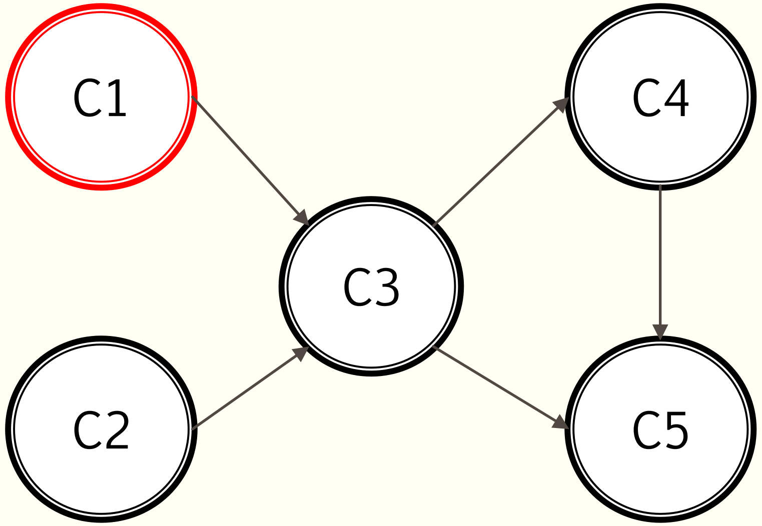
Example 2 of DFS



Example 3 of DFS



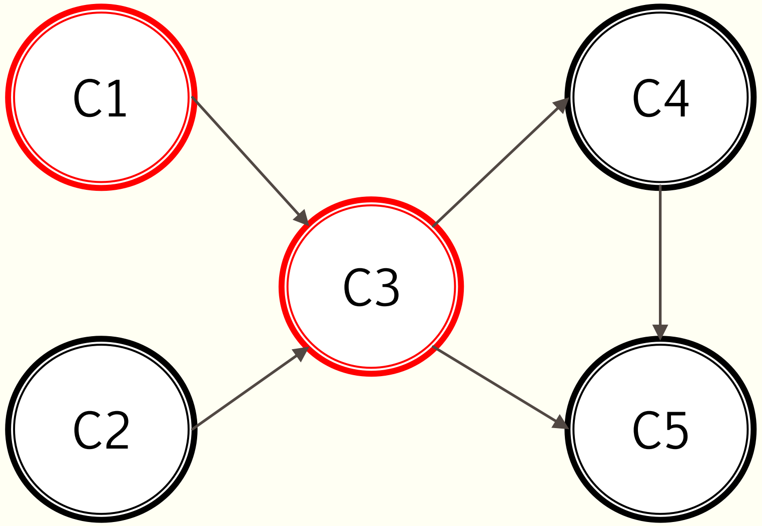
Example 3 of DFS



Order into stack	Order out of stack
C1	

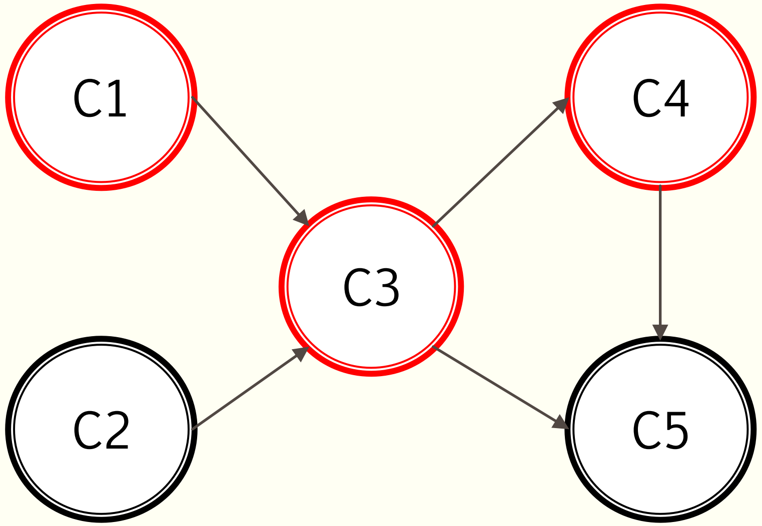
Choose a source node for Topological Sorting

Example 3 of DFS



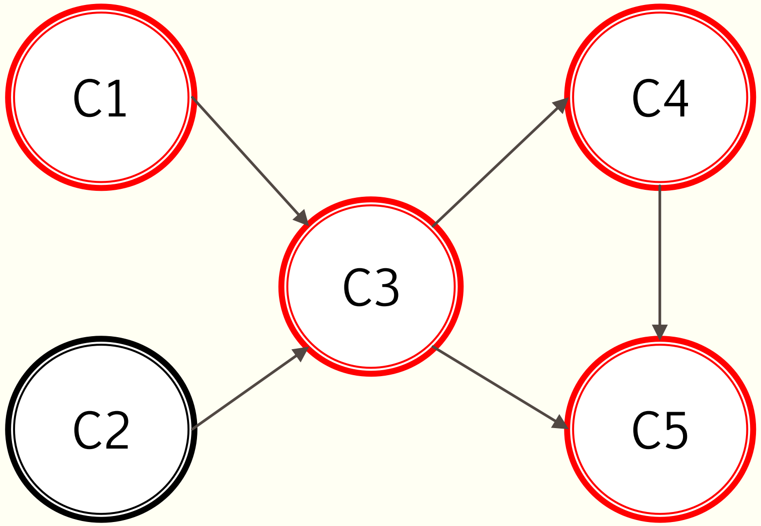
Order into stack	Order out of stack
C1	
C3	

Example 3 of DFS



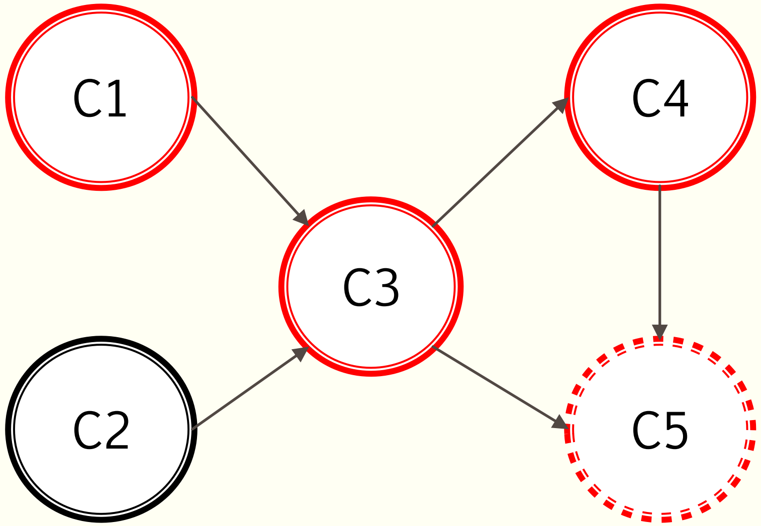
Order into stack	Order out of stack
C1	
C3	
C4	

Example 3 of DFS



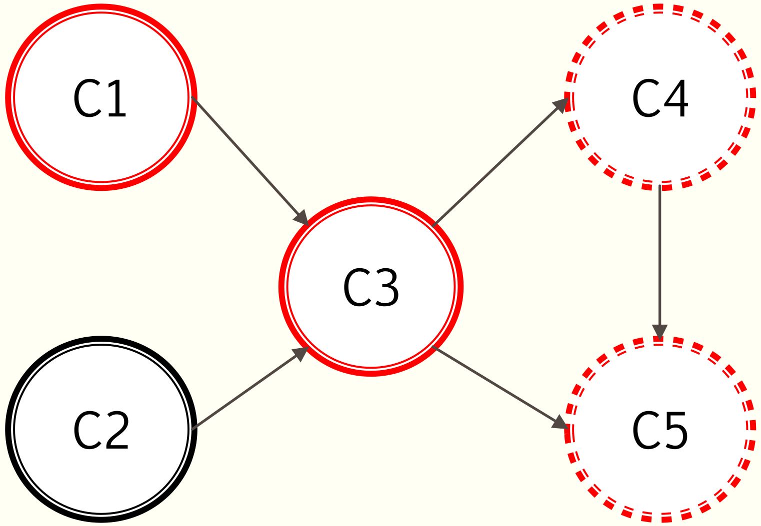
Order into stack	Order out of stack
C1	
C3	
C4	
C5	

Example 3 of DFS



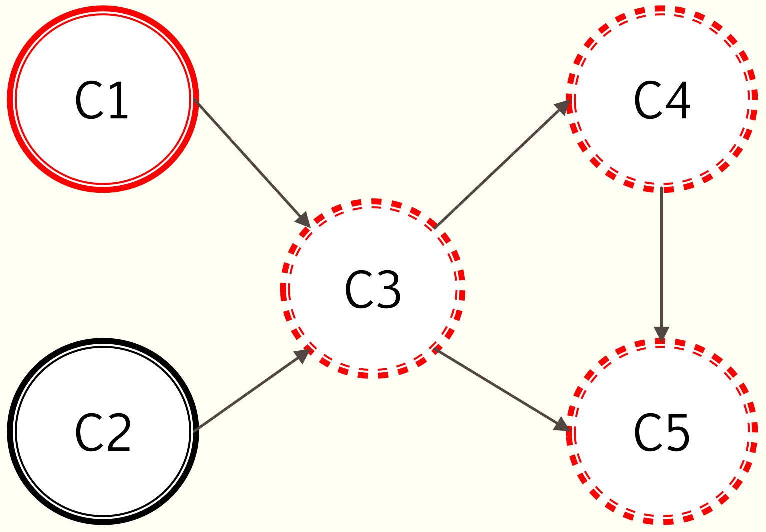
Order into stack	Order out of stack
C1	C5
C3	
C4	
C5	

Example 3 of DFS



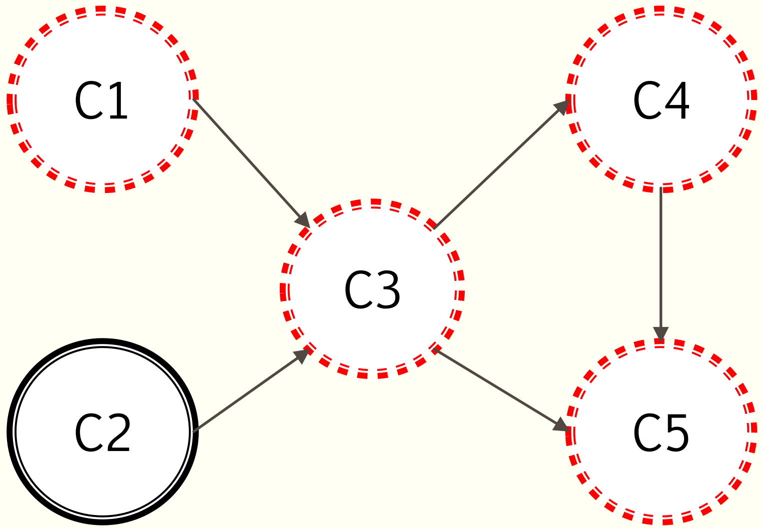
Order into stack	Order out of stack
C1	C5
C3	C4
C4	
C5	

Example 3 of DFS



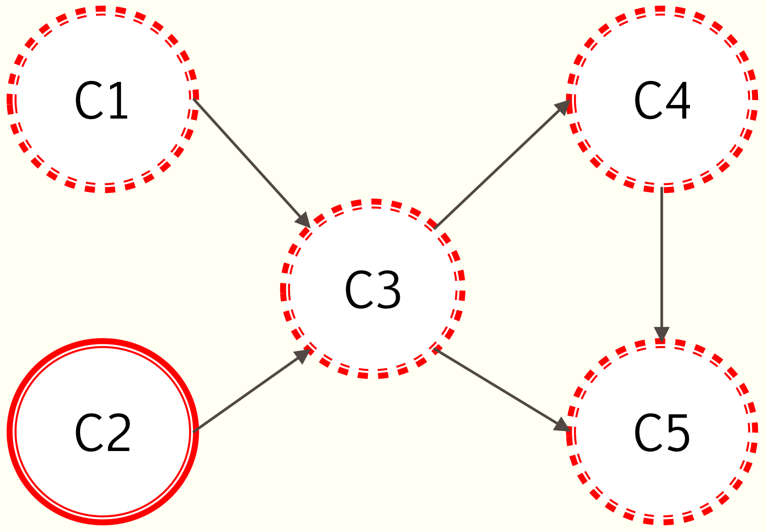
Order into stack	Order out of stack
C1	C5
C3	C4
C4	C3
C5	

Example 3 of DFS



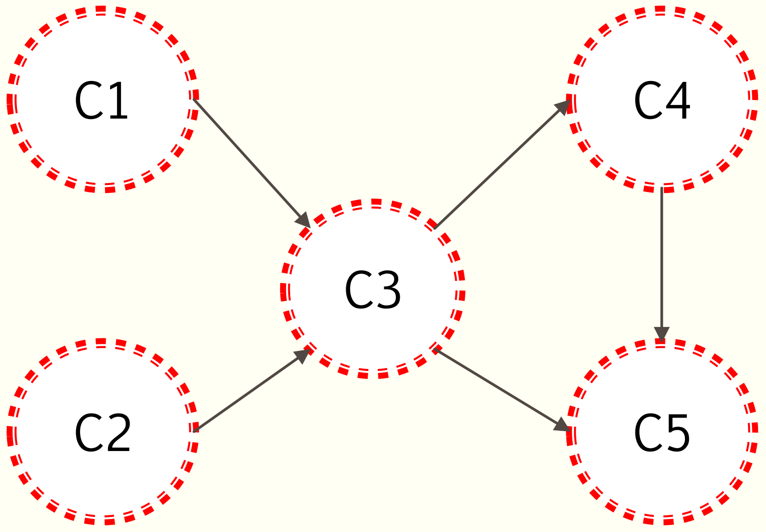
Order into stack	Order out of stack
C1	C5
C3	C4
C4	C3
C5	C1

Example 3 of DFS



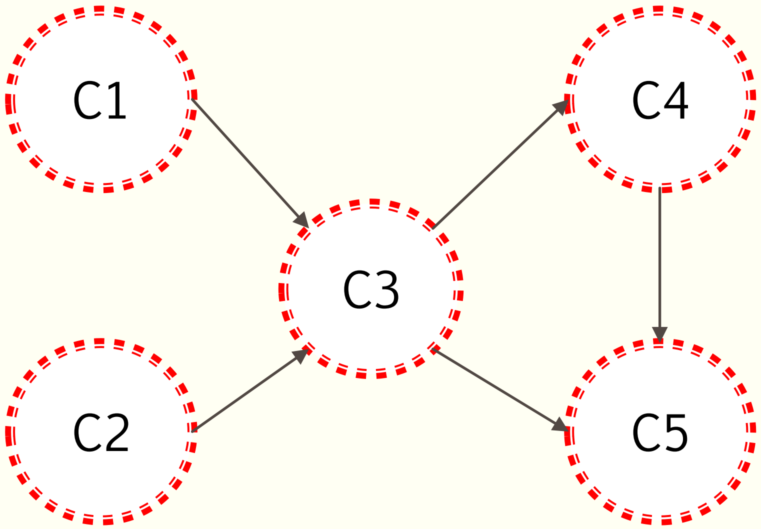
Order into stack	Order out of stack
C1	C5
C3	C4
C4	C3
C5	C1
C2	

Example 3 of DFS

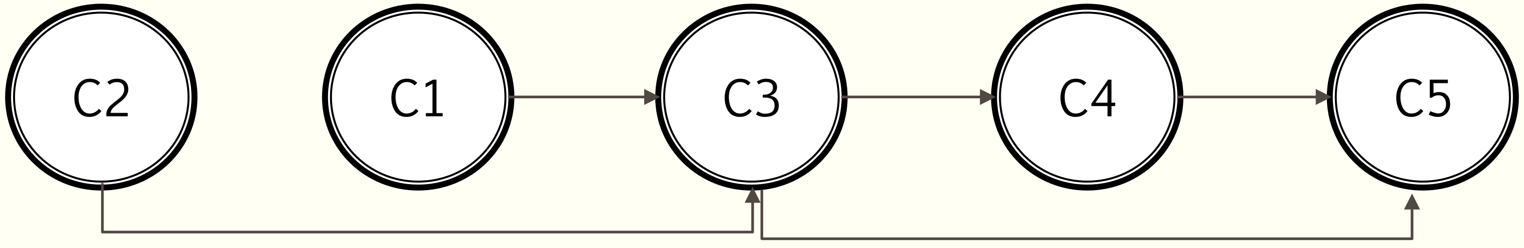


Order into stack	Order out of stack
C1	C5
C3	C4
C4	C3
C5	C1
C2	C2

Example 3 of DFS – Topological Sorting



Order into stack	Order out of stack
C1	C5
C3	C4
C4	C3
C5	C1
C2	C2



DFS complexity

- If we use adjacency matrix, $\Theta(|V|^2)$, number of vertices squared
- If we use adjacency list, $\Theta(|V|+|E|)$, number of vertices and edges
- DFS can check for <u>connectivity</u> and <u>acyclicity</u>

DFS with explicit stack

```
DFS(G,v)   ( v is the vertex where the search starts )
   Stack S := {};   ( start with an empty stack )
   for each vertex u, set visited[u] := false;
   push S, v;
   while (S is not empty) do
      u := pop S;
      if (not visited[u]) then
         visited[u] := true;
         for each unvisited neighbour w of u
            push S, w;
      end if
   end while
END DFS()
```

Topological Sorting

- Set of 5 required courses {C1, C2, C3, C4, C5}

Topological Sorting

- Set of 5 required courses {C1, C2, C3, C4, C5}
- Courses can be taken in any order, but need to follow prerequisite rules:
  - C1, C2 no prereq
  - C3 requires C1 and C2
  - C4 requires C3
  - C5 requires C3 and C4

Topological Sorting

- Set of 5 required courses {C1, C2, C3, C4, C5}
- Courses can be taken in any order, but need to follow prerequisite rules:
  - C1, C2 no prereq
  - C3 requires C1 and C2
  - C4 requires C3
  - C5 requires C3 and C4
- Student can only take 1 course per term
  - In which order should the student take the courses?

Topological Sorting – First Algorithm

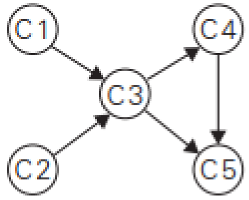


FIGURE 4.6 Digraph representing the prerequisite structure of five courses.

