CS202 - Algorithm Analysis

Graph Algorithms - Module2

Aravind Mohan

Allegheny College

April 6, 2023

Graph Traversal

- **Graph traversal** is a process used to visit each node or vertex in a connected graph.
- Two popular algorithms are generally used for the traversal of a graph, namely, Depth First Search (DFS) and Breadth First Search (BFS).

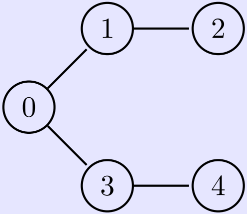
An important: application of graph traversal is to detect cycles in a Graph.

Graph Traversal

- **Graph traversal** is a process used to visit each node or vertex in a connected graph.
- Two popular algorithms are generally used for the traversal of a graph, namely, Depth First Search (DFS) and Breadth First Search.

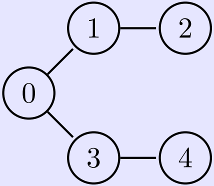
An important: application of graph traversal is to detect cycles in a Graph.

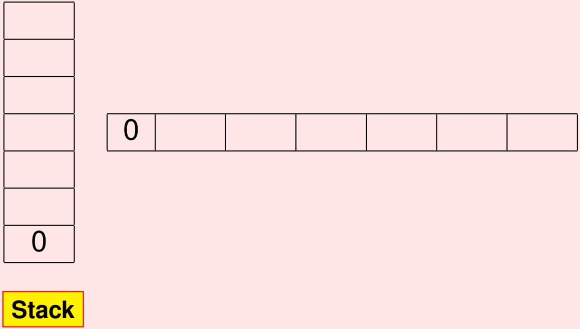
DFS example



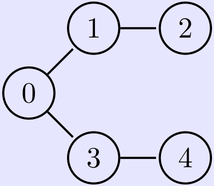
Traverse graph by starting from 0

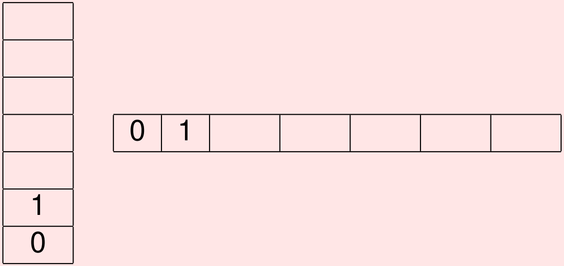
DFS example





DFS example



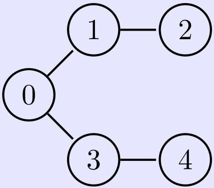


1
0

0	1					

Stack

DFS example

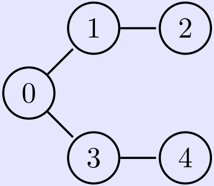


2
1
0

Stack

0	1	2				

DFS example

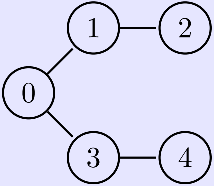


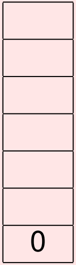
1
0

Stack

0	1	2				

DFS example

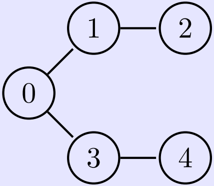


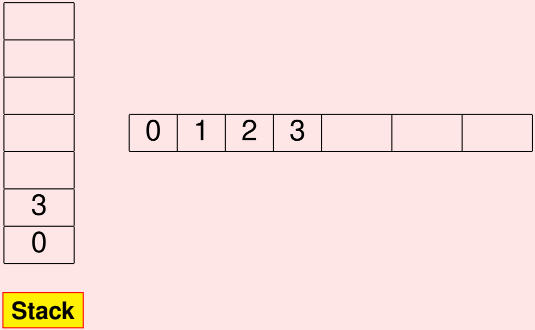


Stack

0	1	2				

DFS example



