Algorithms

Graph Searching Techniques:

Breadth-First Search (BFS)

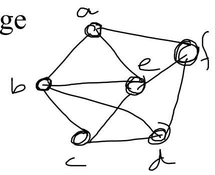
Level Order Traversal using Queue

Graph Searching

Quene

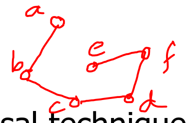
a, b, e, f, c, d

- Given: a graph G = (V, E), directed or undirected
- Goal: methodically explore every vertex and every edge
- Ultimately: build a tree on the graph
  - Pick a vertex as the root
  - Choose certain edges to produce a tree
  - Note: might also build a *forest* if graph is not connected

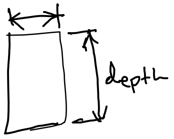


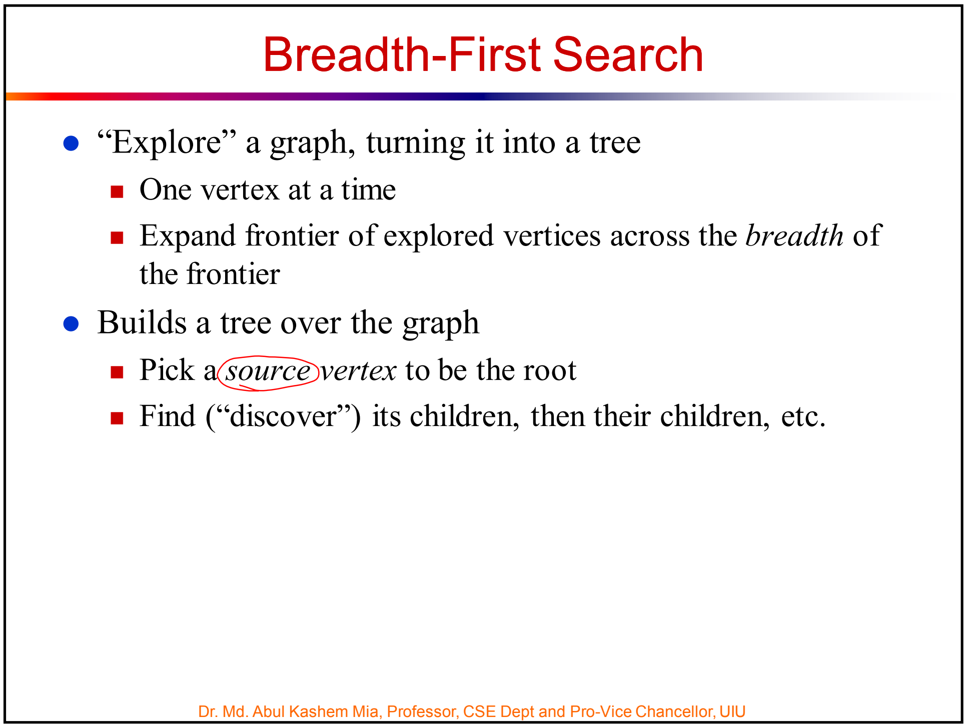
Stack a, b, c, d, f, e

backtrack



- There are two standard graph traversal techniques:
  - Breadth-First Search (BFS) ← FIFO Queue (explicit)
  - Depth-First Search (DFS) ← stack (implicit) recursion

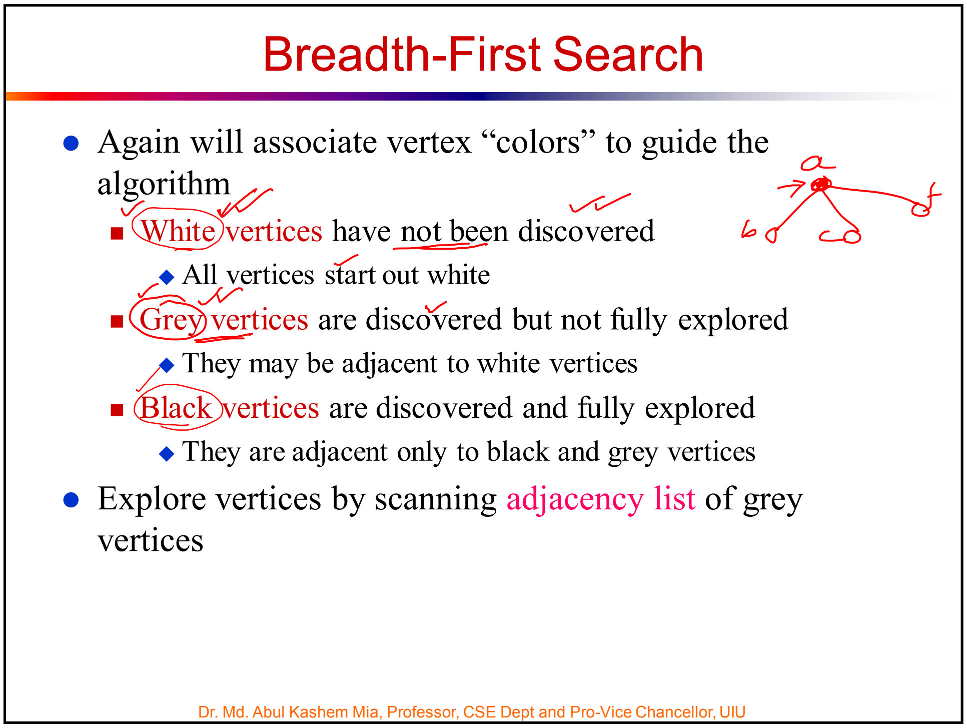




Breadth-First Search

- "Explore" a graph, turning it into a tree
  - One vertex at a time
  - Expand frontier of explored vertices across the *breadth* of the frontier
- Builds a tree over the graph
  - Pick a *source vertex* to be the root
  - Find ("discover") its children, then their children, etc.