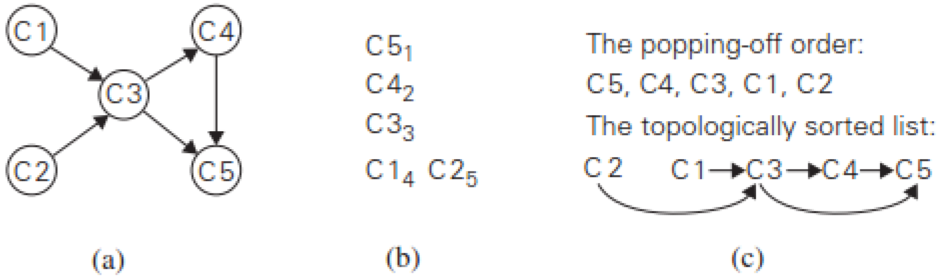
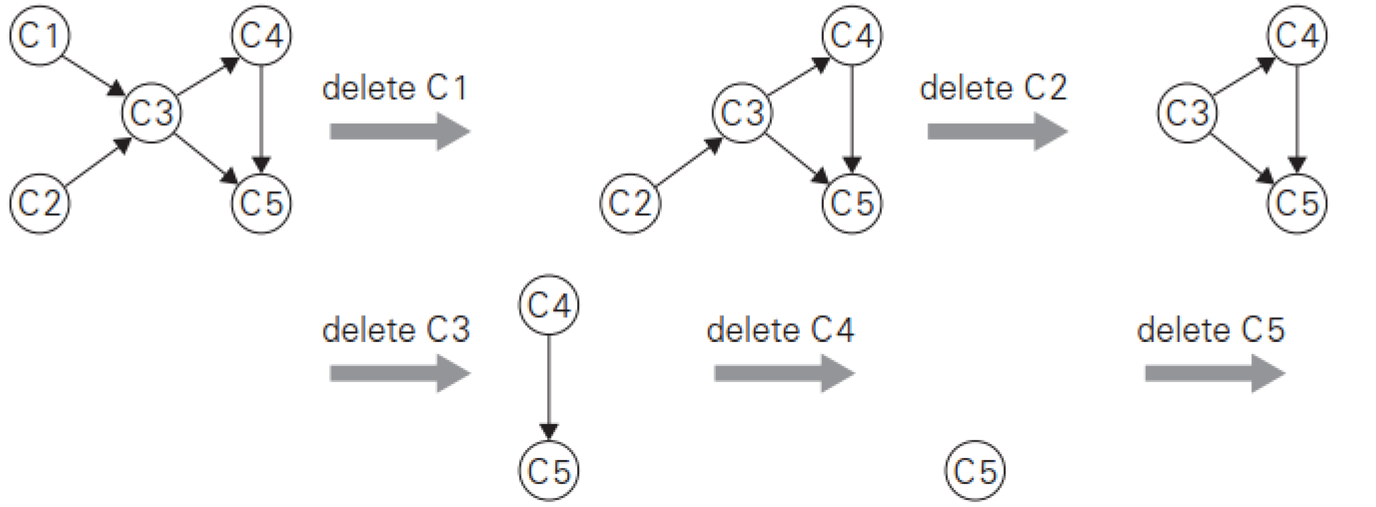


FIGURE 4.7 (a) Digraph for which the topological sorting problem needs to be solved. (b) DFS traversal stack with the subscript numbers indicating the popping-off order. (c) Solution to the problem.

Topological Sorting – Second Algorithm



The solution obtained is C1, C2, C3, C4, C5

FIGURE 4.8 Illustration of the source-removal algorithm for the topological sorting problem. On each iteration, a vertex with no incoming edges is deleted from the digraph.

Topological Sorting

- If directed graph has cycles topological sorting is not possible

BFS

- Breadth First Search graph traversal

BFS

- Breadth First Search graph traversal
- Uses queue, FIFO

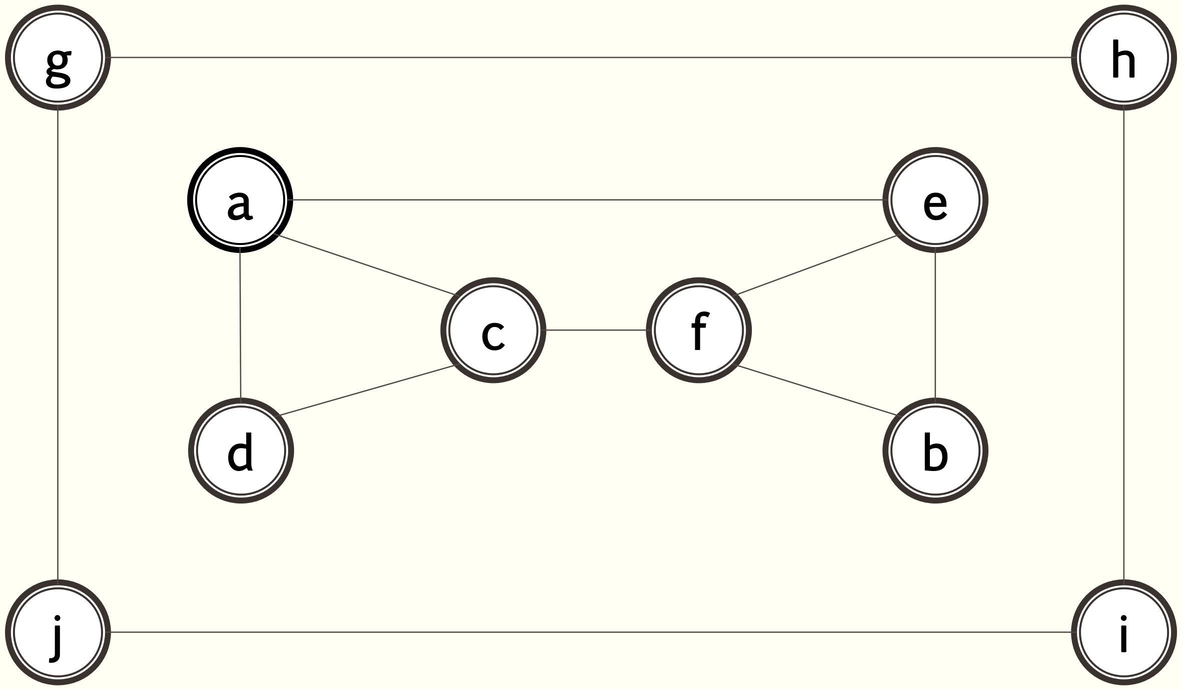
BFS

ALGORITHM *BFS*(G)

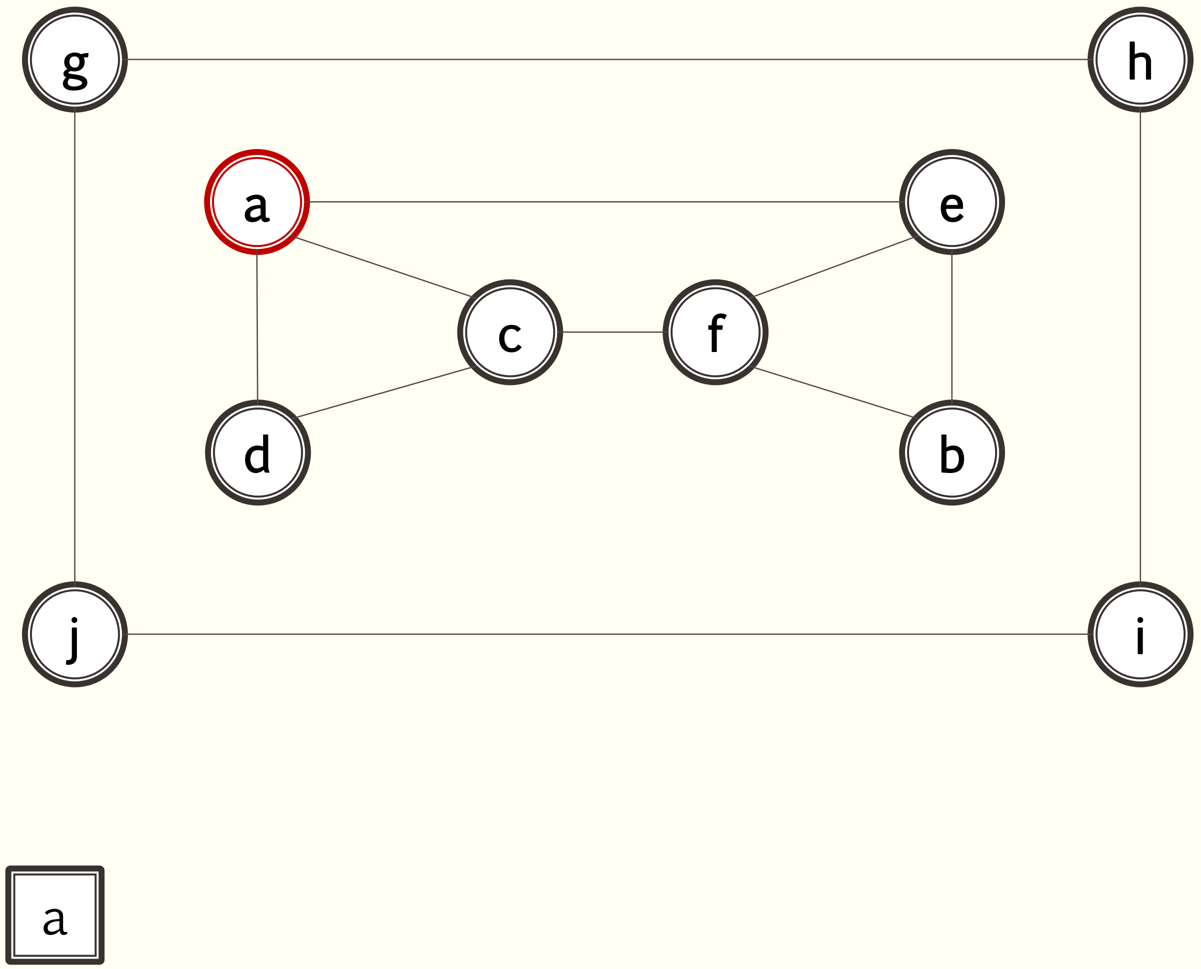
```
//Implements a breadth-first search traversal of a given graph
//Input: Graph G = ⟨V, E⟩
//Output: Graph G with its vertices marked with consecutive integers
//in the order they have been visited by the BFS traversal
mark each vertex in V with 0 as a mark of being "unvisited"
count ← 0
for each vertex v in V do
    if v is marked with 0
        bfs(v)

bfs(v)
//visits all the unvisited vertices connected to vertex v by a path
//and assigns them the numbers in the order they are visited
//via global variable count
count ← count + 1;   mark v with count and initialize a queue with v
while the queue is not empty do
    for each vertex w in V adjacent to the front vertex do
        if w is marked with 0
            count ← count + 1;   mark w with count
            add w to the queue
    remove the front vertex from the queue
```