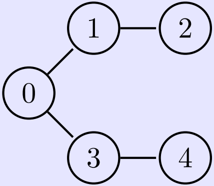


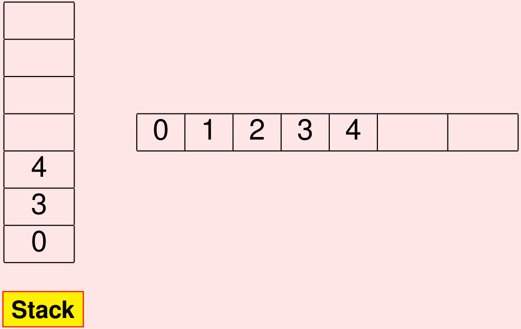
0	1	2	3			

```
3
0
```

Stack

DFS example



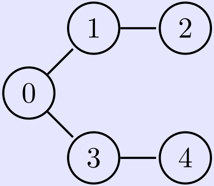


4
3
0

Stack

0	1	2	3	4		

DFS example

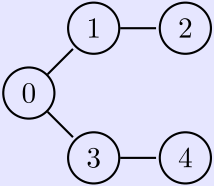


3
0

Stack

0	1	2	3	4		

DFS example [redacted]

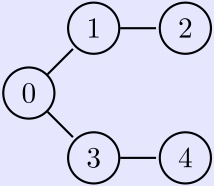


0

Stack

0	1	2	3	4		

DFS example



0	1	2	3	4		

Stack

Depth First Search (DFS) Algorithm

DFS(G, u)

Input: Graph G = (V, E) directed or undirected, vertex u (element of) V

Output: DFS traversal order from node u

```
create a Stack S
mark u as visited
push(S, u)
while S is not empty
    v = peek(S)
    if v has an unvisited neighbour w then
        mark w as visted
        push(S, w)
    else
        pop(S)
    end if
end while
```

Complexity Analysis

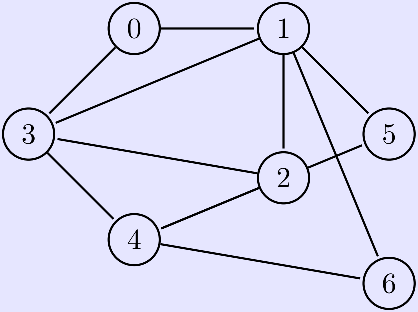
Time Complexity - O(V + E)

Detect Cycle

How do we detect cycles in a Graph using DFS?

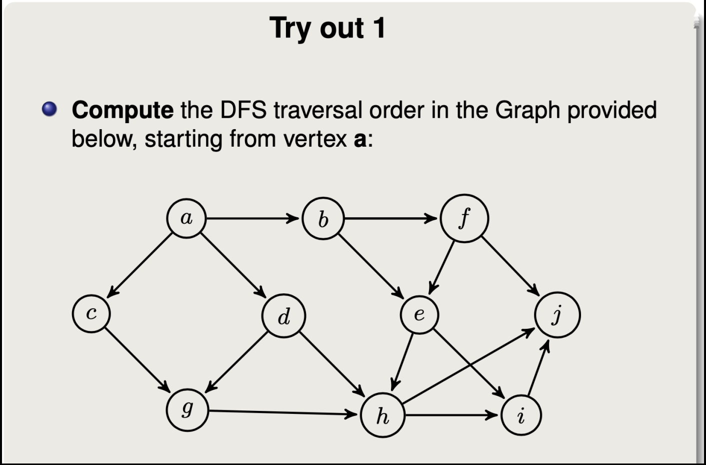
During DFS traversal, if we find a node that is already in the Stack then there is a cycle.

Let us Try this



Traverse graph by starting from 0

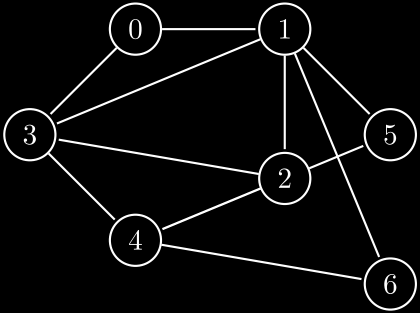
DFS Exercise



Try out 1

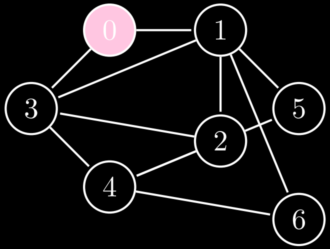
- **Compute** the DFS traversal order in the Graph provided below, starting from vertex **a**:

BFS example



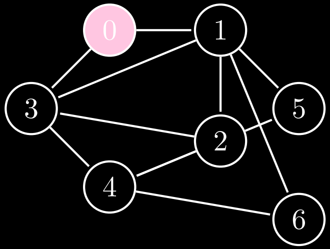
Traverse graph by starting from 0

BFS example (1)