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

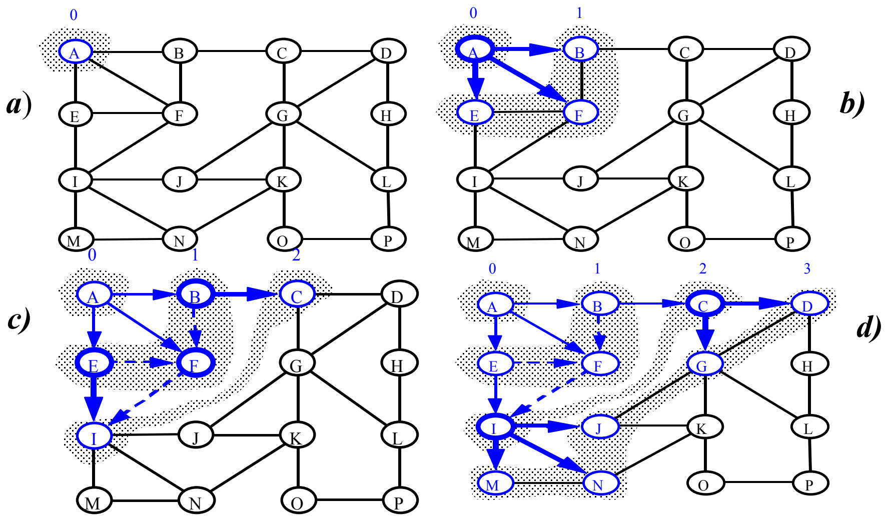


Breadth-First Search

- Again will associate vertex "colors" to guide the algorithm
  - White vertices have not been discovered
    - All vertices start out white
  - Grey vertices are discovered but not fully explored
    - They may be adjacent to white vertices
  - Black vertices are discovered and fully explored
    - They are adjacent only to black and grey vertices
- Explore vertices by scanning adjacency list of grey vertices

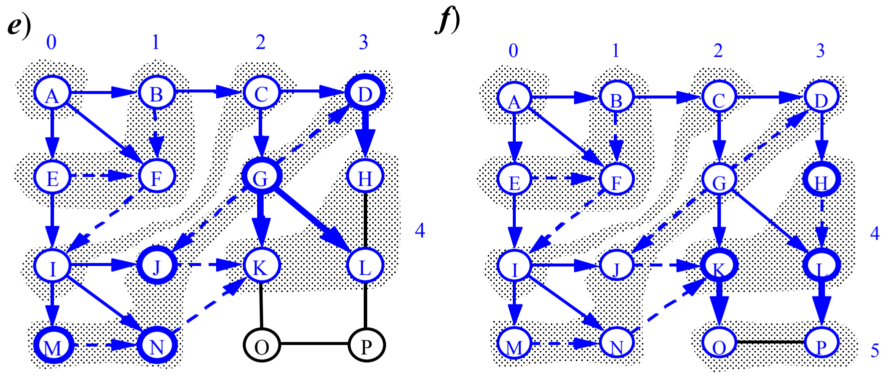
Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