BFS example

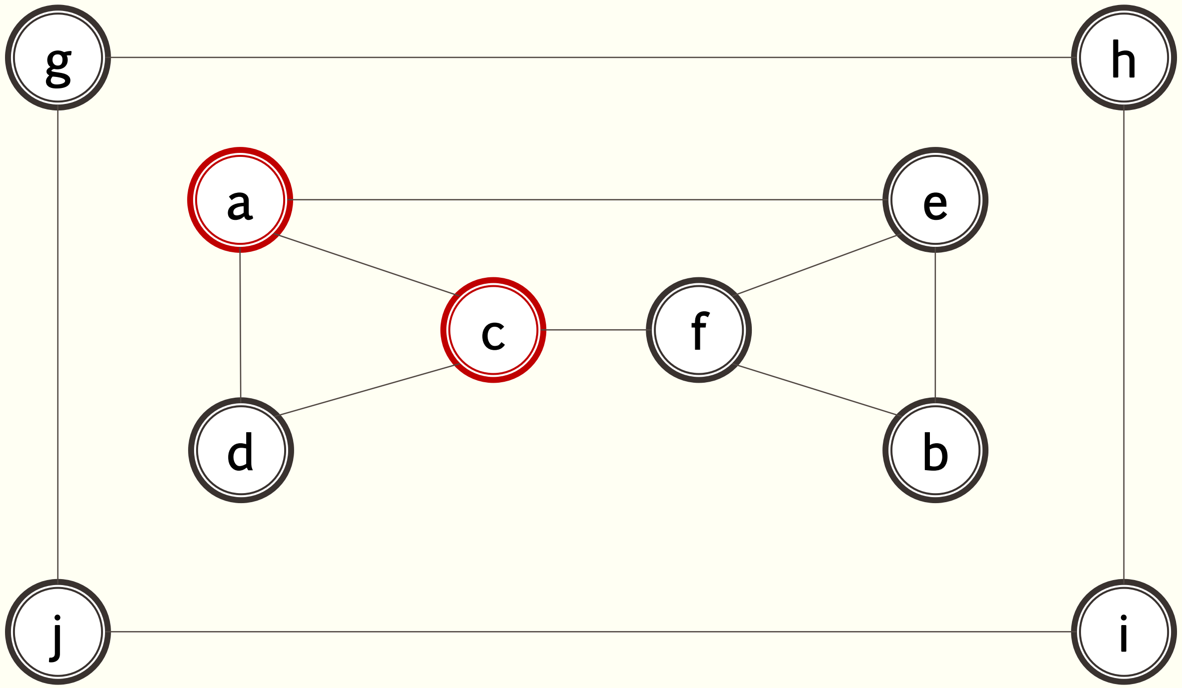


BFS example



Into queue order
a

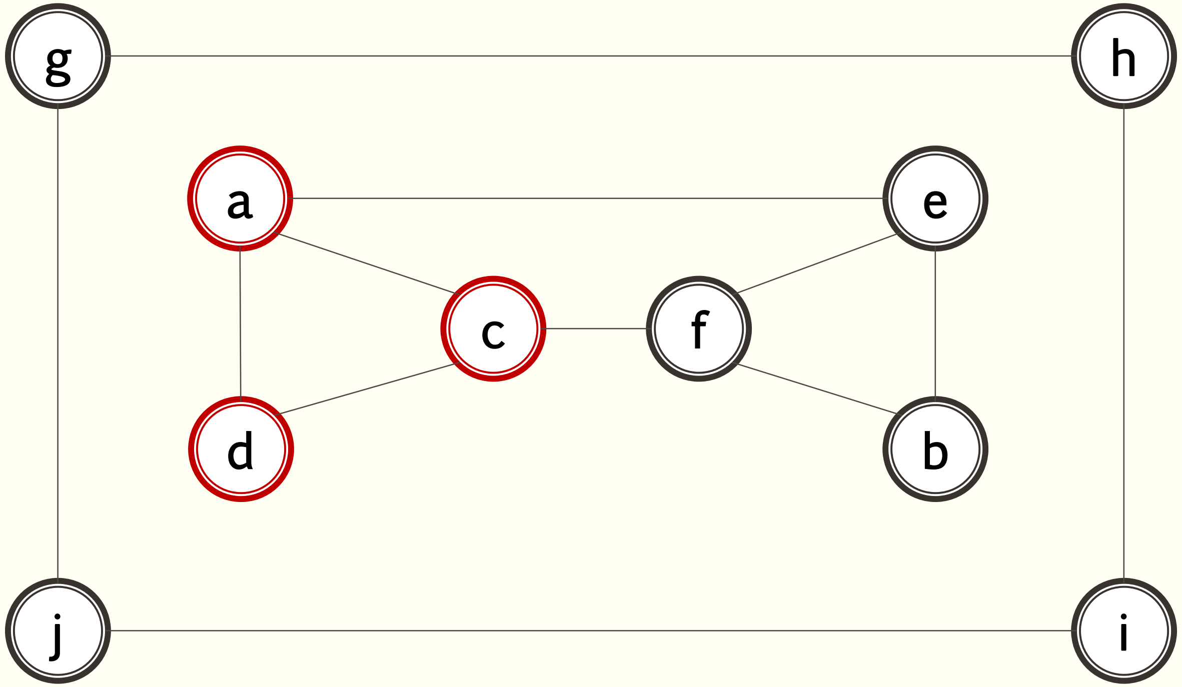
BFS example



a	c

Into queue order
a
c

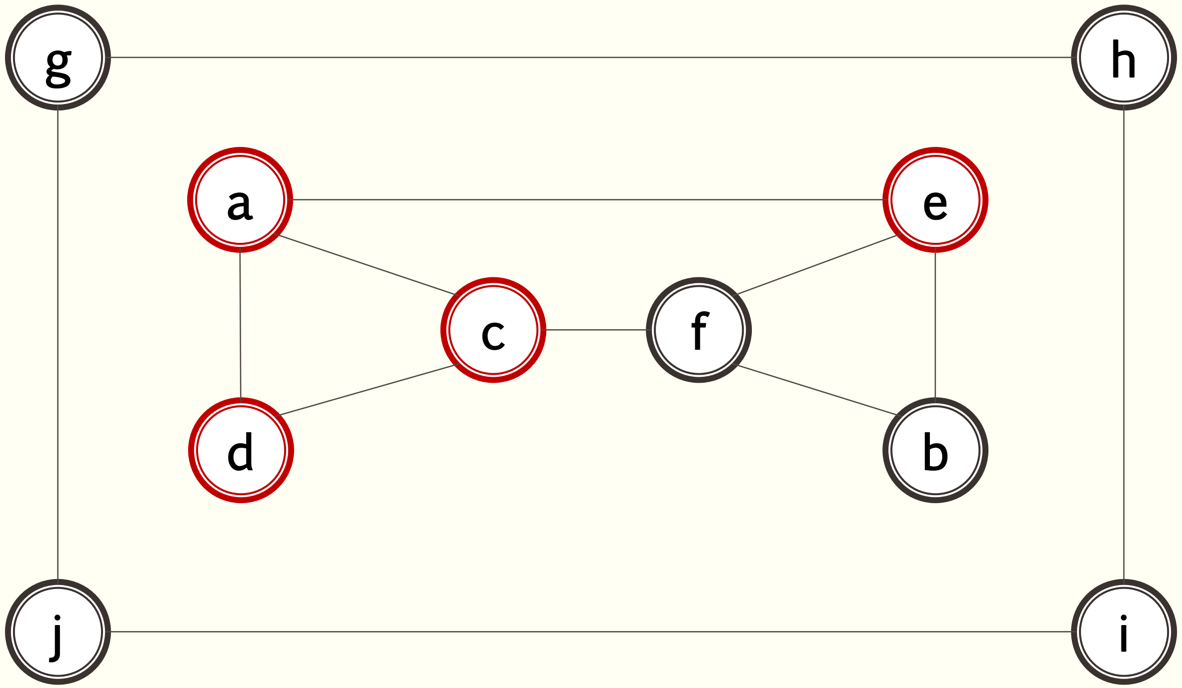
BFS example



Into queue order
a
c
d

a	c	d

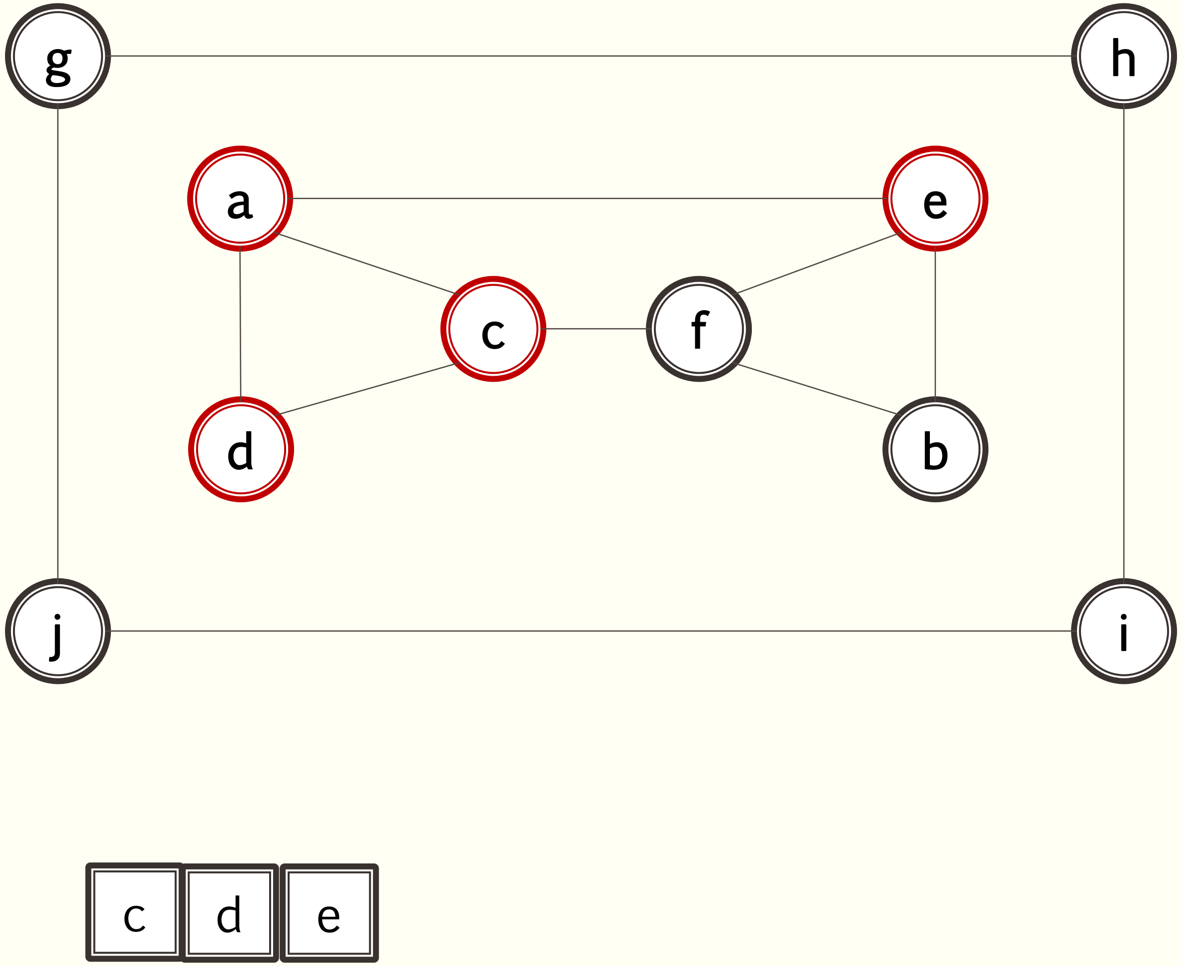
BFS example



a	c	d	e

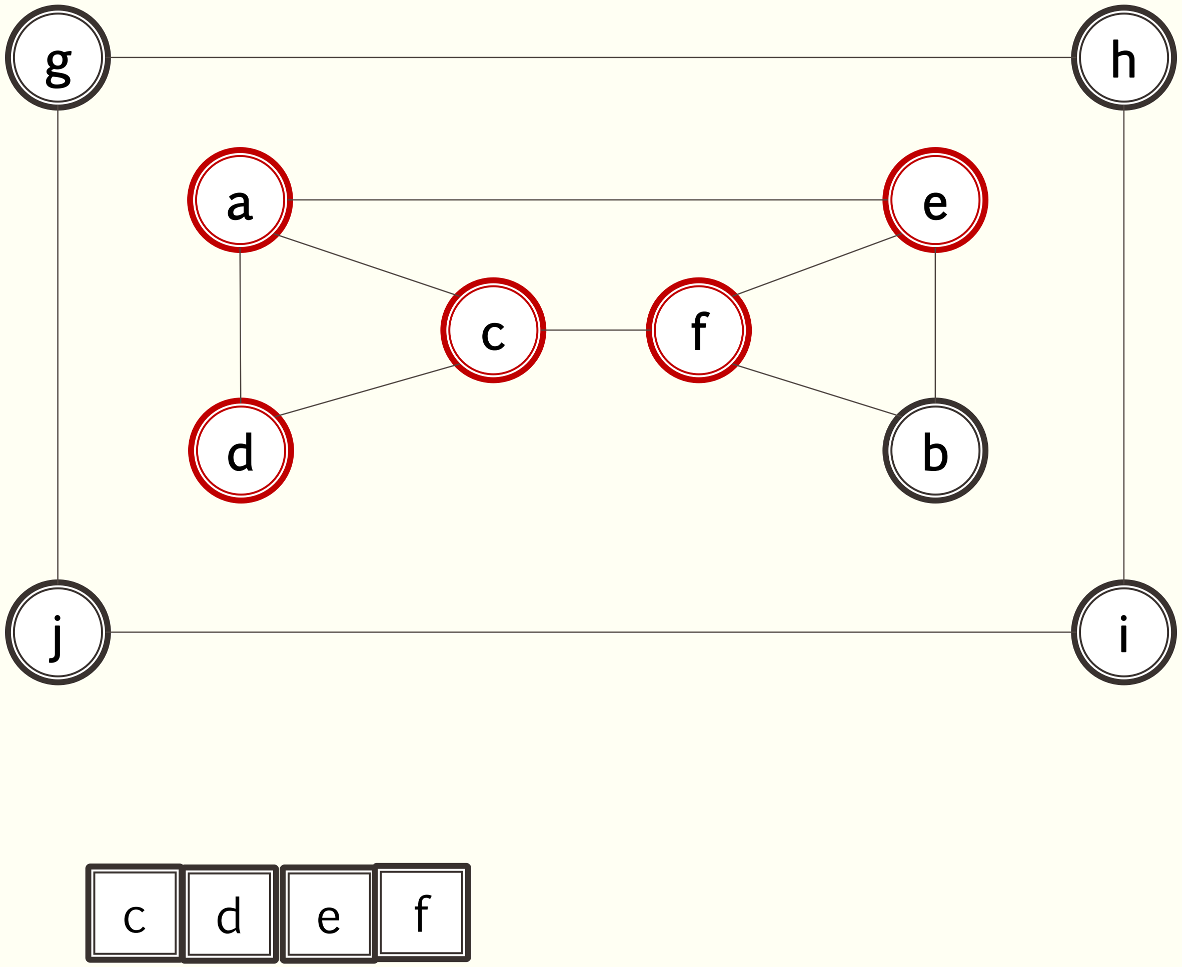
Into queue order
a
c
d
e

BFS example



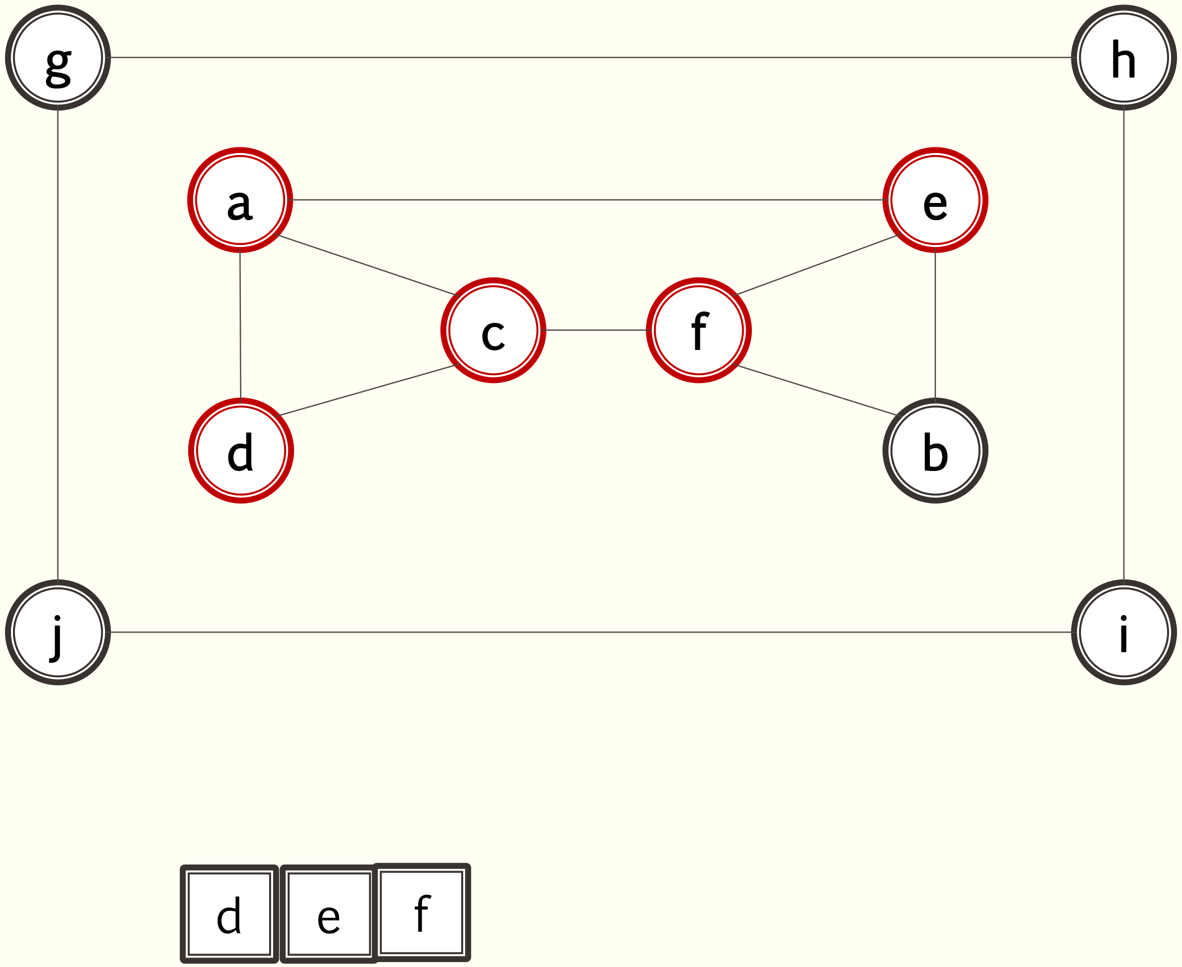
Into queue order
a
c
d
e

BFS example



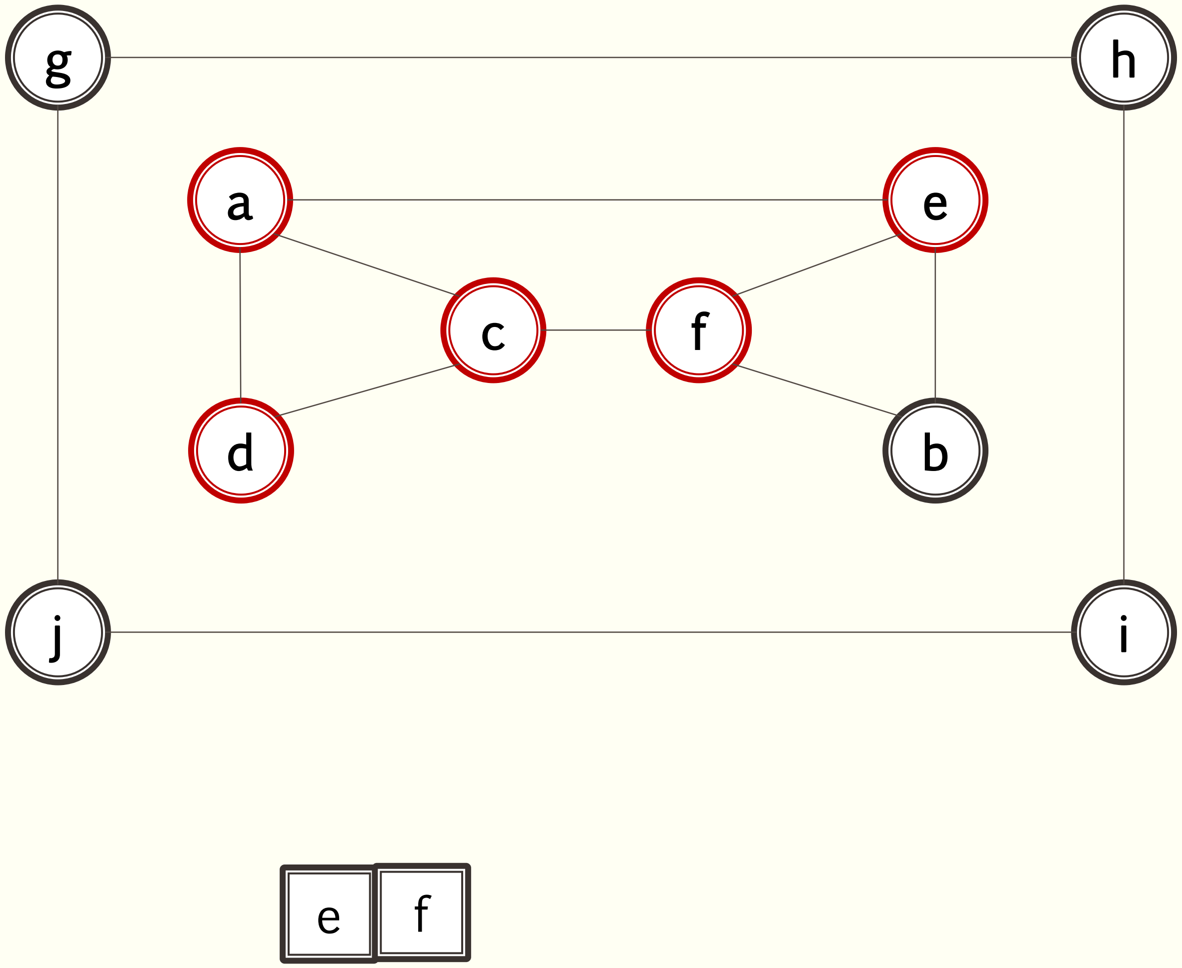
Into queue order
a
c
d
e
f

BFS example



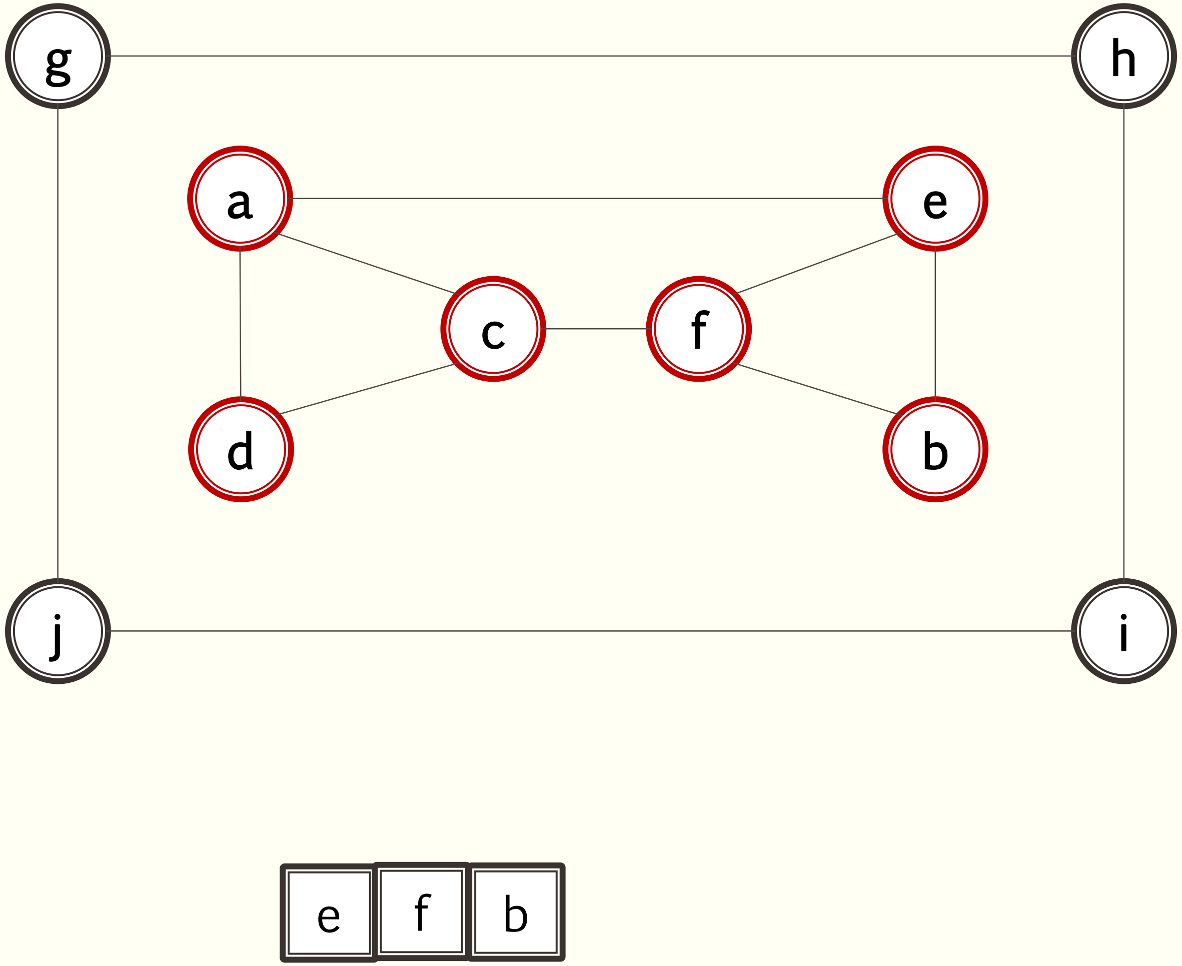
Into queue order
a
c
d
e