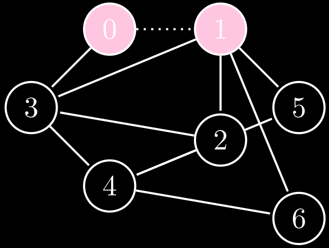
Queue	0						
Result							

BFS example (1)



Queue	~~0~~	1	3				
Result	0						

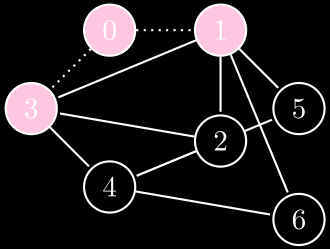
BFS example (1)



Queue	~~1~~	3	2	5	6		

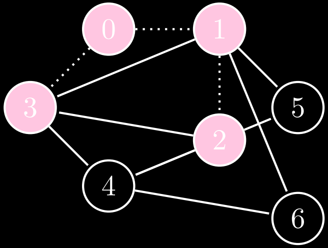
Result	0	1					

BFS example (1)



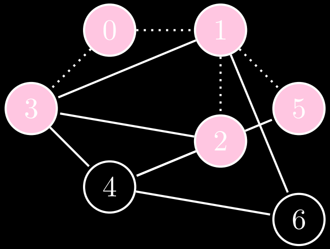
Queue	~~3~~	2	5	6	4		
Result	0	1	3				

BFS example (1)



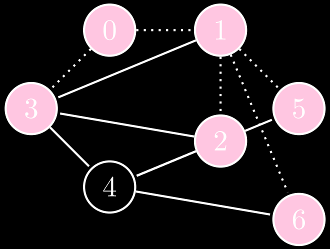
Queue	~~2~~	5	6	4			
Result	0	1	3	2			

BFS example (1)



Queue	~~5~~	6	4				
Result	0	1	3	2	5		

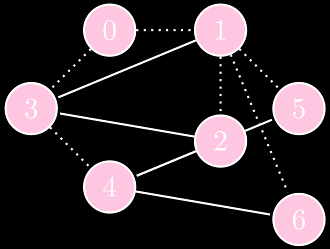
BFS example (1)



Queue	~~6~~	4					

Result	0	1	3	2	5	6	

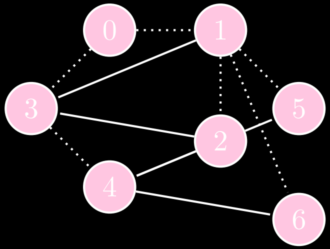
BFS example (1)



Queue	4						

Result	0	1	3	2	5	6	4

BFS example (1)



Queue							
Result	0	1	3	2	5	6	4

Breadth First Search (BFS) Algorithm

BFS(G, u)

Input: Graph G = (V, E) directed or undirected, vertex u (element of) V

Output: BFS traversal order from node u

```
create a Queue Q
enqueue(Q, u)
while(Q is not empty)
  s = Dequeue(Q)
  if (visited of vertex s == false)
    result.add(s)
    visited of vertex s = true
  end if
  for all m (element of) neighbours of s
    if (visited of vertex m == false and m not in Q)
      enqueue(Q, m)
    end if
  end for
end while
return result
```

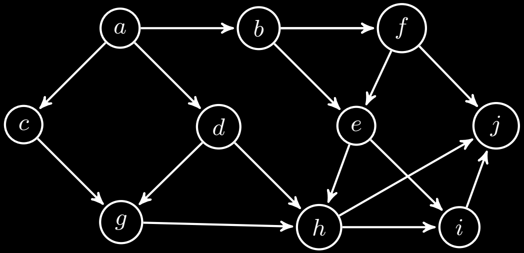
Complexity Analysis

Time Complexity - O(V + E)

Graph - Exercise

Try out 2

- **Compute** the BFS traversal order in the Graph provided below, starting from vertex **a**. Show the Queue and Result array in your solution.



Next:

- **Graph Shortest Path Algorithms:** Dijkstras algorithm.

Reading Assignment

Sedgewick 4.1 and 4.2

Questions?

Please ask if there are any Questions
through Slack, Email, and/or during the virtual office hours!