BFS - A Graphical Representation

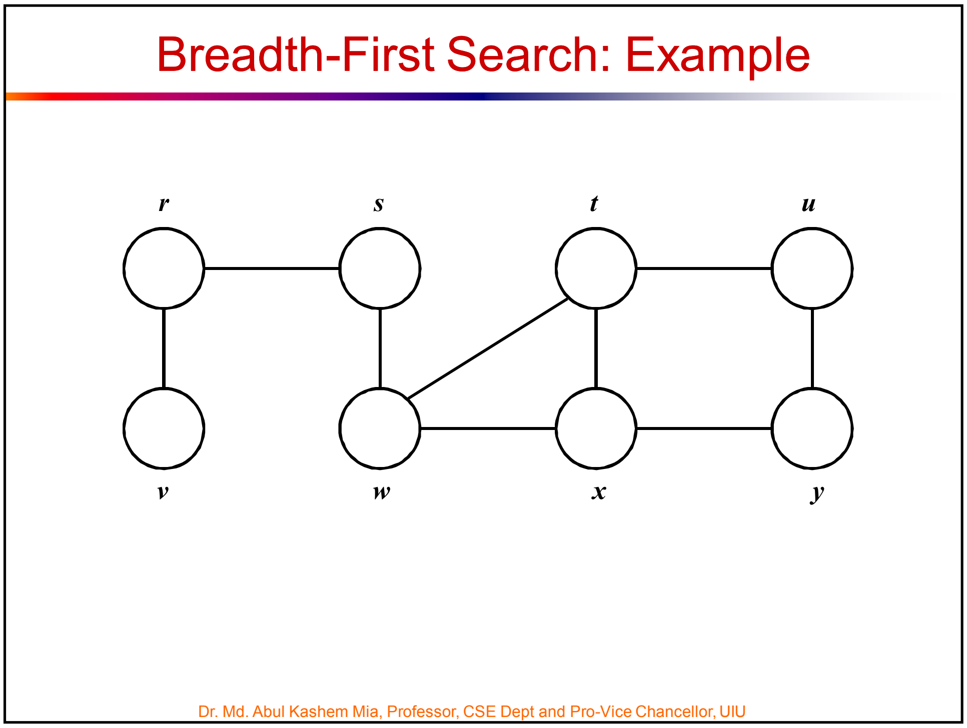


Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

BFS - A Graphical Representation

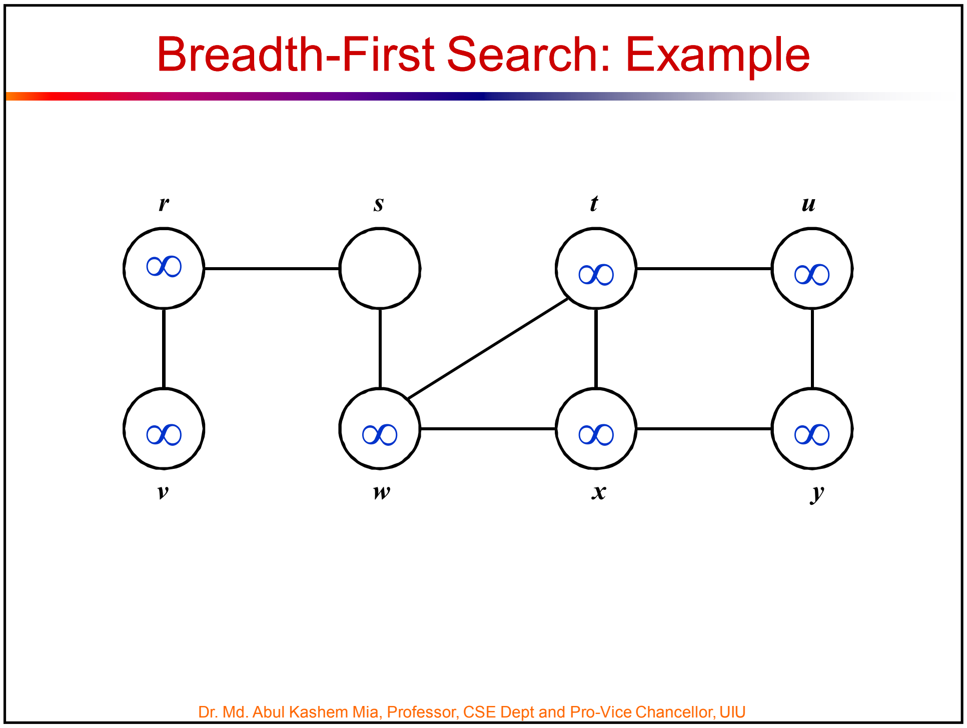


Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU



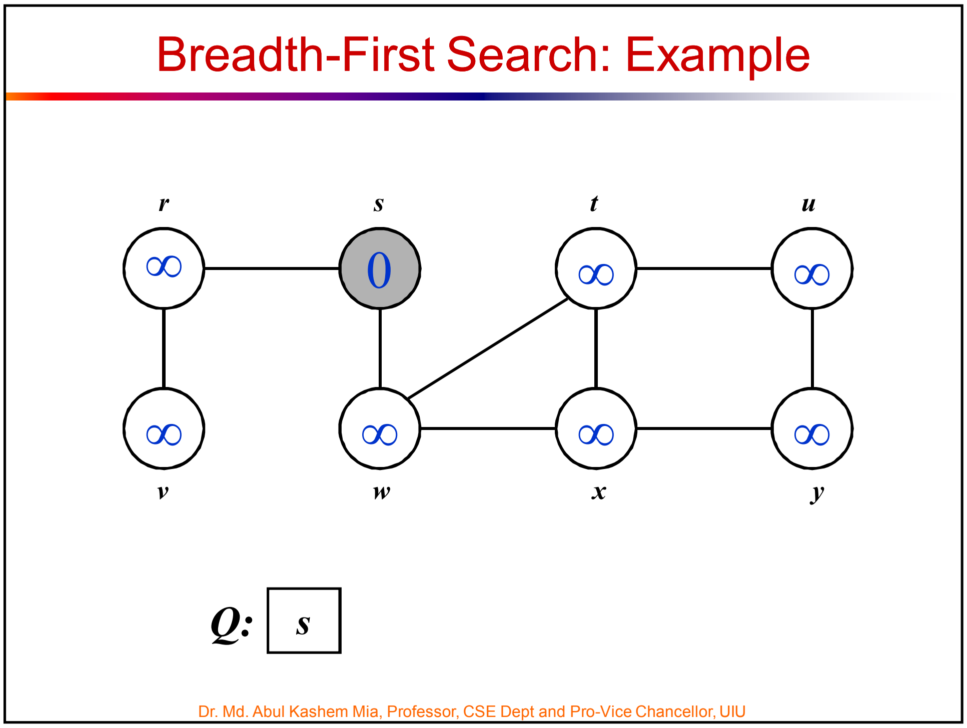
Breadth-First Search: Example

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU



Breadth-First Search: Example

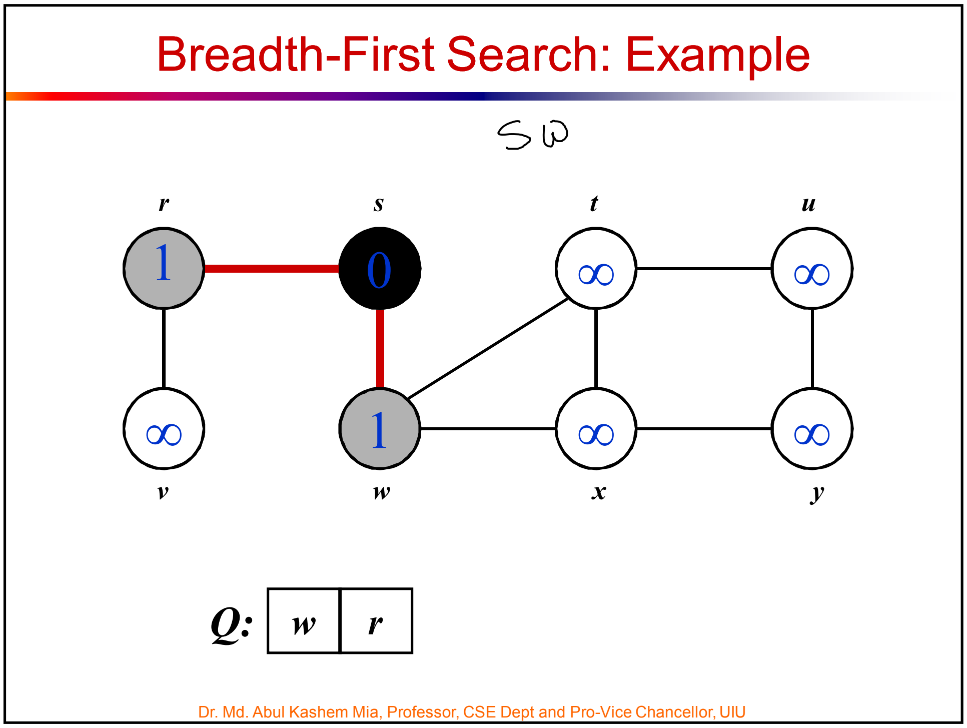
Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU



Breadth-First Search: Example

Q: | *s* |

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU



Breadth-First Search: Example

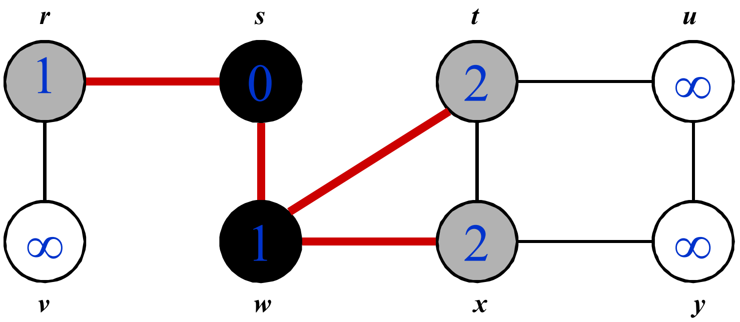
S W

Q: | *w* | *r* |

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

Breadth-First Search: Example

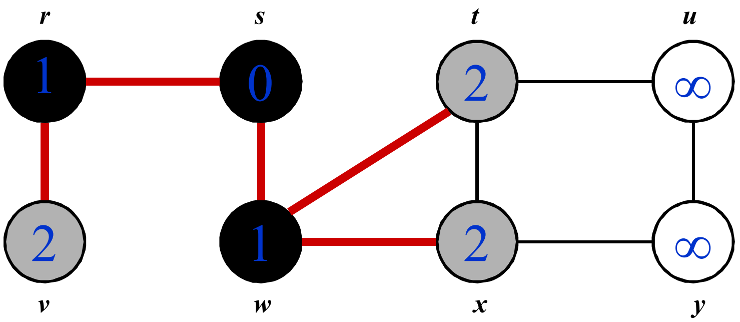
s o r



Q:	*r*	*t*	*x*

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

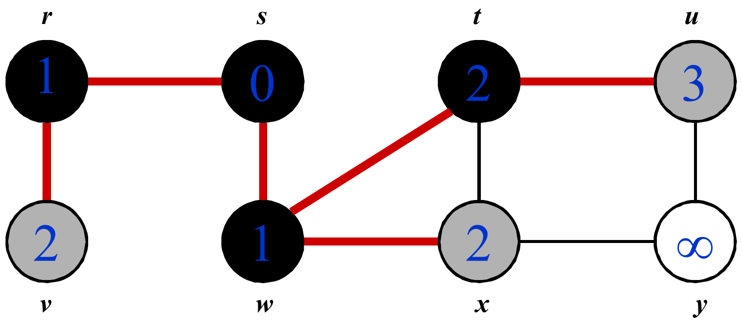
Breadth-First Search: Example



Q:	*t*	*x*	*v*

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

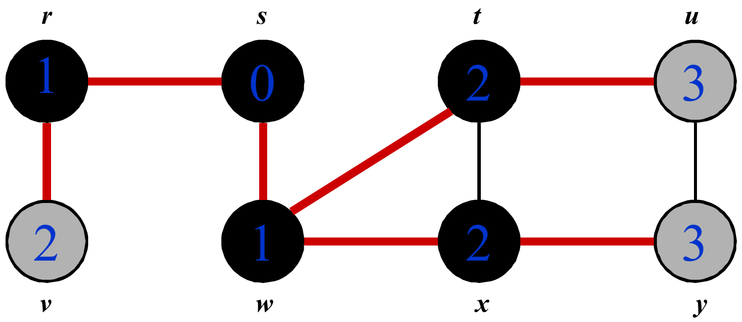
Breadth-First Search: Example



Q:			
	x	*v*	*u*

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

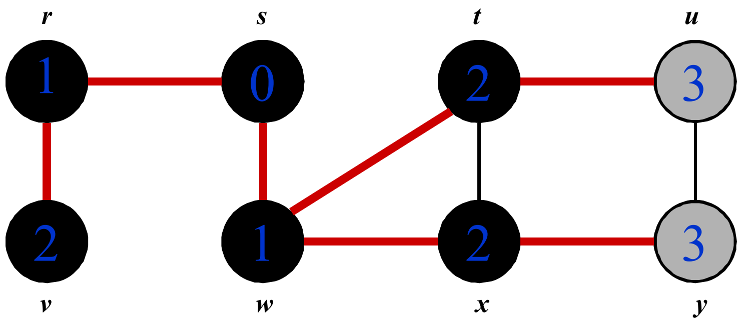
Breadth-First Search: Example