f

BFS example



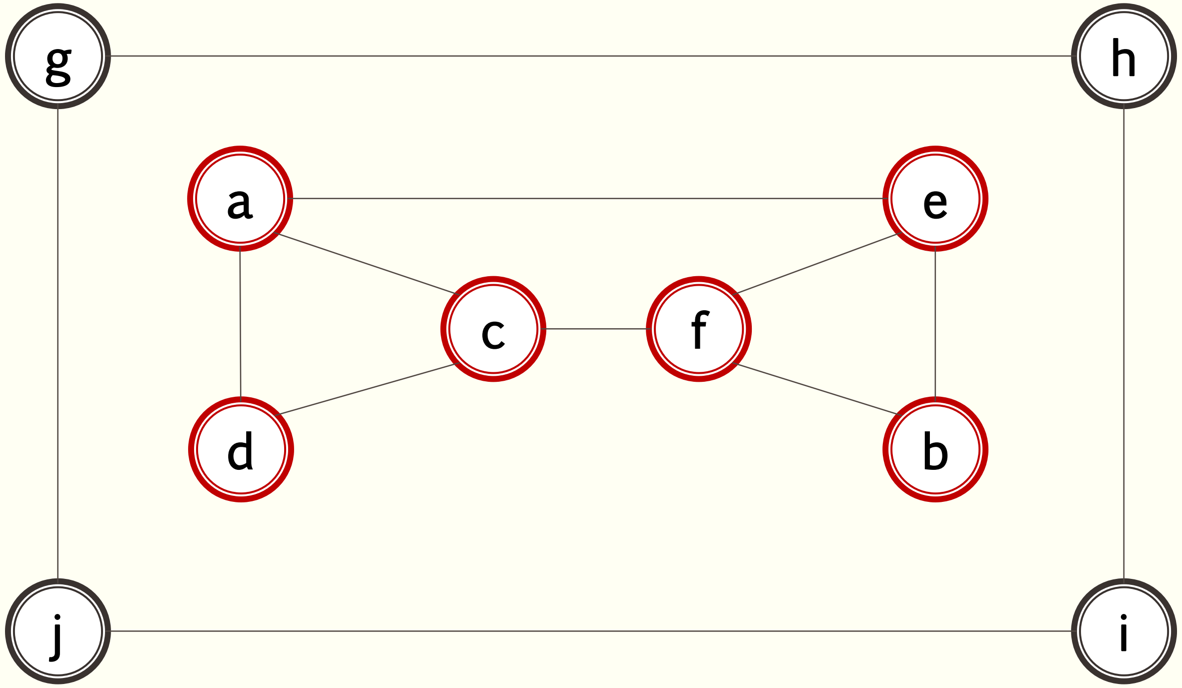
Into queue order
a
c
d
e
f

BFS example



Into queue order
a
c
d
e
f
b

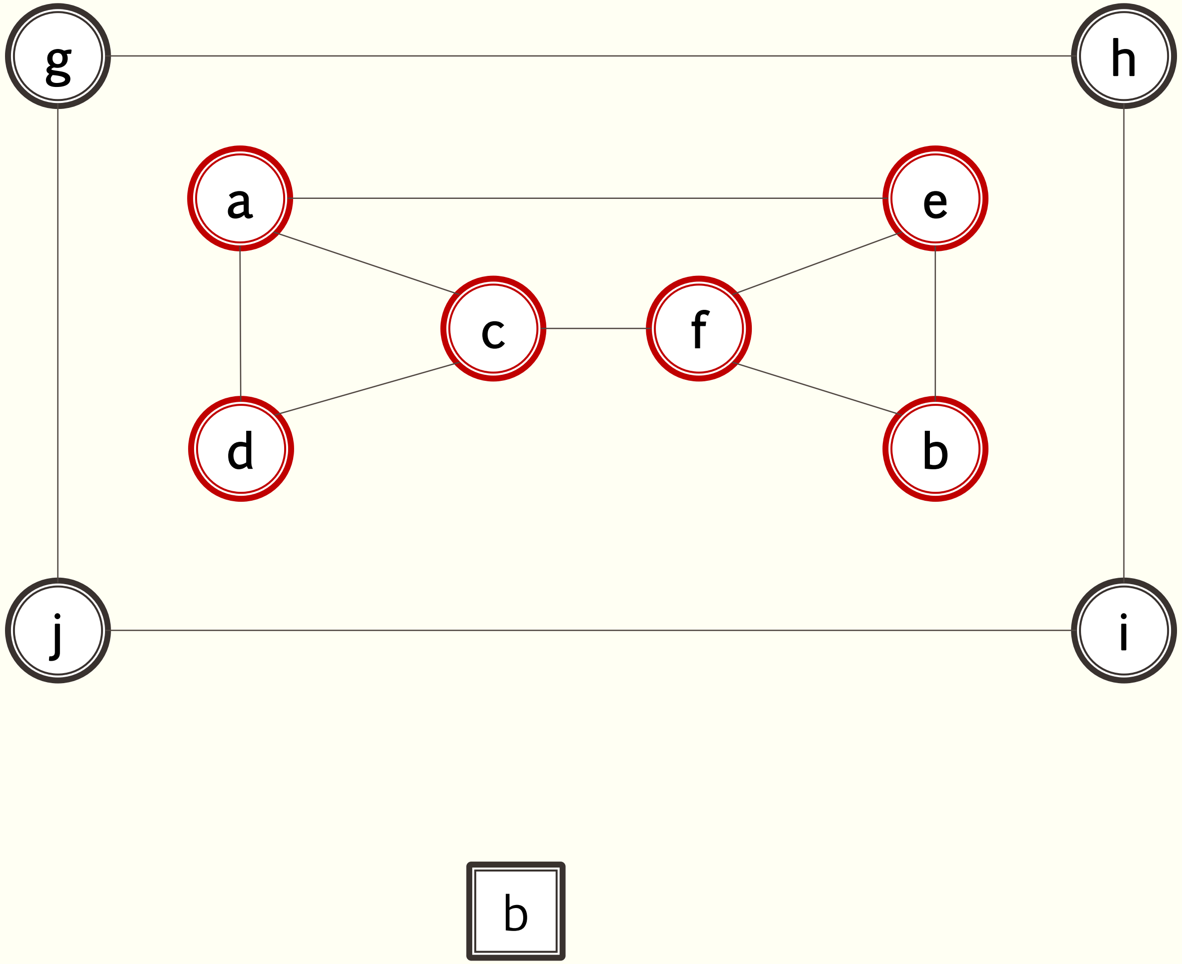
BFS example



f	b

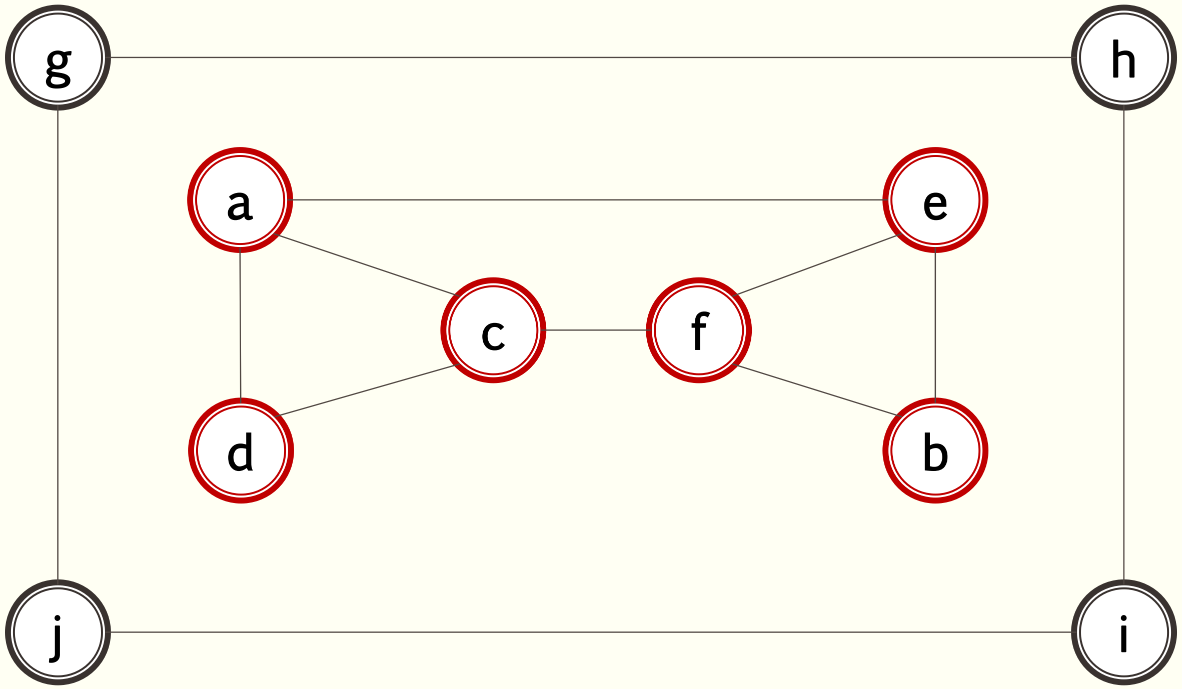
Into queue order
a
c
d
e
f
b

BFS example



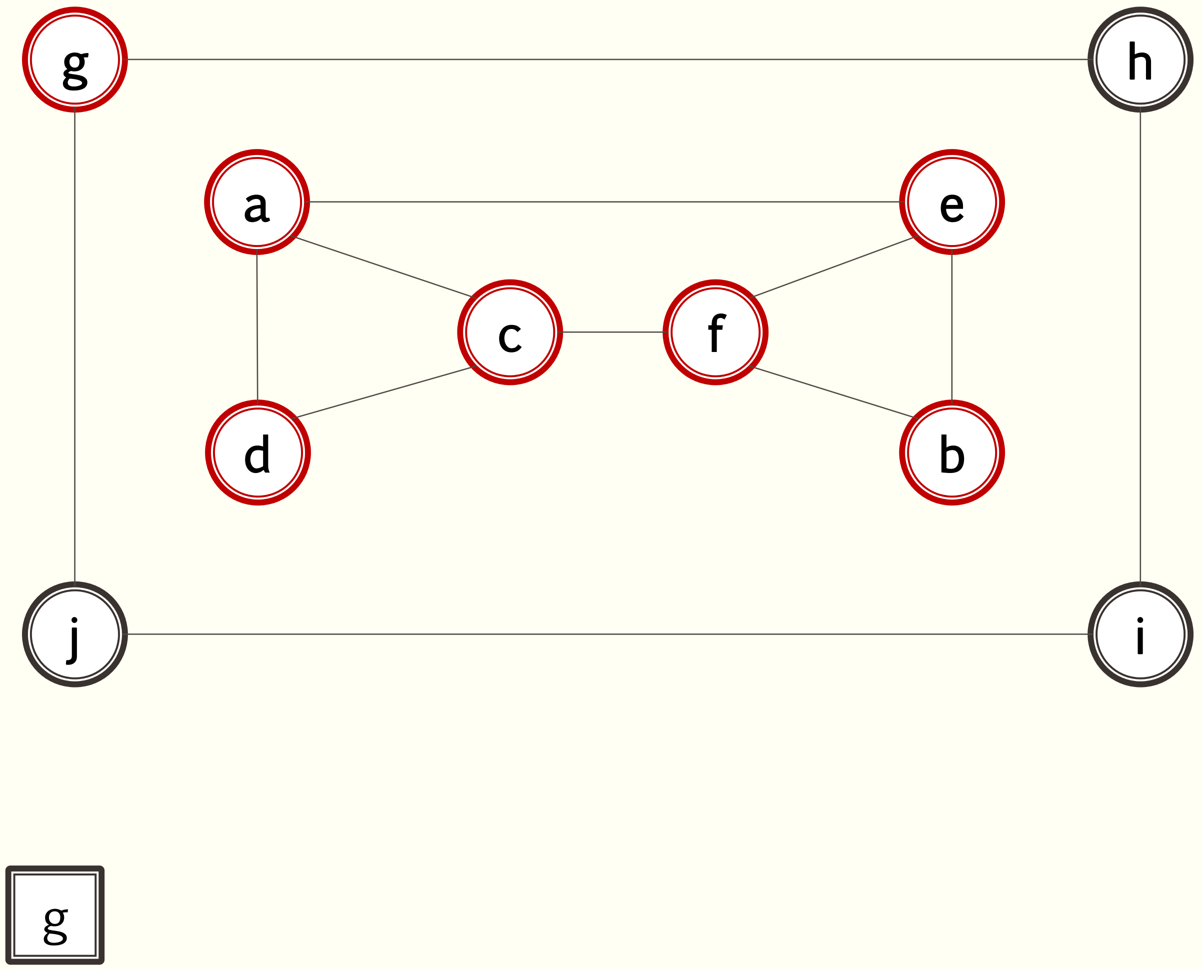
Into queue order
a
c
d
e
f
b

BFS example



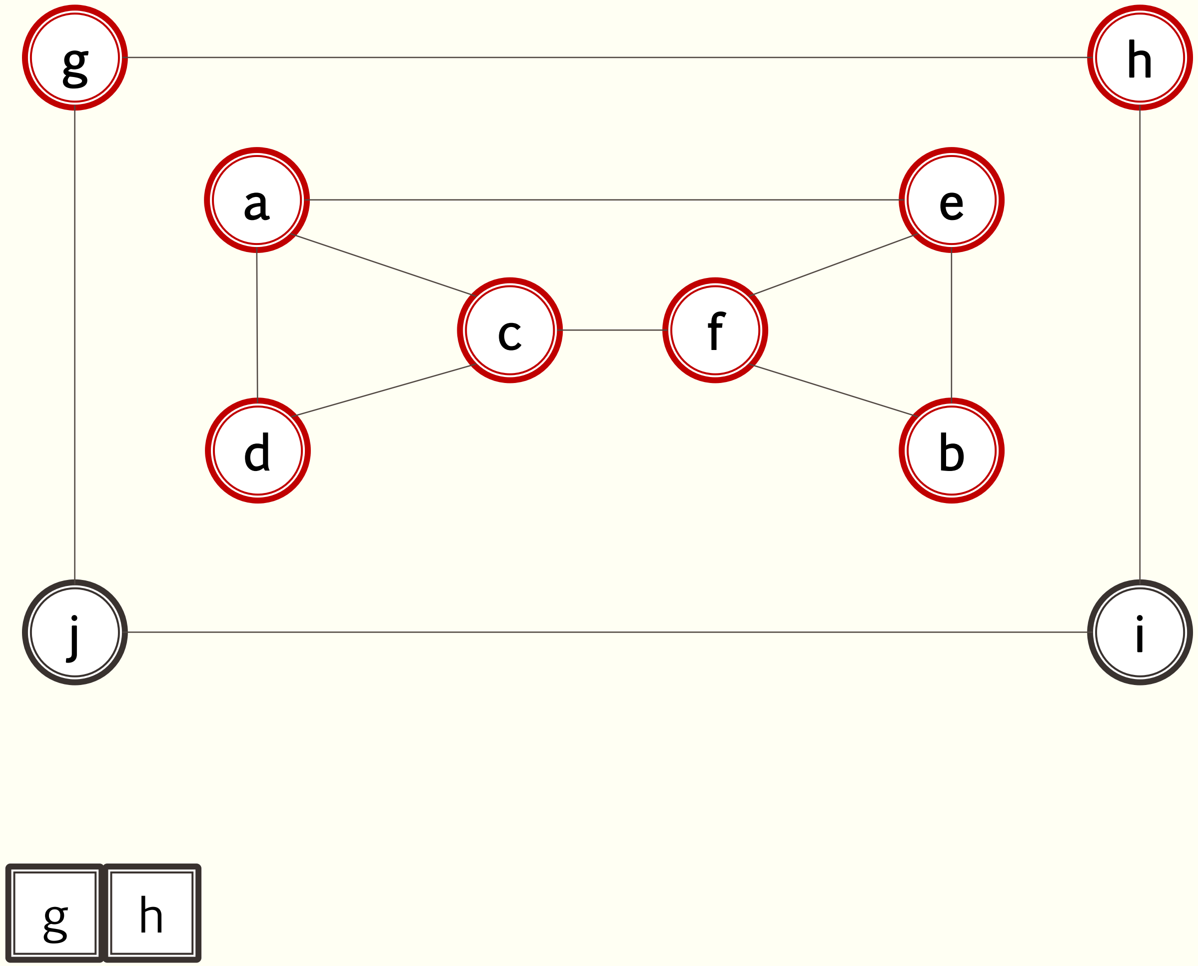
Into queue order
a
c
d
e
f
b

BFS example



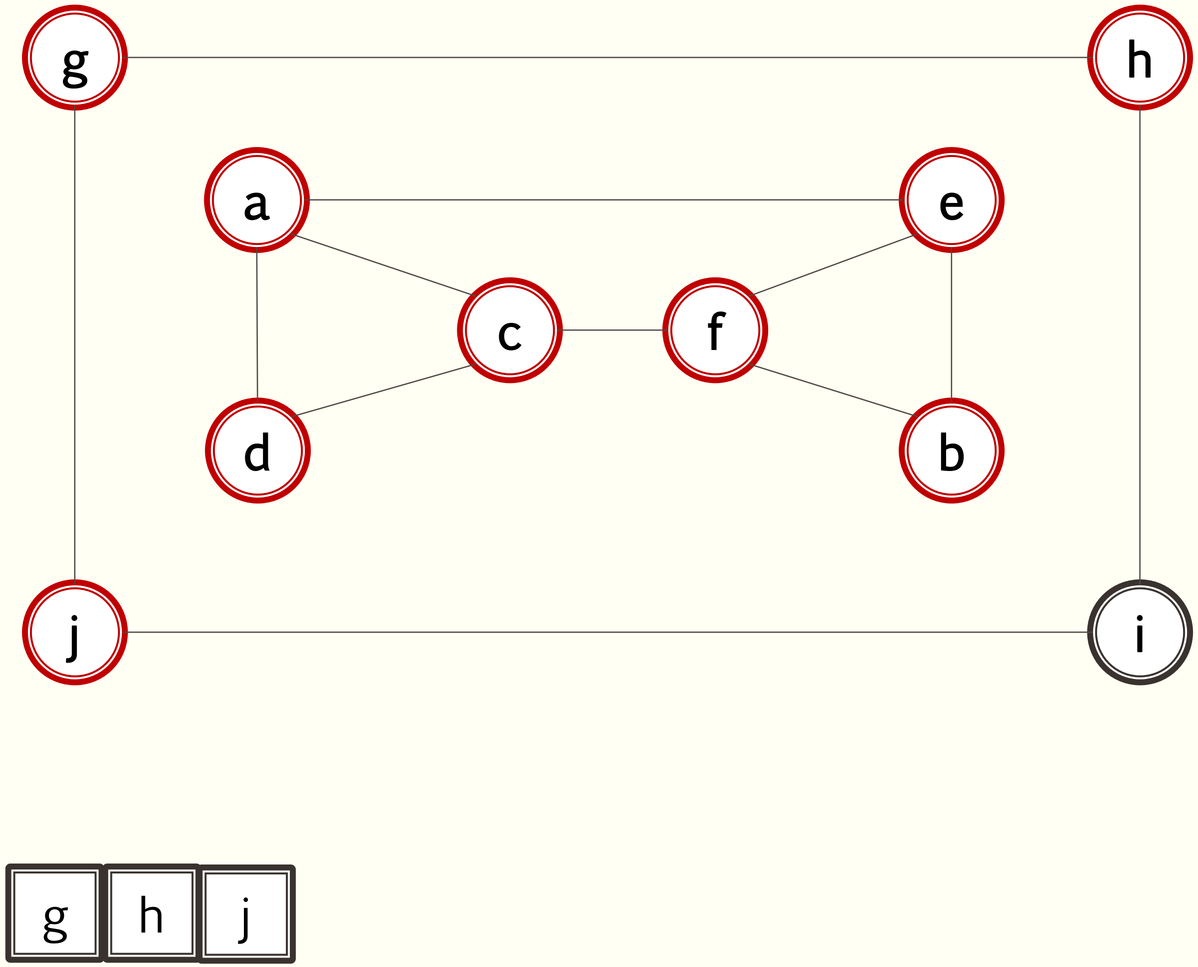
Into queue order
a
c
d
e
f
b
g

BFS example



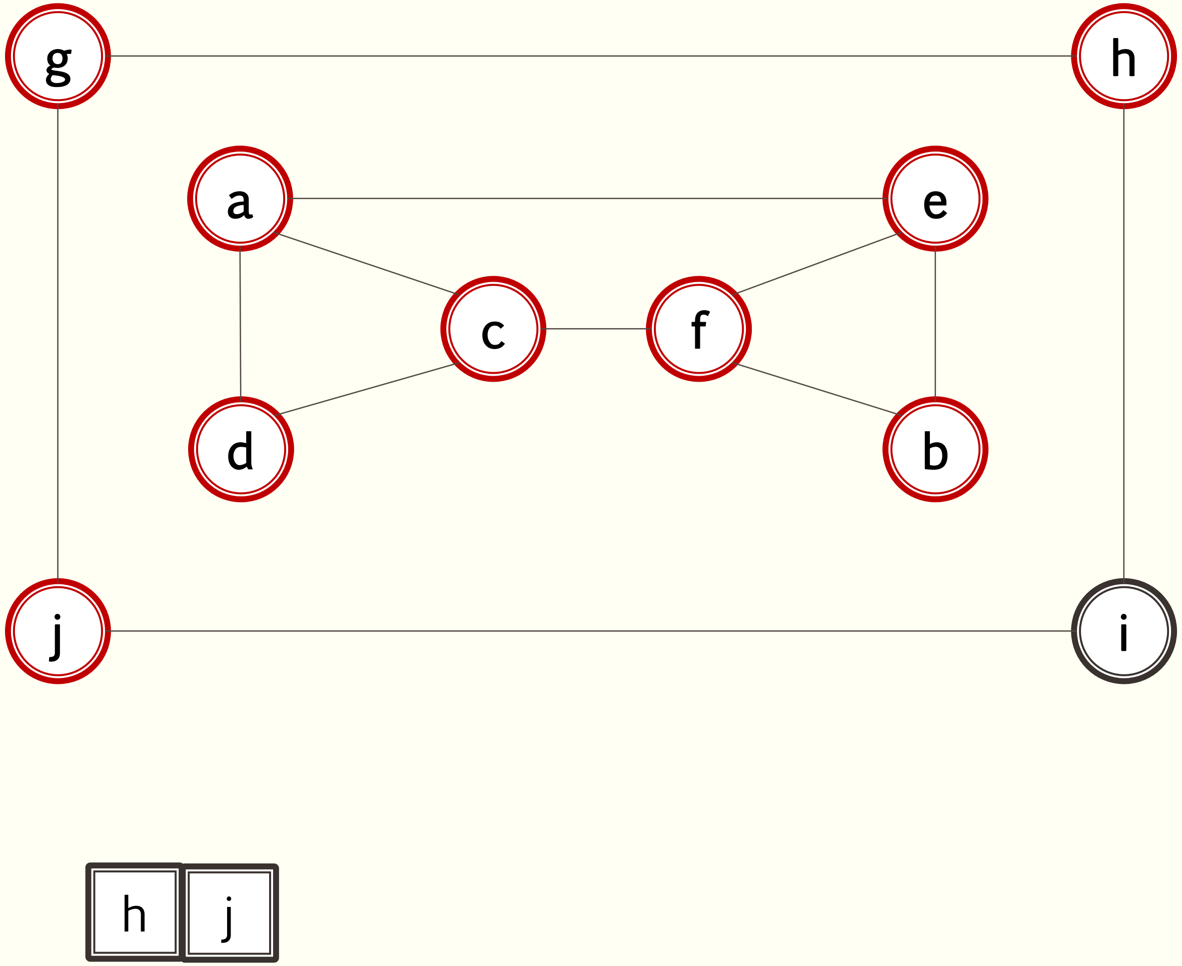
Into queue order
a
c
d
e
f
b
g
h

BFS example