Q:			
	v	*u*	*y*

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

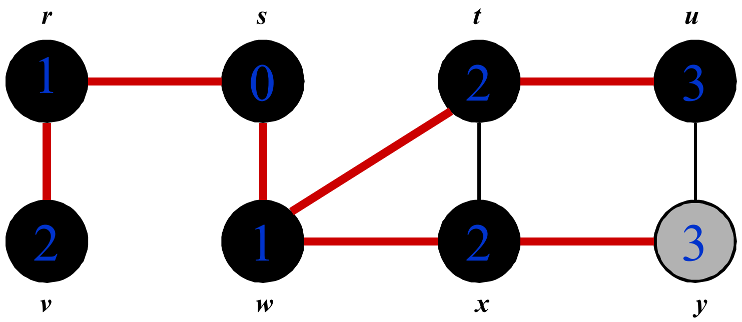
Breadth-First Search: Example



Q:	*u*	*y*

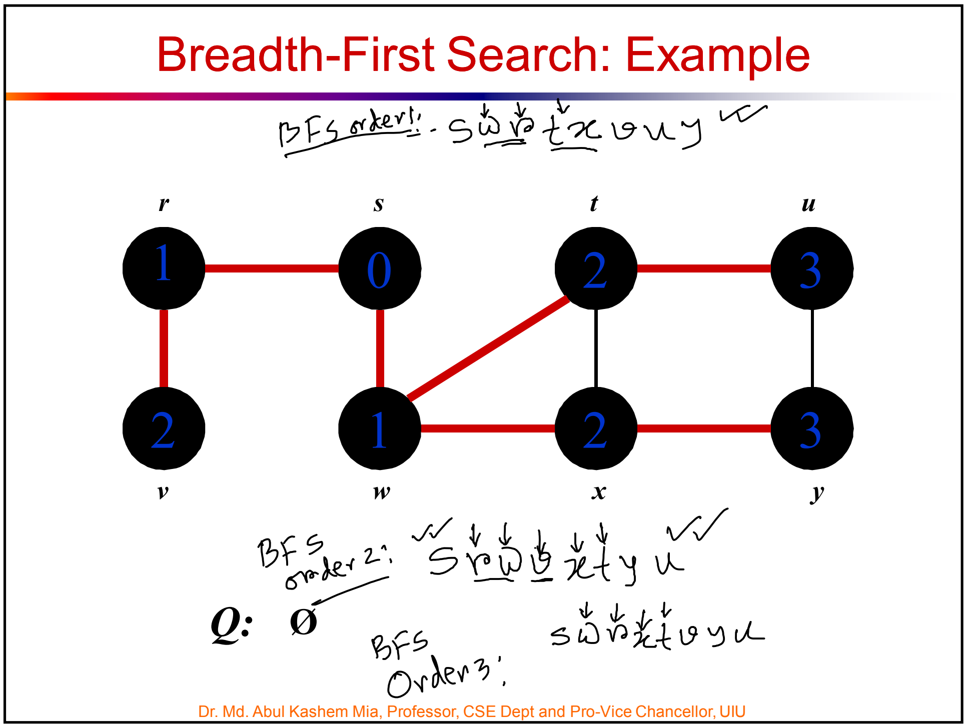
Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

Breadth-First Search: Example



Q:	*y*

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU



Breadth-First Search: Example

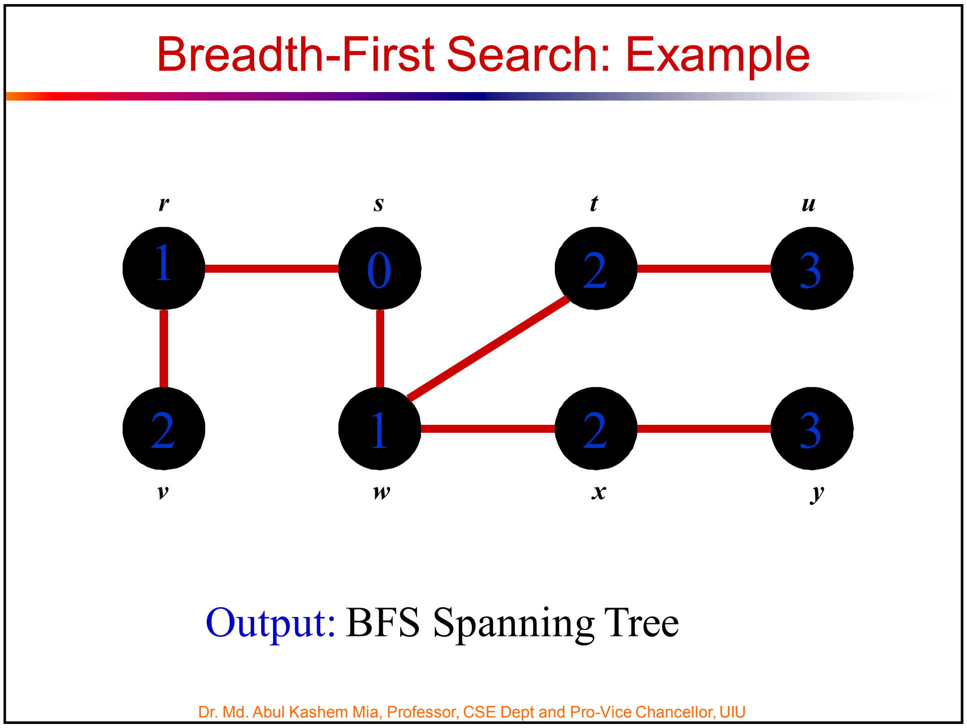
BFS order1: - s w r t x v u y ✓✓

BFS order 2: ✓✓ s r w v x t y u ✓✓

Q: Ø

BFS Order3: s w r x t v y u

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU



Breadth-First Search: Example

Output: BFS Spanning Tree

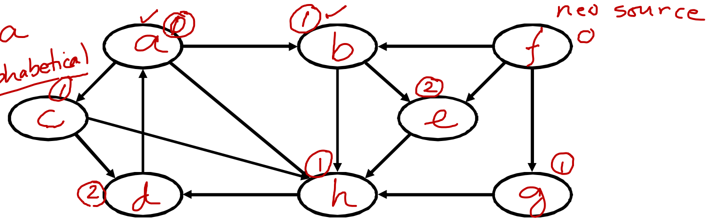
Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