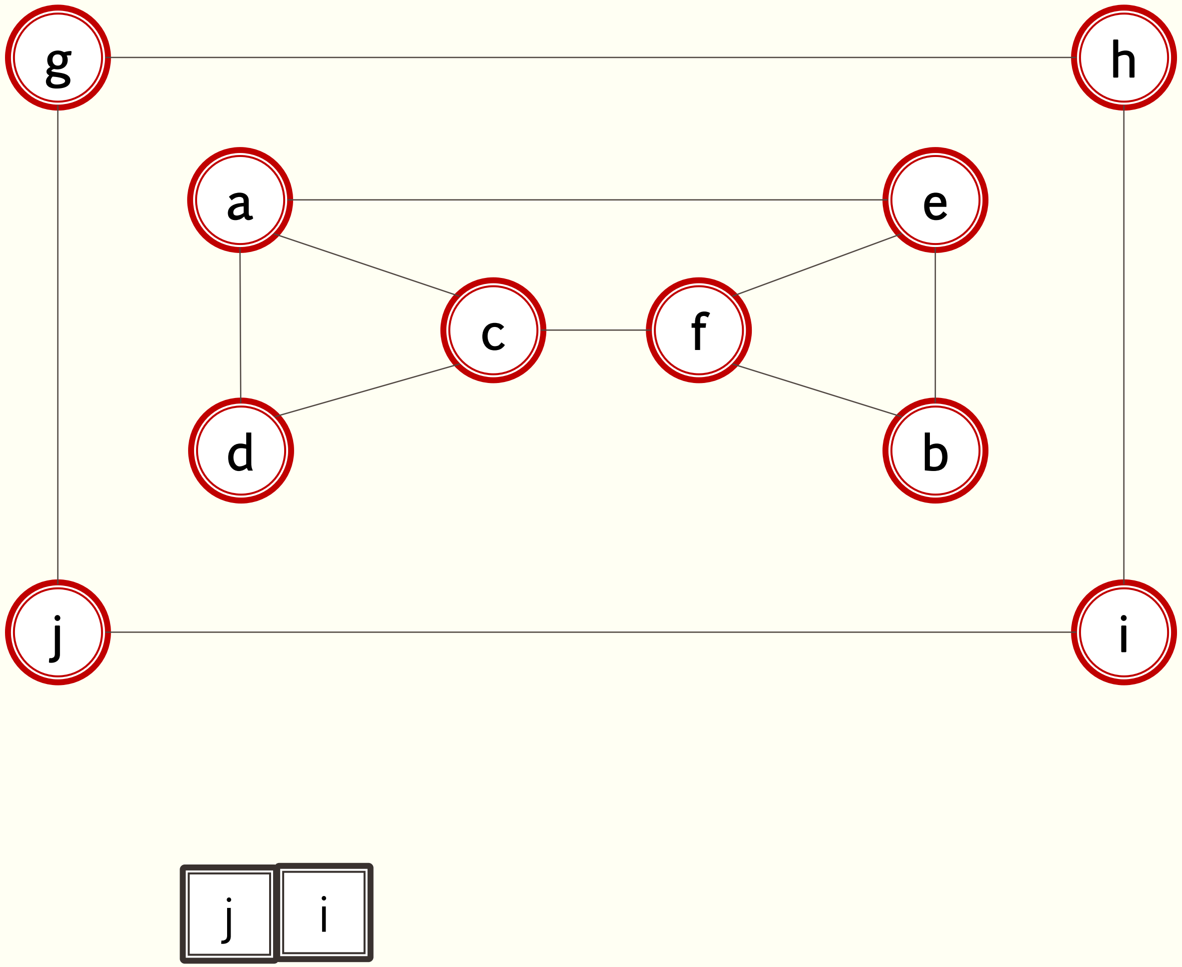
Into queue order
a
c
d
e
f
b
g
h
j

BFS example



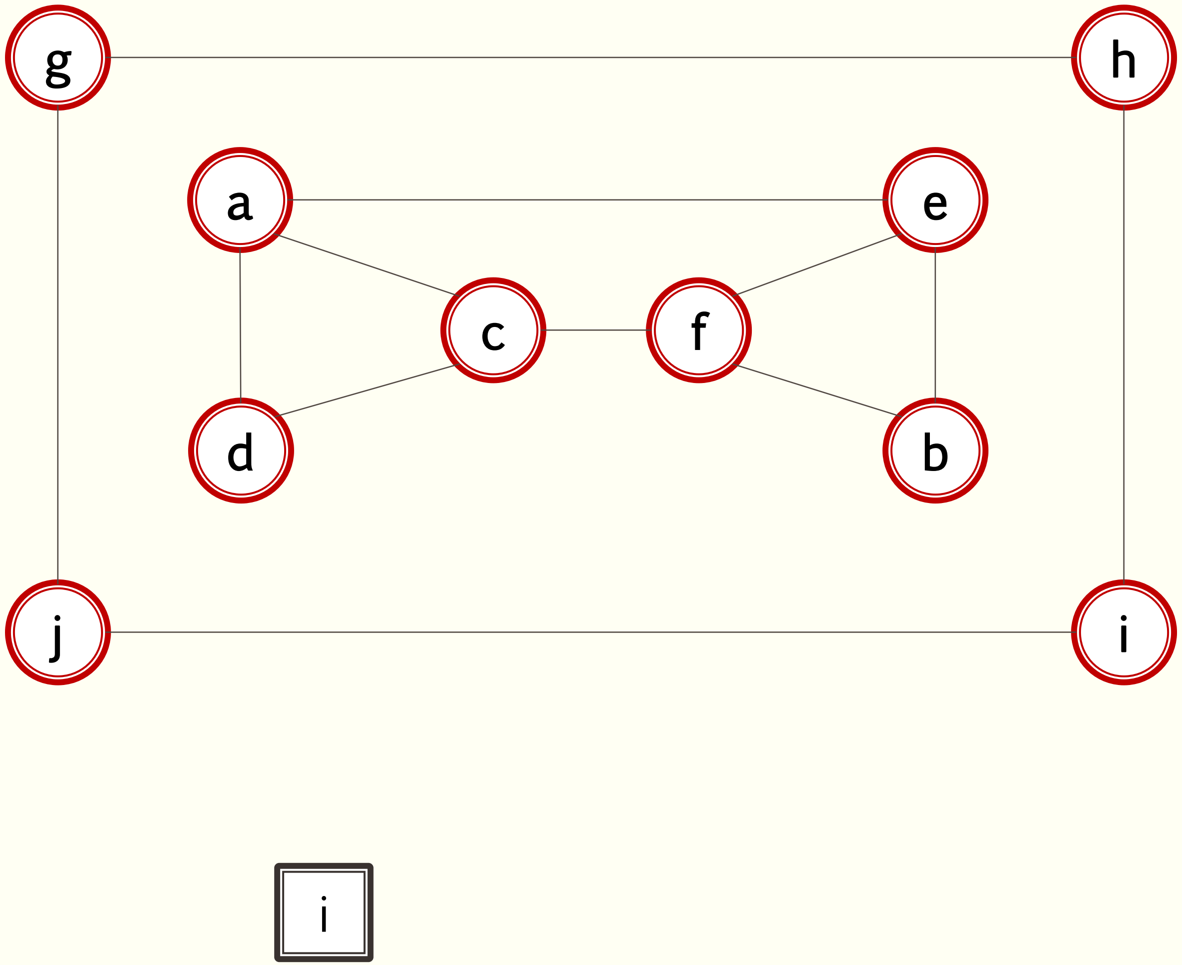
Into queue order
a
c
d
e
f
b
g
h
j

BFS example



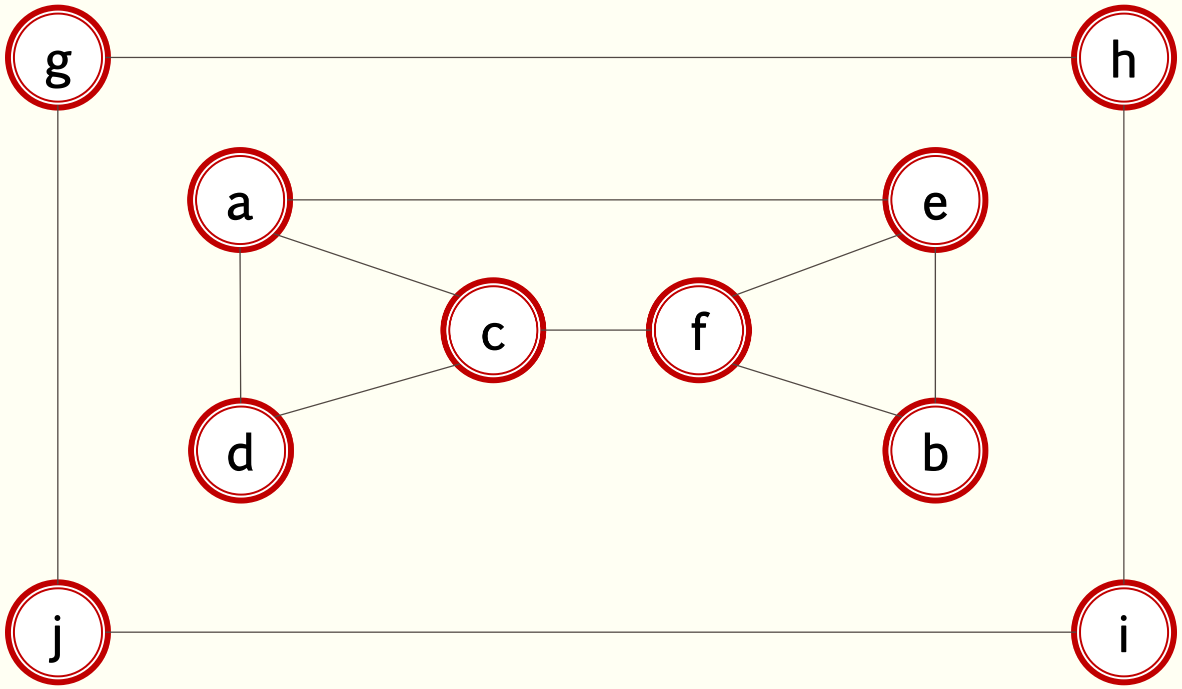
Into queue order
a
c
d
e
f
b
g
h
j
i

BFS example



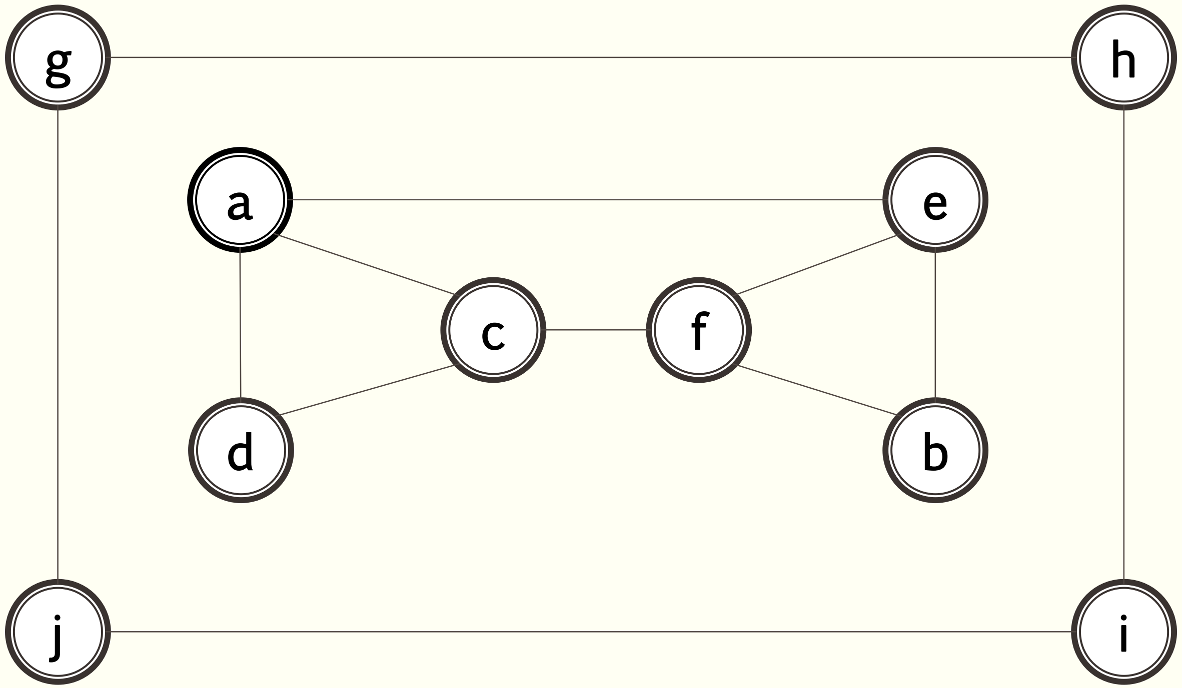
Into queue order
a
c
d
e
f
b
g
h
j
i

BFS example

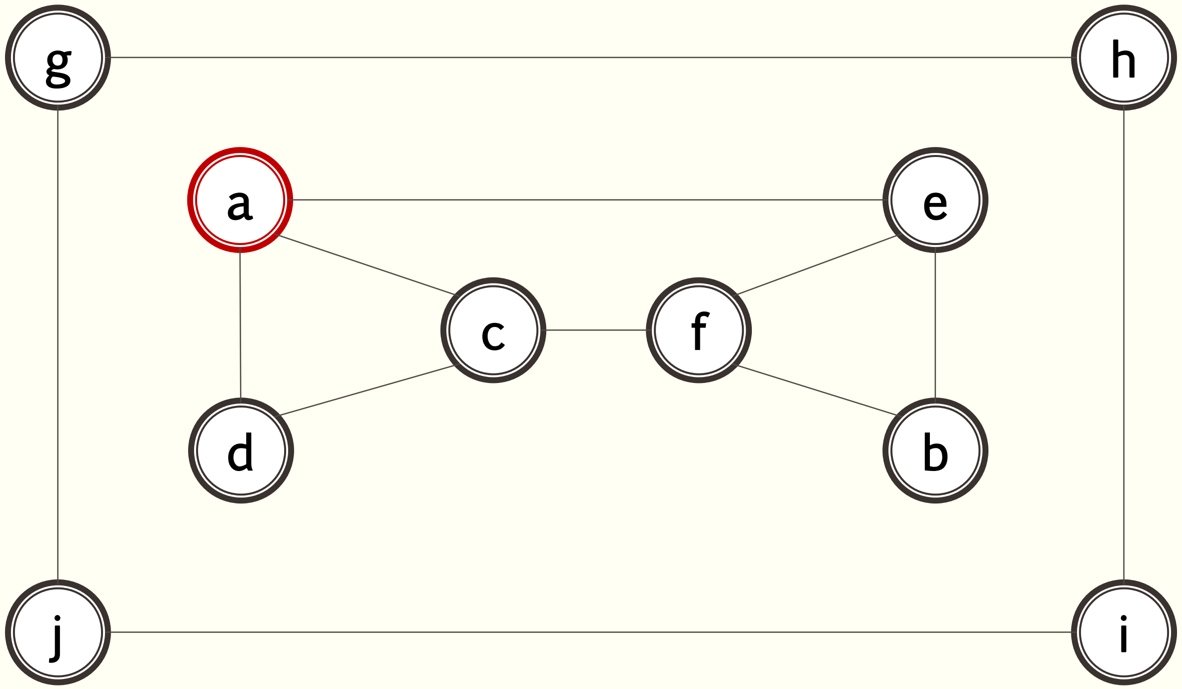


Into queue order
a
c
d
e
f
b
g
h
j
i

DFS for same graph

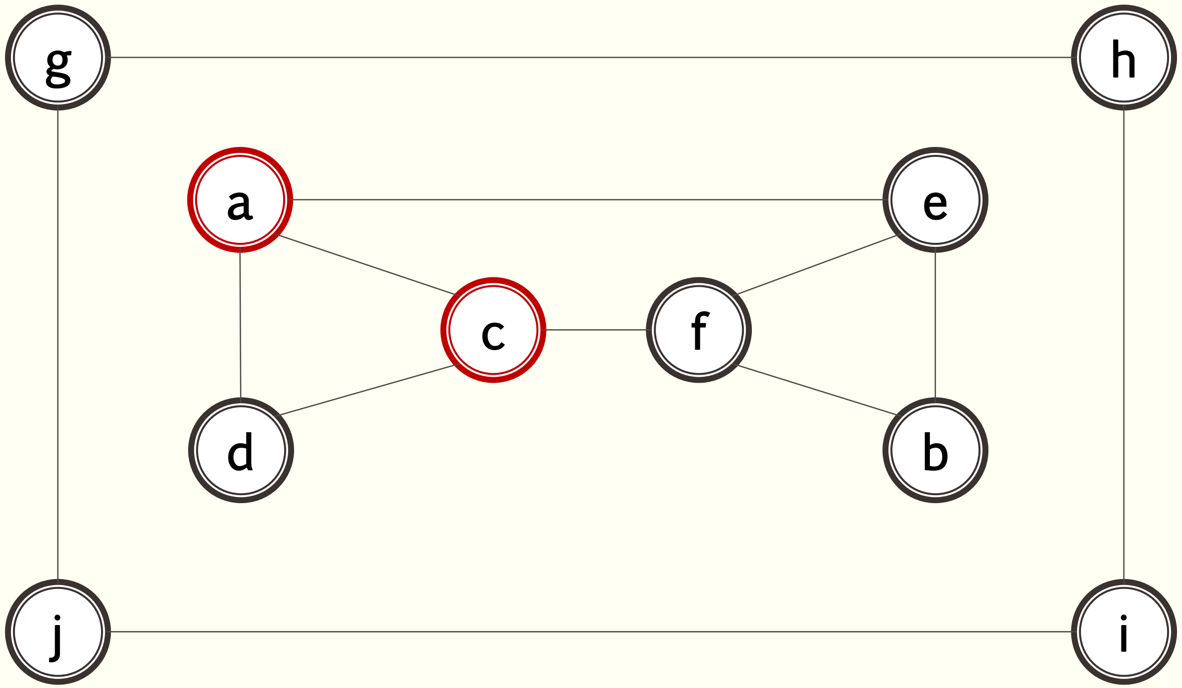


DFS for same graph



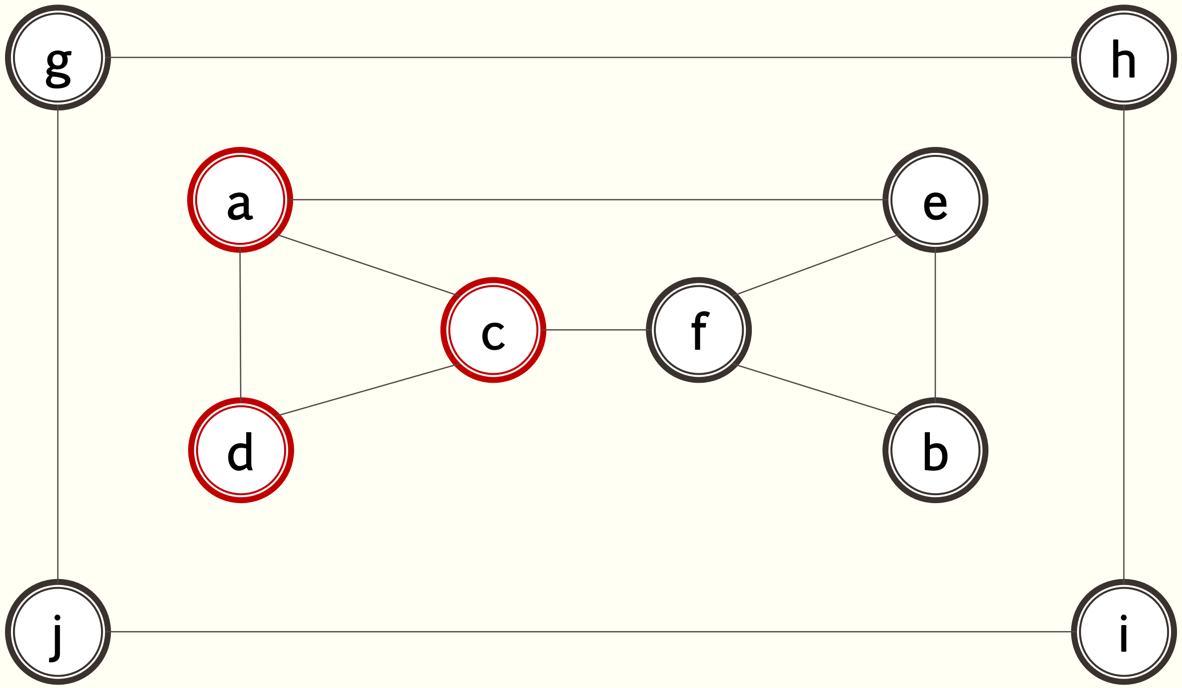
Into stack order	out of stack order
a	

DFS for same graph