Breadth-First Search: Example 2

Source : a

Go by alphabetical orders

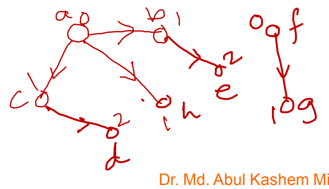
new source



BFS Order:

a b c h e d f g

BFS Spanning ~~Tree~~ Forest



Queue

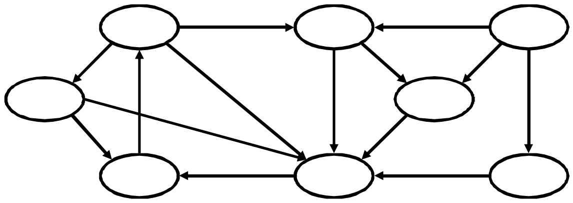
a → b c h → c h e

h e d → e d → d → ∅

f → g → ∅

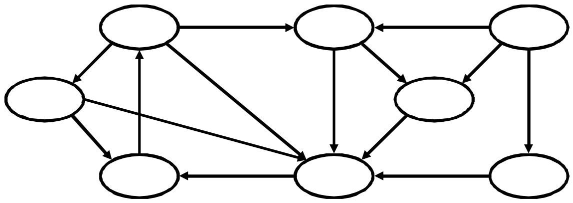
Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

Breadth-First Search: Example 2



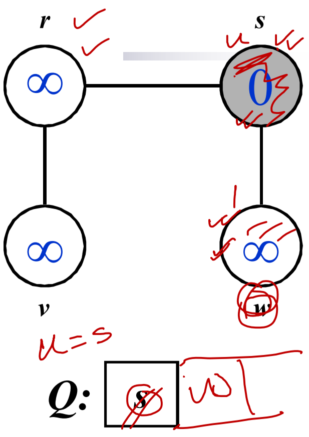
Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

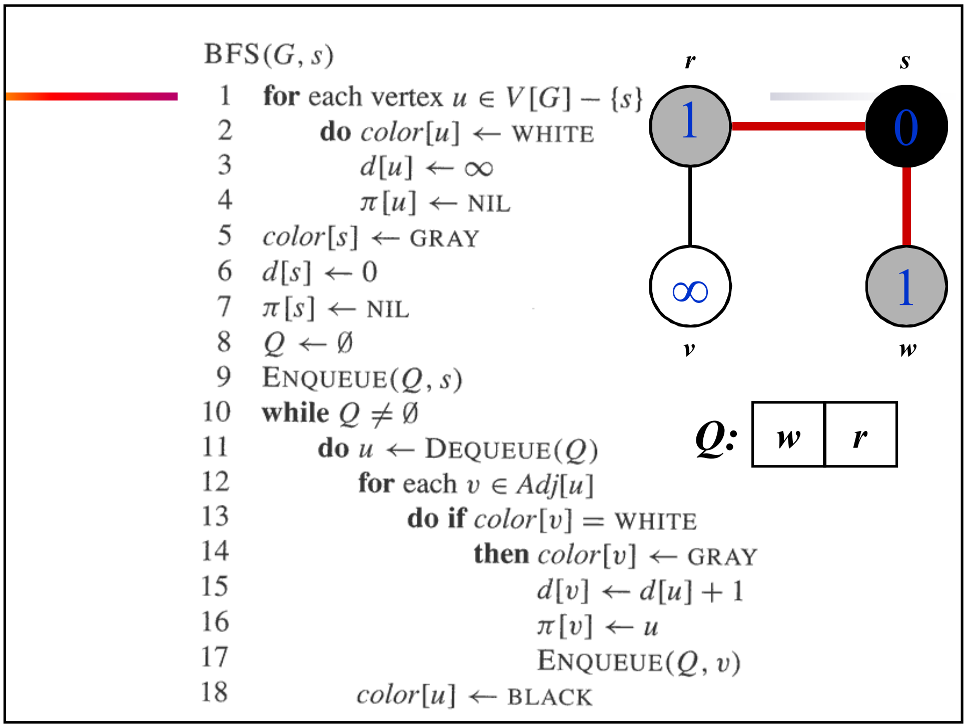
Breadth-First Search: Example 2



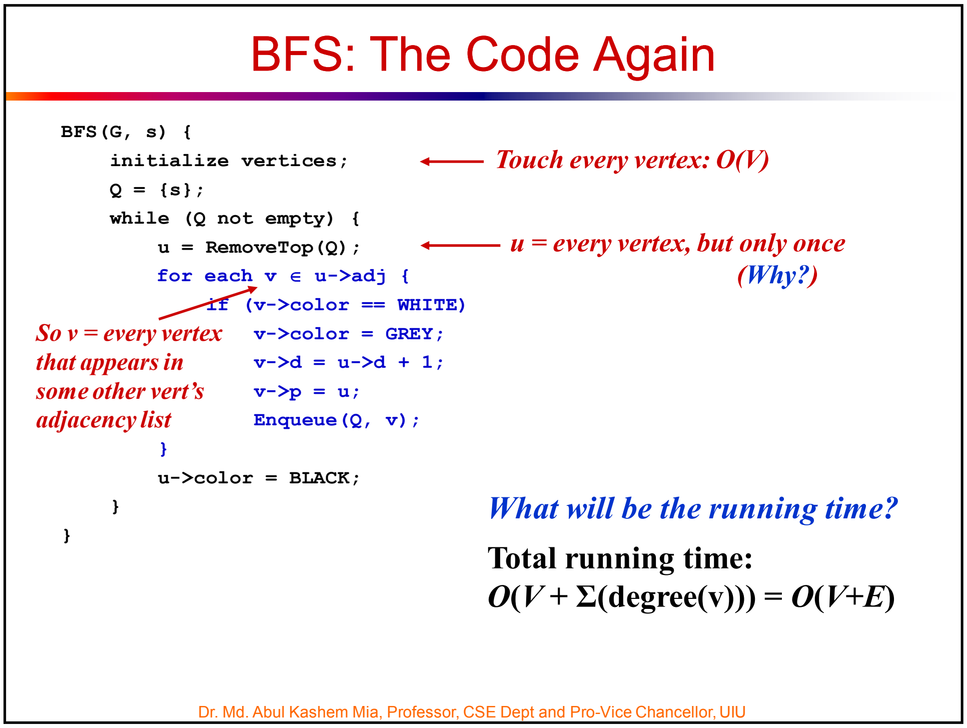
Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

```
BFS(G, s)
for each vertex $u \in V[G] - \{s\}$
    do $color[u] \leftarrow$ WHITE
        $d[u] \leftarrow \infty$
        $\pi[u] \leftarrow$ NIL
$color[s] \leftarrow$ GRAY
$d[s] \leftarrow 0$
$\pi[s] \leftarrow$ NIL
$Q \leftarrow \emptyset$
ENQUEUE($Q$, $s$)
while $Q \neq \emptyset$
    do $u \leftarrow$ DEQUEUE($Q$)
        for each $v \in Adj[u]$
            do if $color[v] =$ WHITE
                then $color[v] \leftarrow$ GRAY
                    $d[v] \leftarrow d[u] + 1$
                    $\pi[v] \leftarrow u$
                    ENQUEUE($Q$, $v$)
        $color[u] \leftarrow$ BLACK
```





```
BFS(G, s)
for each vertex $u \in V[G] - \{s\}$
    do $color[u] \leftarrow$ WHITE
        $d[u] \leftarrow \infty$
        $\pi[u] \leftarrow$ NIL
$color[s] \leftarrow$ GRAY
$d[s] \leftarrow 0$
$\pi[s] \leftarrow$ NIL
$Q \leftarrow \emptyset$
ENQUEUE($Q, s$)
while $Q \neq \emptyset$
    do $u \leftarrow$ DEQUEUE($Q$)
        for each $v \in Adj[u]$
            do if $color[v] =$ WHITE
                then $color[v] \leftarrow$ GRAY
                    $d[v] \leftarrow d[u] + 1$
                    $\pi[v] \leftarrow u$
                    ENQUEUE($Q, v$)
        $color[u] \leftarrow$ BLACK
```



BFS: The Code Again

```
BFS(G, s) {
    initialize vertices;
    Q = {s};
    while (Q not empty) {
        u = RemoveTop(Q);
        for each v ∈ u->adj {
            if (v->color == WHITE)
                v->color = GREY;
                v->d = u->d + 1;
                v->p = u;
                Enqueue(Q, v);
        }
        u->color = BLACK;
    }
}
```

Touch every vertex: O(V)

u = every vertex, but only once (Why?)

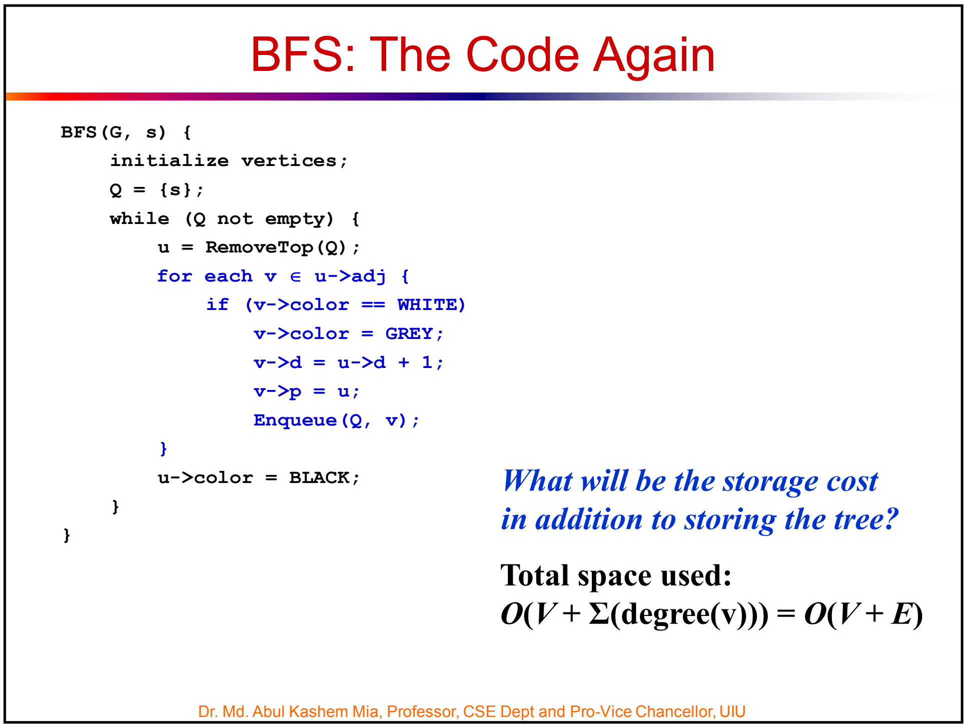
So v = every vertex that appears in some other vert's adjacency list

What will be the running time?