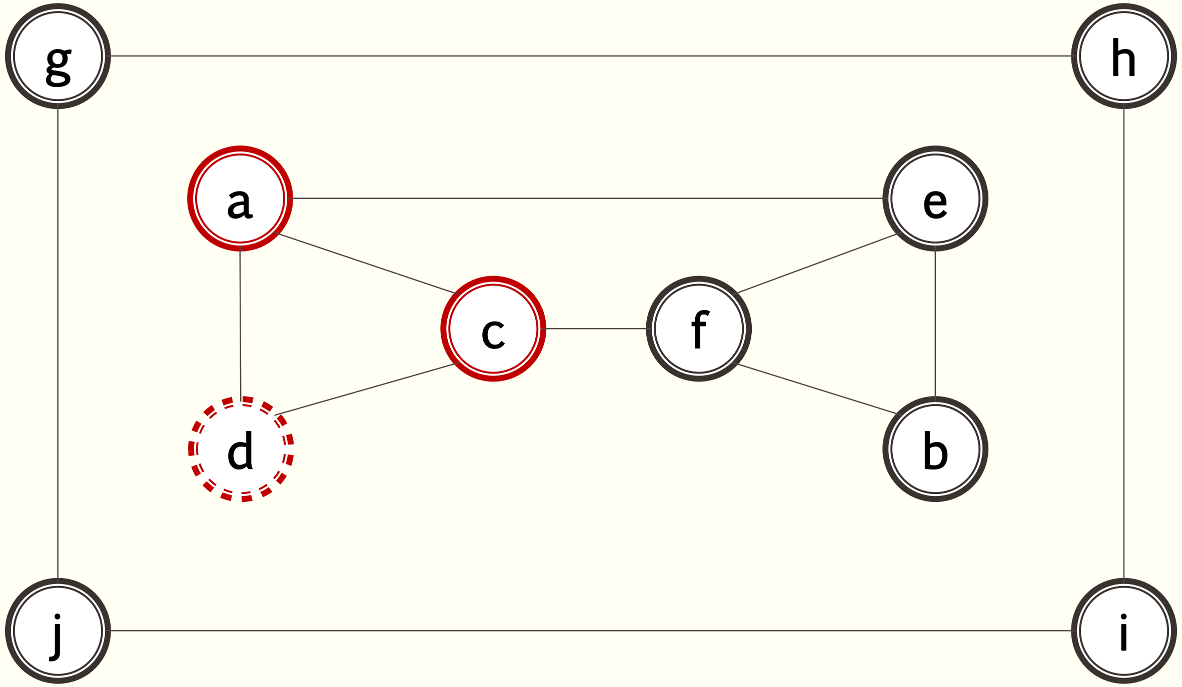
Into stack order	out of stack order
a	
c	

DFS for same graph



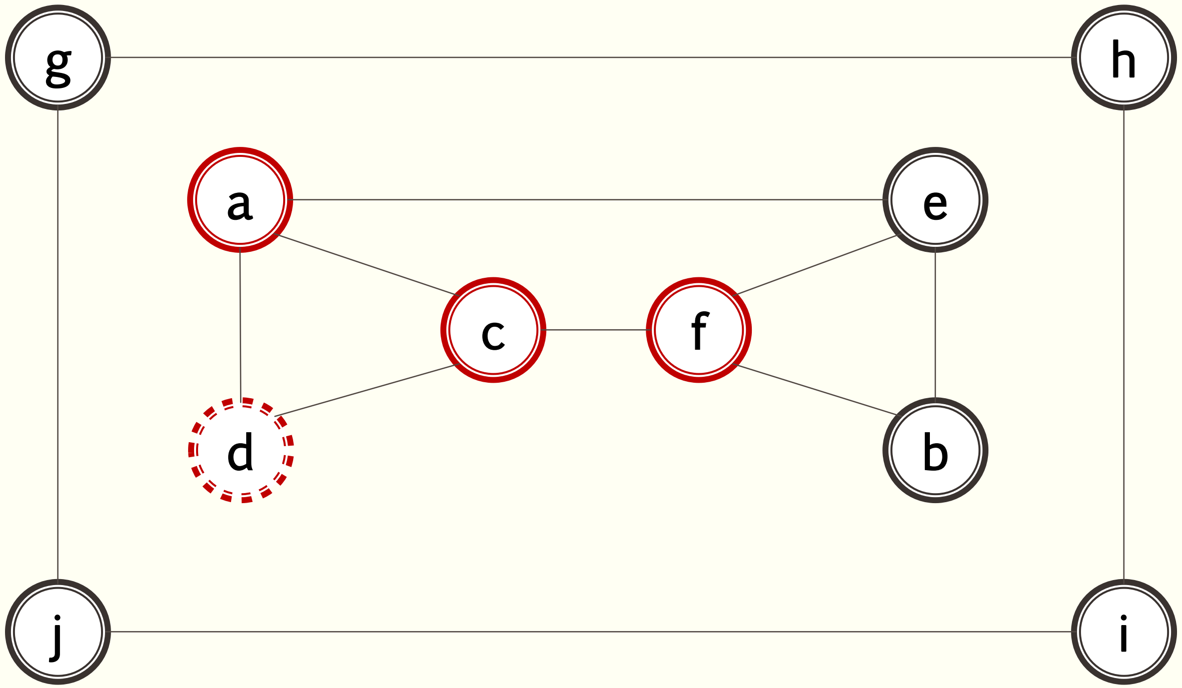
Into stack order	out of stack order
a	
c	
d	

DFS for same graph



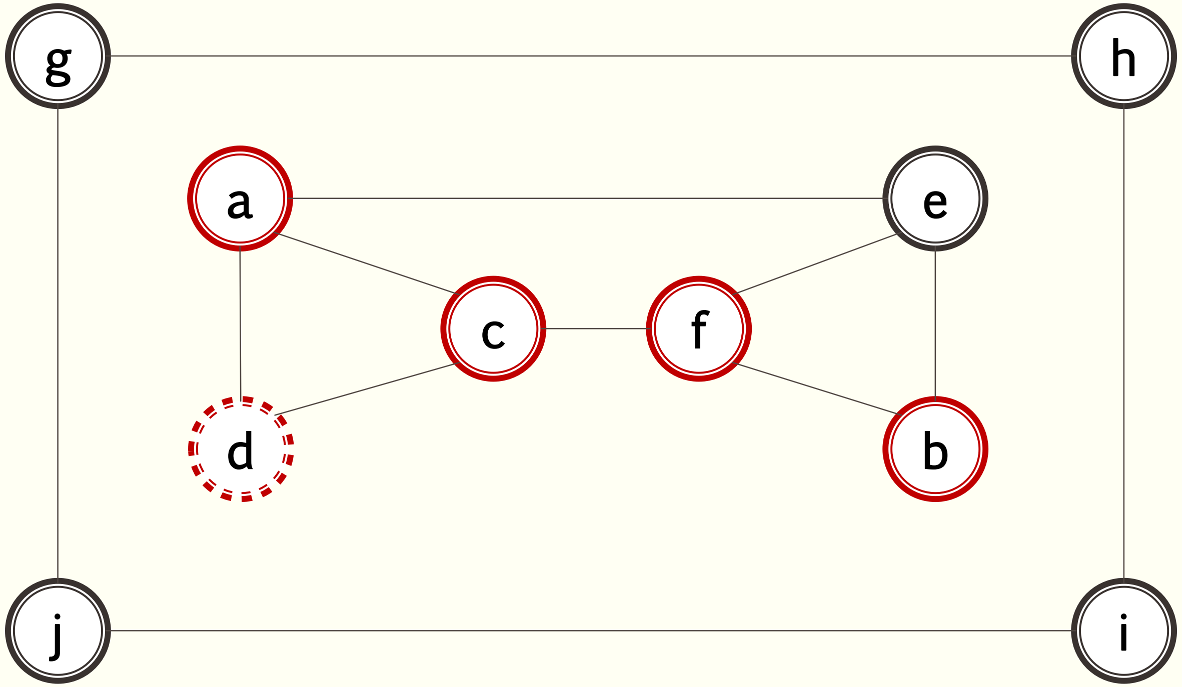
Into stack order	out of stack order
a	d
c	
d	

DFS for same graph



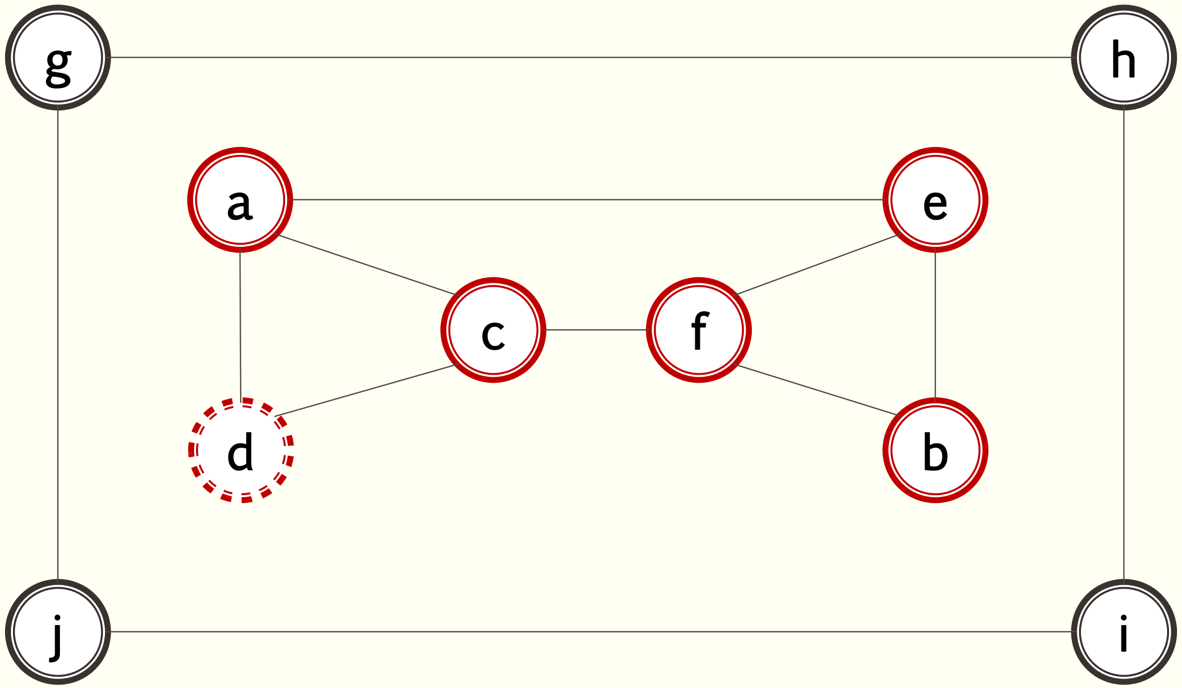
Into stack order	out of stack order
a	d
c	
d	
f	

DFS for same graph



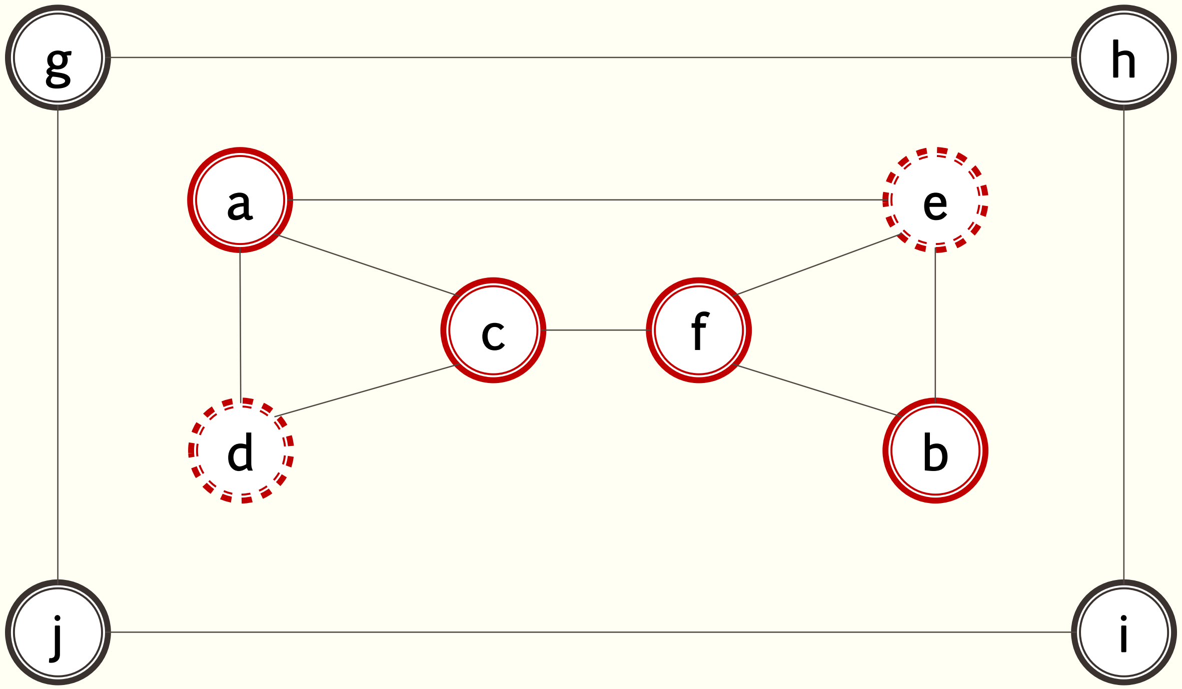
Into stack order	out of stack order
a	d
c	
d	
f	
b	

DFS for same graph



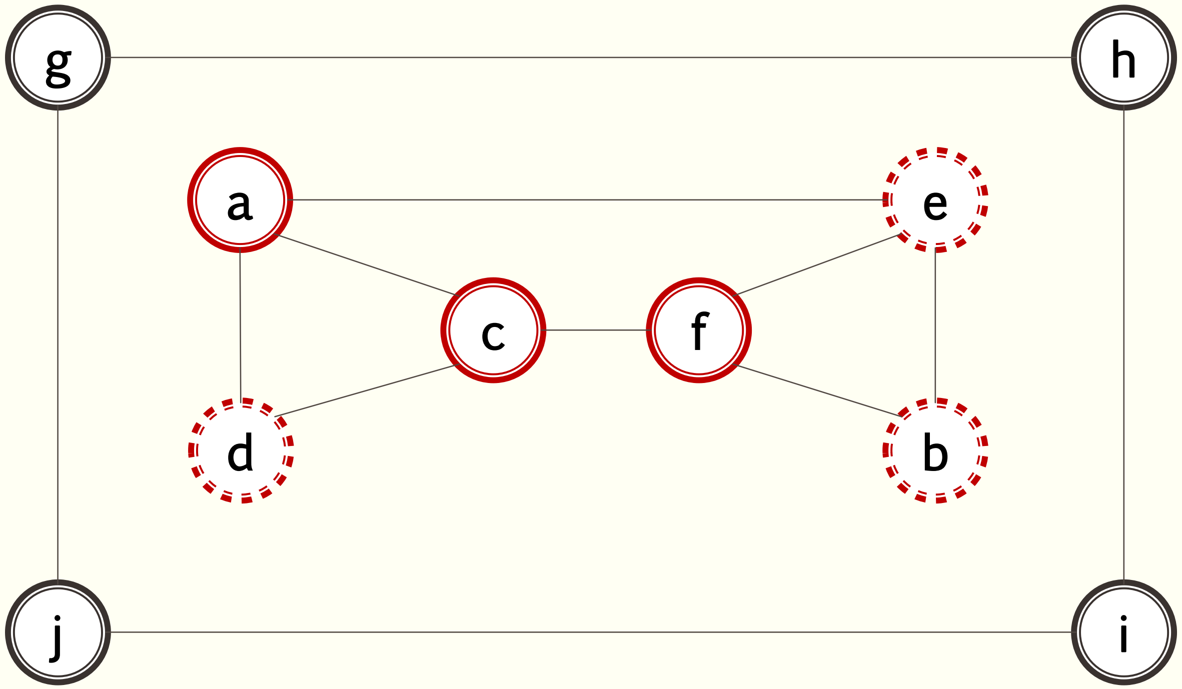
Into stack order	out of stack order
a	d
c	
d	
f	
b	
e	

DFS for same graph



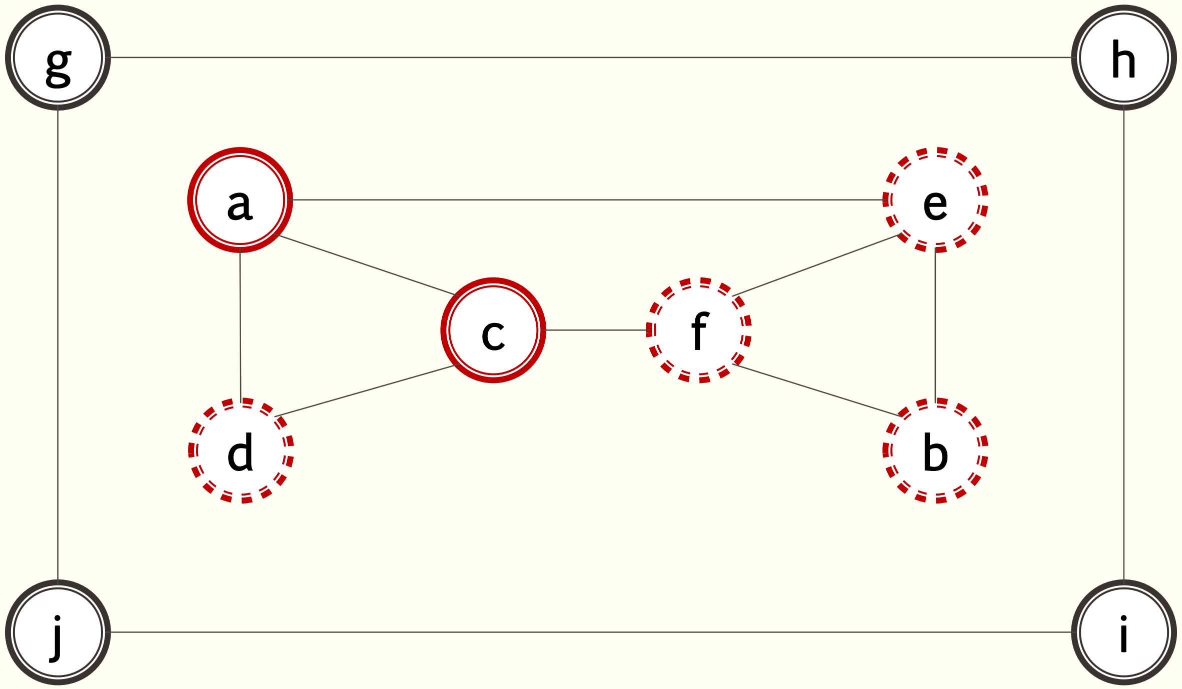
Into stack order	out of stack order
a	d
c	e
d	
f	
b	
e	

DFS for same graph



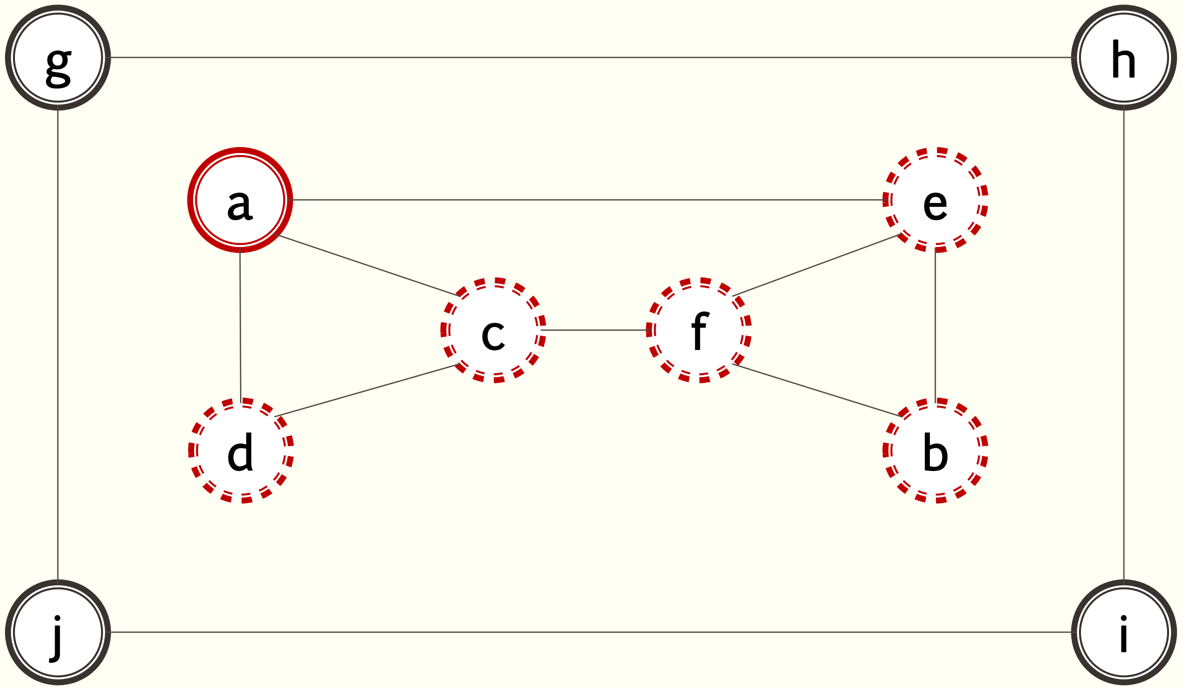
Into stack order	out of stack order
a	d
c	e
d	b
f	
b	
e	

DFS for same graph



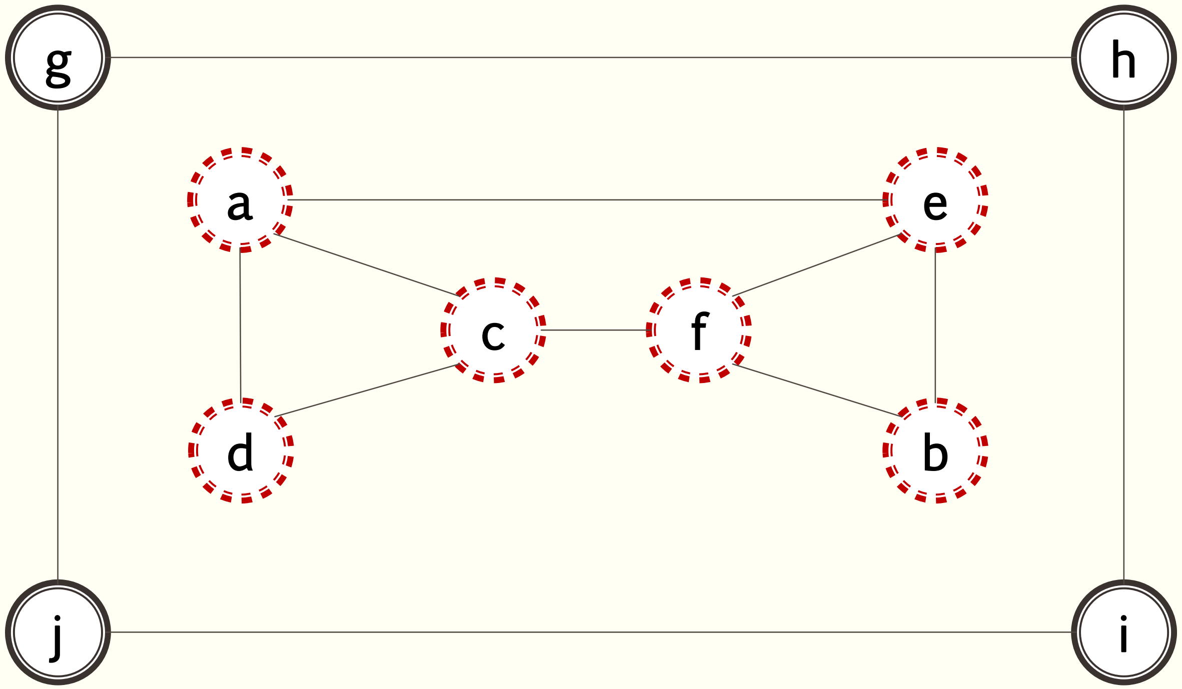
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	
e	

DFS for same graph



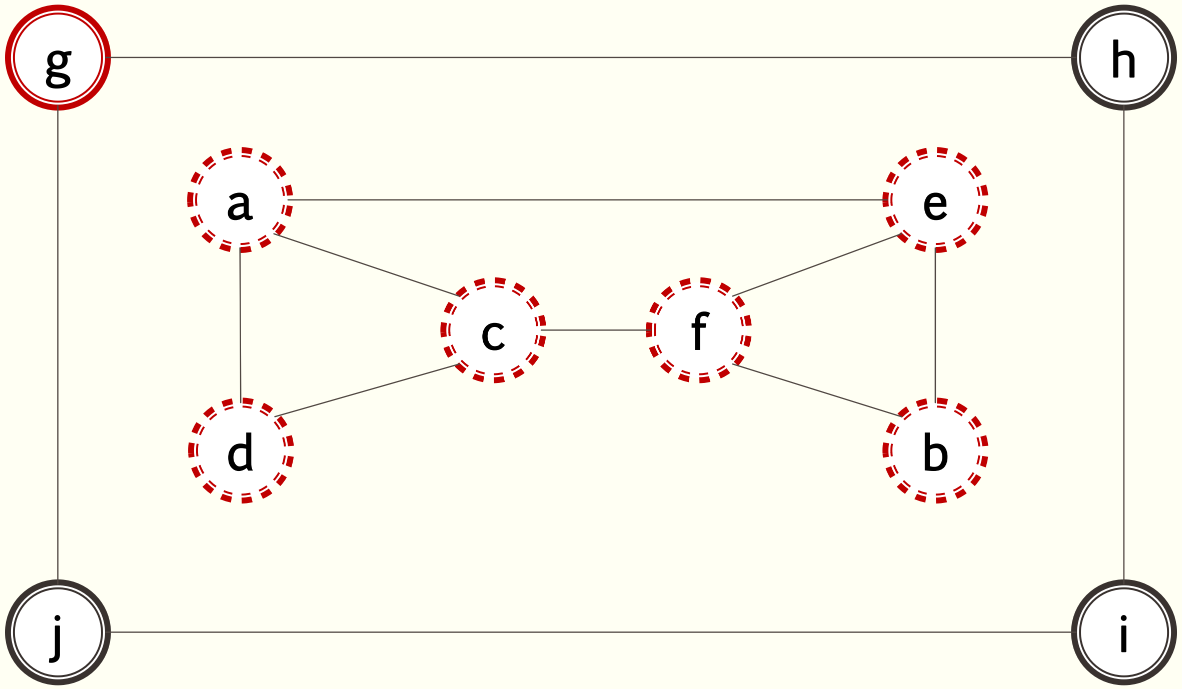
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	

DFS for same graph



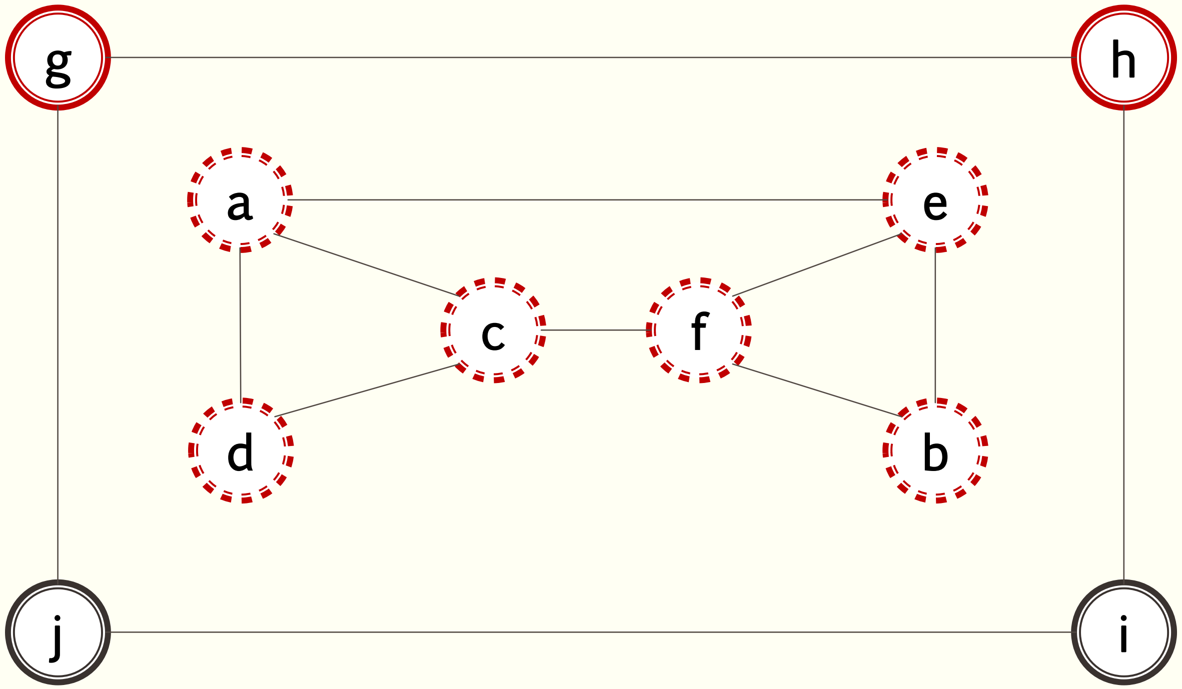
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a

DFS for same graph



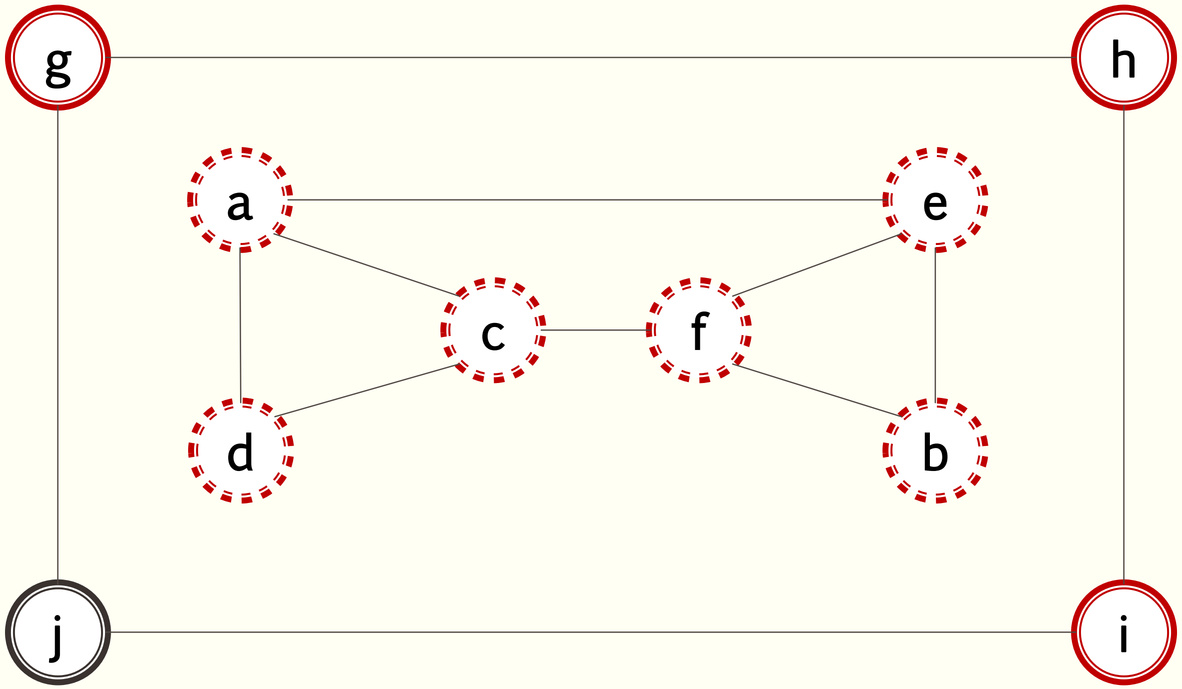
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a
g	

DFS for same graph



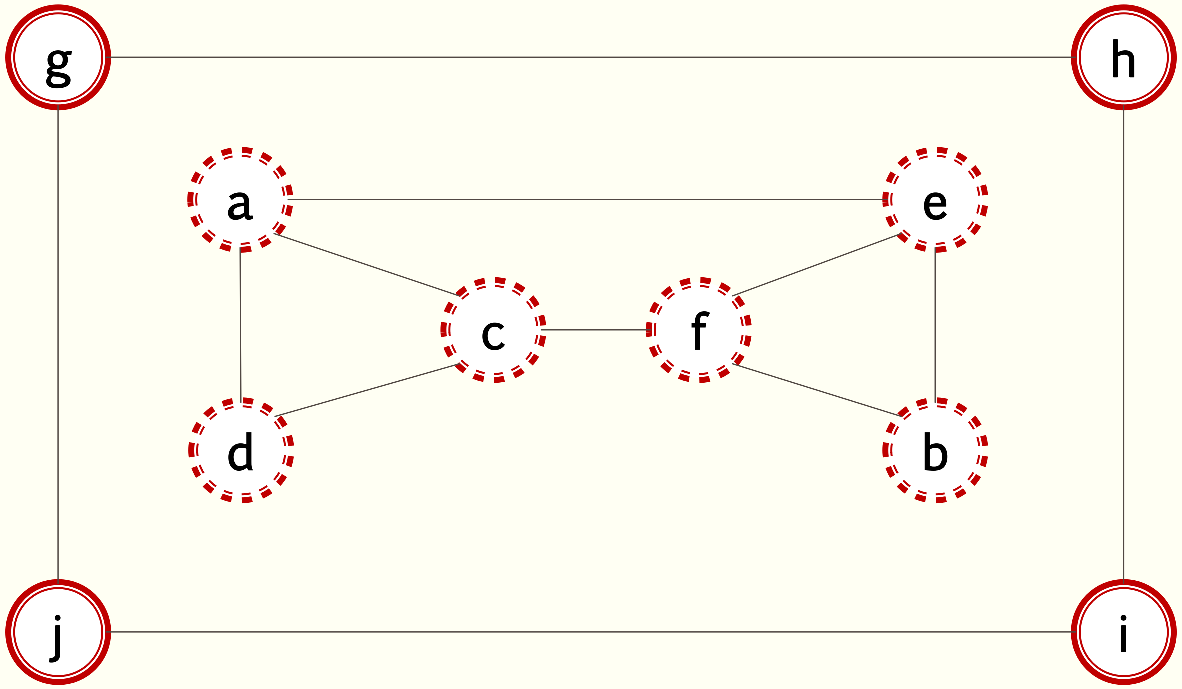
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a
g	
h	

DFS for same graph



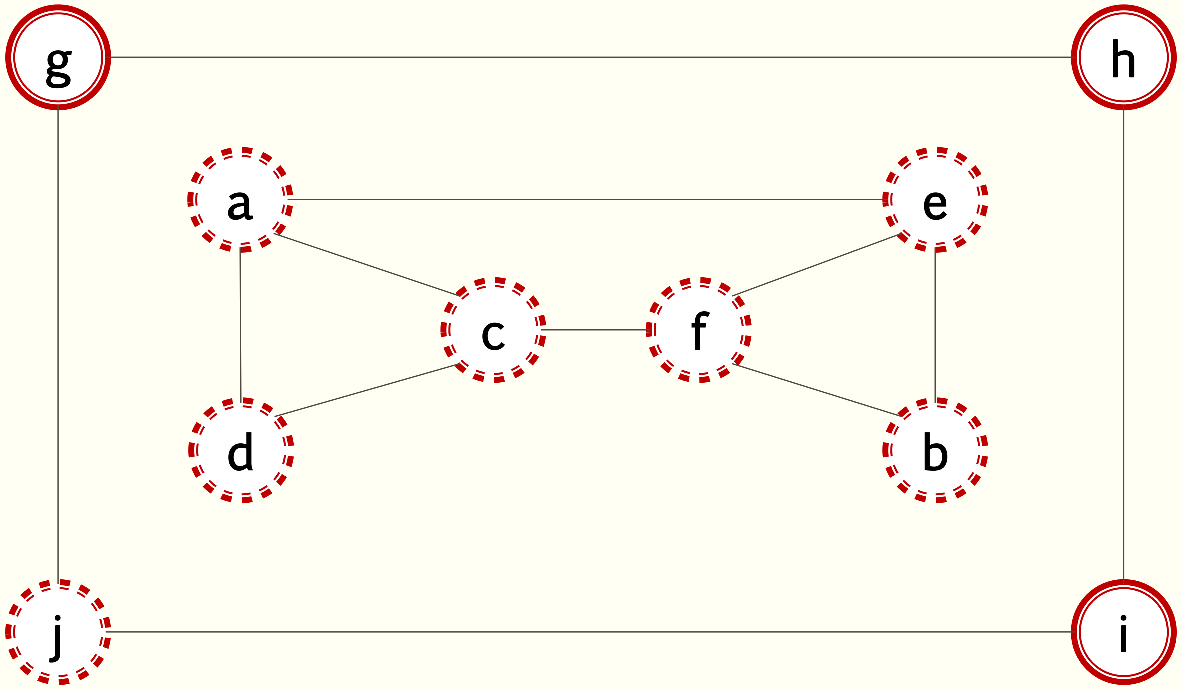
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a
g	
h	
i	