Total running time:

$O(V + \Sigma(\text{degree}(v))) = O(V+E)$

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

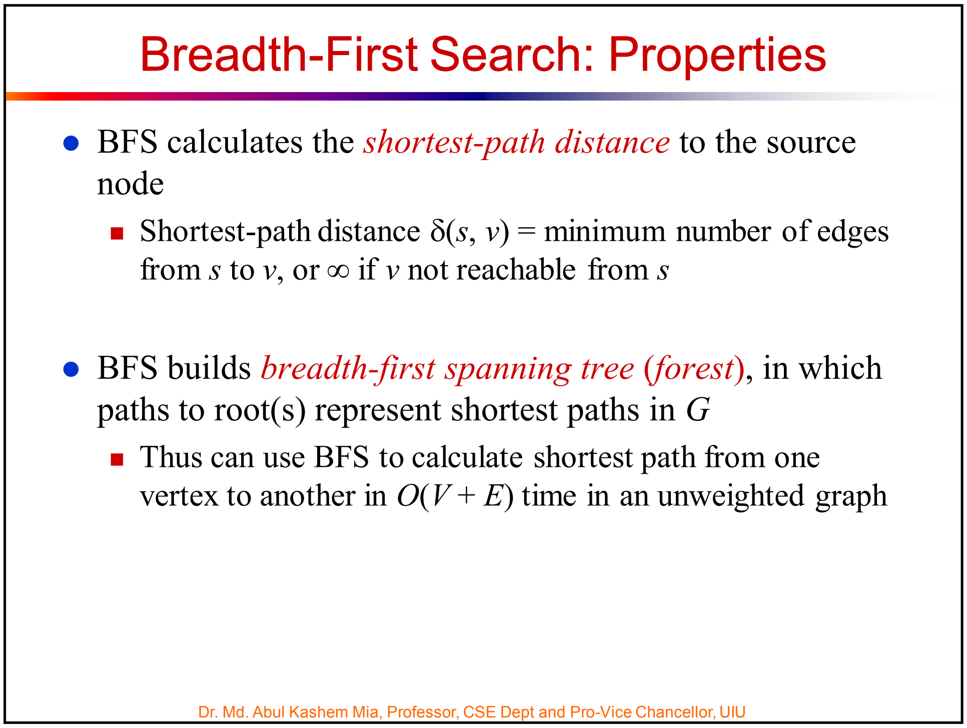


BFS: The Code Again

```
BFS(G, s) {
    initialize vertices;
    Q = {s};
    while (Q not empty) {
        u = RemoveTop(Q);
        for each v ∈ u->adj {
            if (v->color == WHITE)
                v->color = GREY;
                v->d = u->d + 1;
                v->p = u;
                Enqueue(Q, v);
        }
        u->color = BLACK;
    }
}
```

What will be the storage cost in addition to storing the tree?

Total space used:
$\boldsymbol{O(V + \Sigma(\text{degree}(v))) = O(V + E)}$



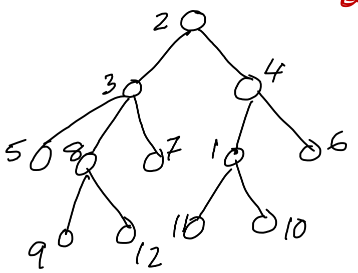
Breadth-First Search: Properties

- BFS calculates the *shortest-path distance* to the source node
  - Shortest-path distance $\delta(s, v)$ = minimum number of edges from s to v, or ∞ if v not reachable from s
- BFS builds *breadth-first spanning tree* (*forest*), in which paths to root(s) represent shortest paths in G
  - Thus can use BFS to calculate shortest path from one vertex to another in $O(V + E)$ time in an unweighted graph

Level Order Traversal using Queue

- In a level order traversal, every node on a level is visited before going to a lower level

Output: 2 3 4 5 8 7 1 6 9 12 11 10



Q: ∅

Q: ~~2~~

Q: ~~3~~ 4

Q: ~~4~~ 5 8 7

Q: ~~5~~ 8 7 1 6

Q: ~~8~~ 7 1 6

Q: ~~7~~ 1 6 9 12

Q: ~~1~~ 6 9 12

Q: ~~6~~ 9 12 11 10

Q: ~~6 9 12 11 10~~

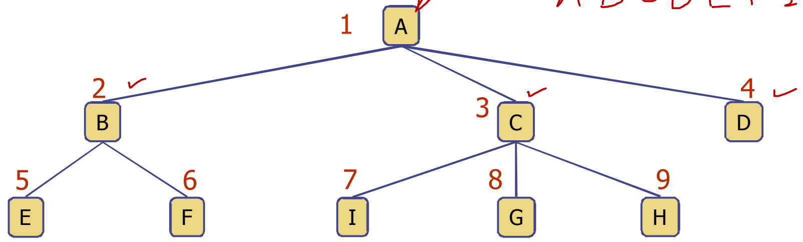
Q = ∅

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU

Level Order Traversal using Queue

- In a level order traversal, every node on a level is visited before going to a lower level

Output: A B C D E F I G H



Q = ∅

Q = ← ~~A~~

Q: ← ~~B~~ C D

Q: ← ~~C~~ D E F ←

Q: ~~D~~ E F I G H

Q: ← ~~E~~ F I G H

Q: ~~F~~ I G H

Q: ~~I~~ G H

Q: ~~G~~ H

Q: ~~H~~

Q: ∅

Dr. Md. Abul Kashem Mia, Professor, CSE Dept and Pro-Vice Chancellor, UIU