DFS for same graph



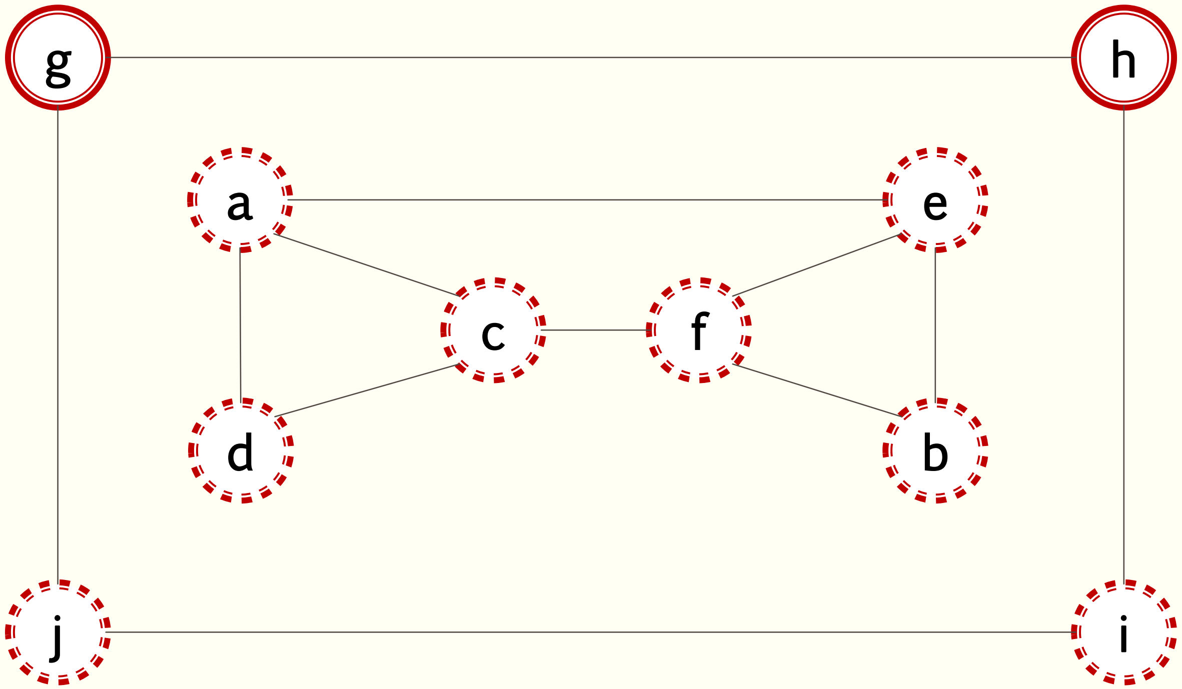
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a
g	
h	
i	
j	

DFS for same graph



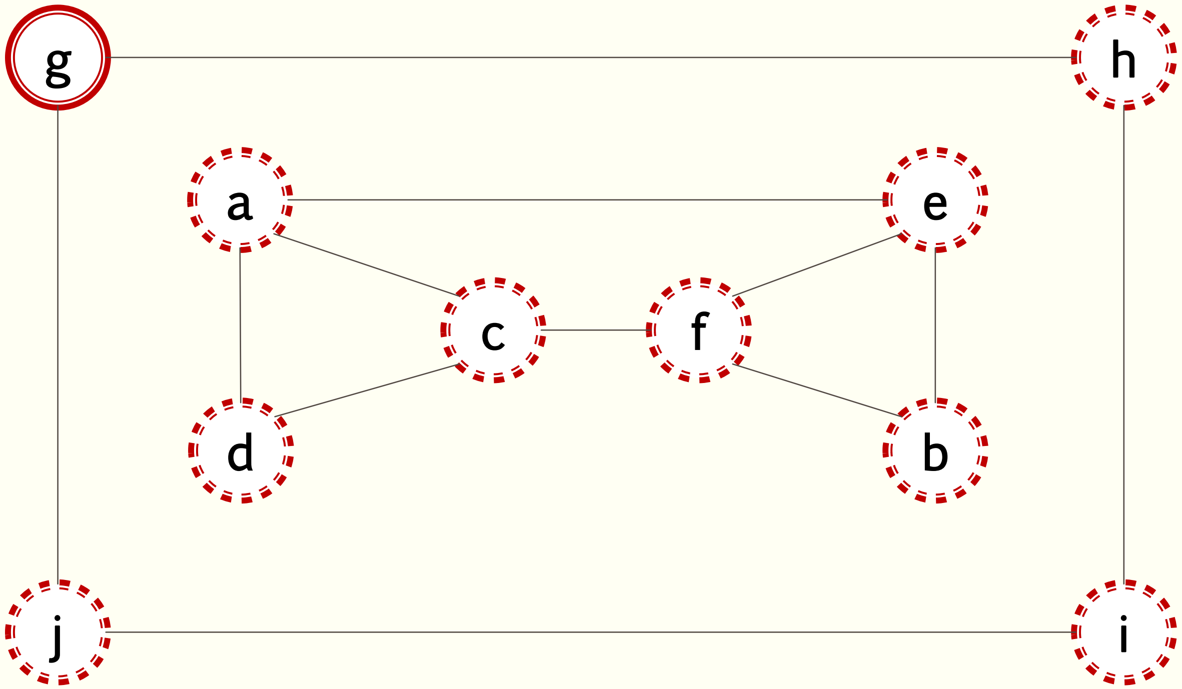
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a
g	j
h	
i	
j	

DFS for same graph



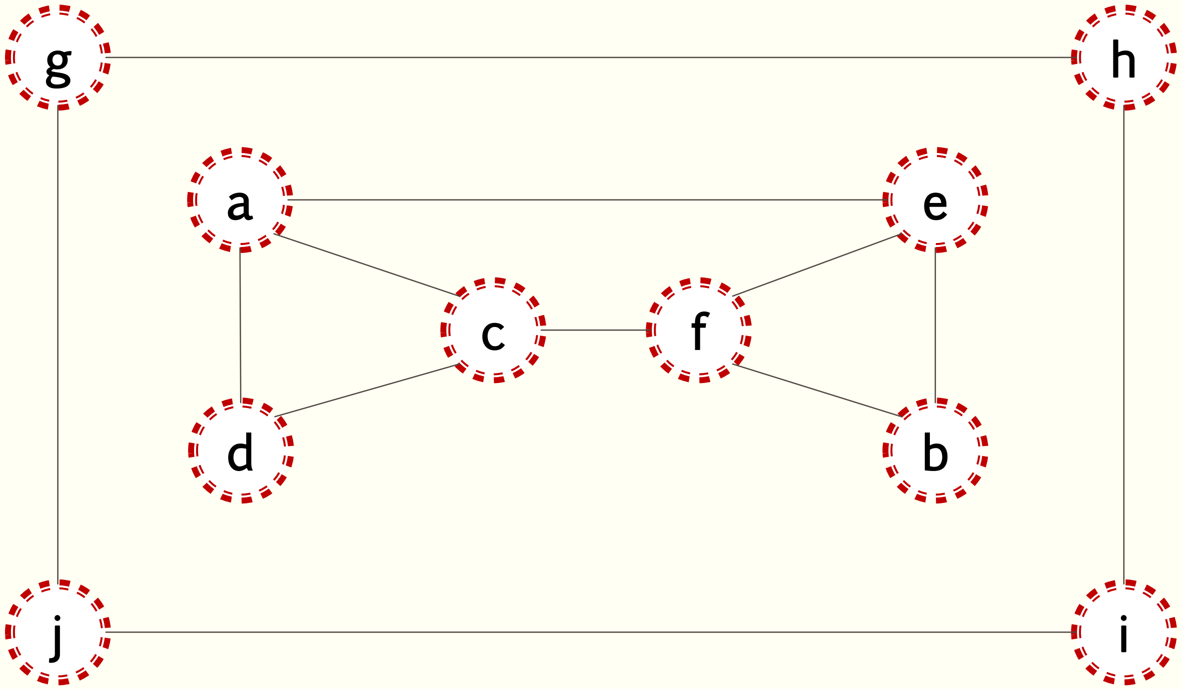
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a
g	j
h	i
i	
j	

DFS for same graph



Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a
g	j
h	i
i	h
j	

DFS for same graph



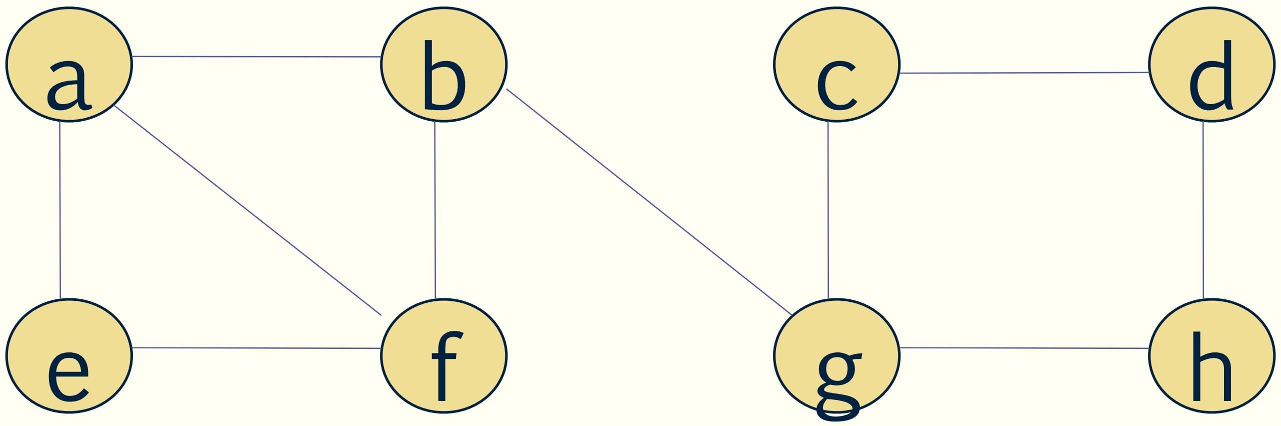
Into stack order	out of stack order
a	d
c	e
d	b
f	f
b	c
e	a
g	j
h	i
i	h
j	g

DFS vs BFS

TABLE 3.1 Main facts about depth-first search (DFS) and breadth-first search (BFS)

	DFS	**BFS**
Data structure	a stack	a queue
Number of vertex orderings	two orderings	one ordering
Edge types (undirected graphs)	tree and back edges	tree and cross edges
Applications	connectivity, acyclicity, articulation points	connectivity, acyclicity, minimum-edge paths
Efficiency for adjacency matrix	$\Theta(\|V^2\|)$	$\Theta(\|V^2\|)$
Efficiency for adjacency lists	$\Theta(\|V\| + \|E\|)$	$\Theta(\|V\| + \|E\|)$

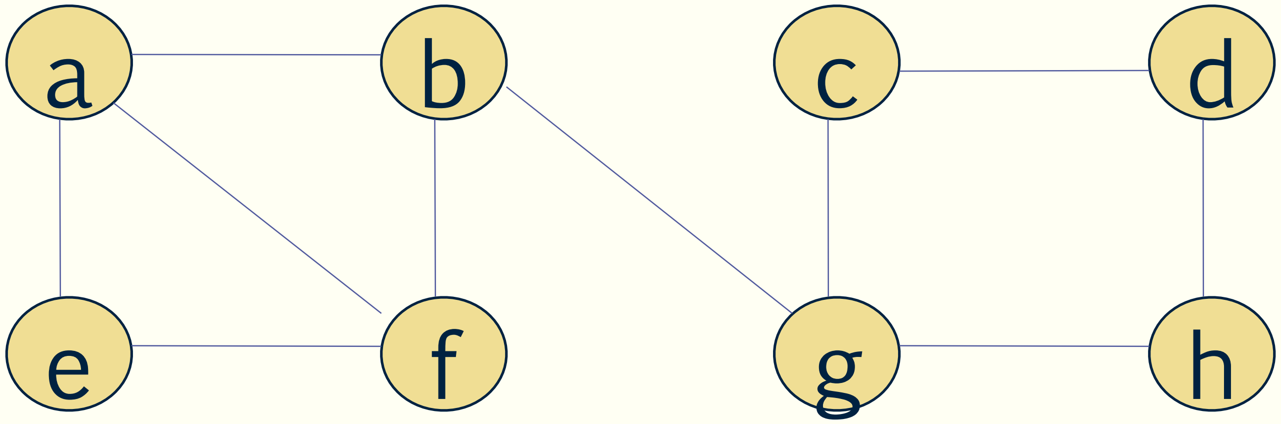
Example: BFS traversal of undirected graph



- BFS tree

- BFS queue

Example: DFS traversal of undirected graph

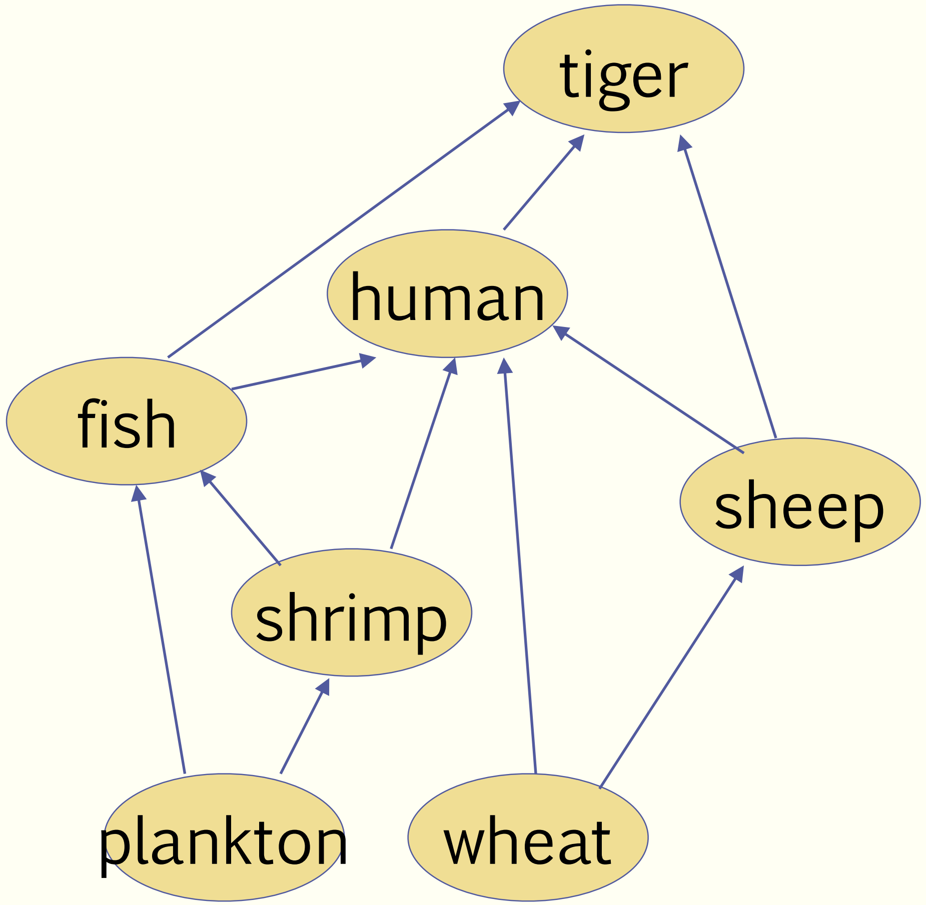


- DFS tree

- DFS stack

Topological Sorting Example

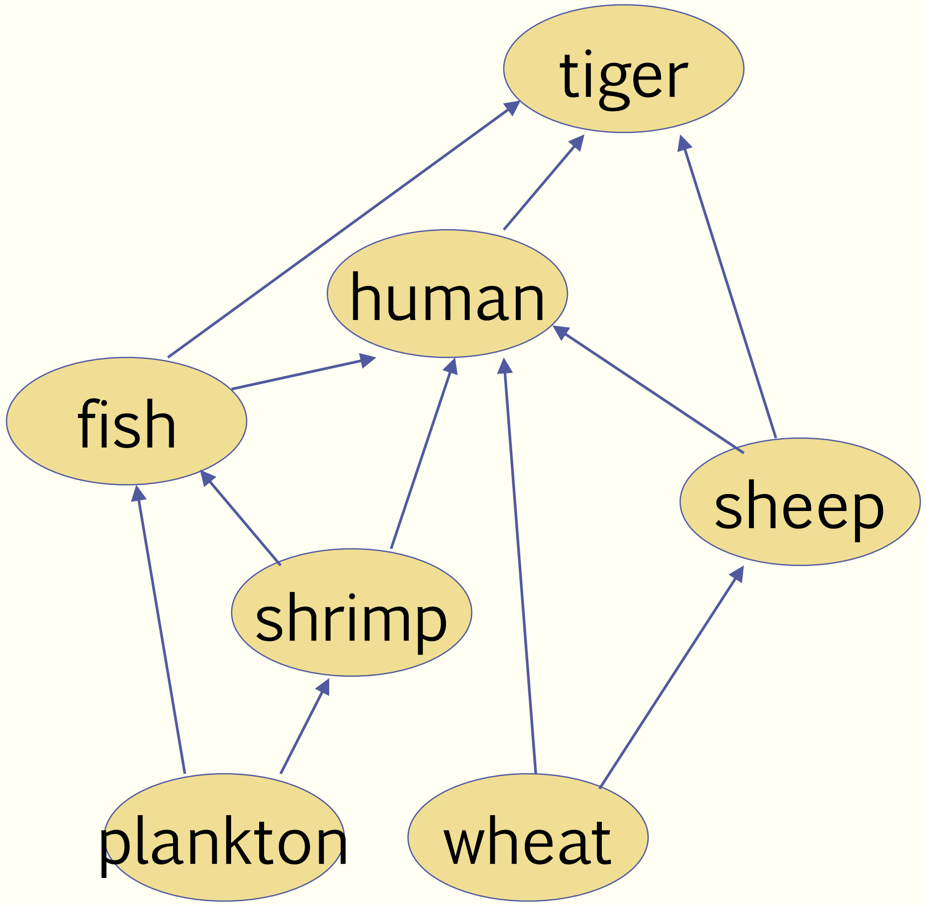
- Order the following in a food chain



Topological Sort using decrease-by-one

- Basic Idea
  - topsort a graph with one less vertex
  - combine the additional vertex with the sorted graph
- Problem:
  - How to choose a vertex that can be easily re-combined?

Which vertex should we remove?



- fish
- shrimp
- plankton
- wheat
- sheep
- human
- tiger