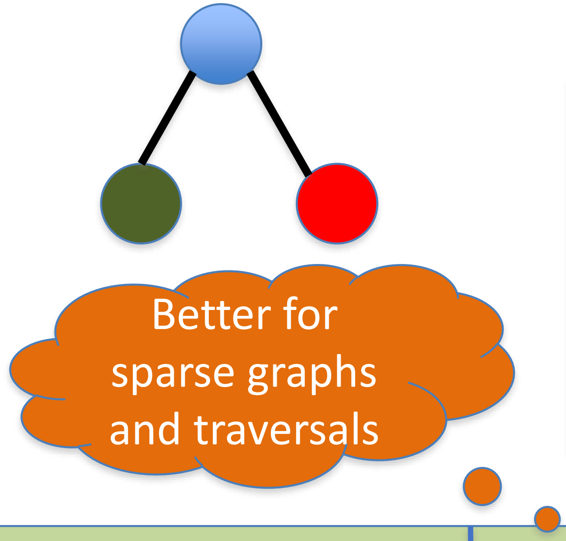
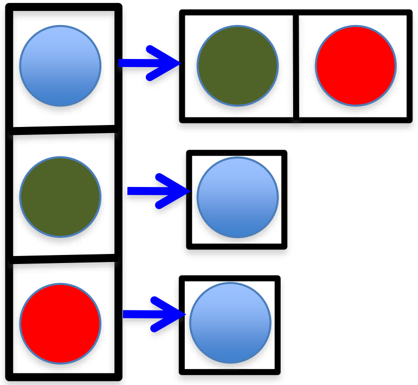
Lecture 14

CSE 331

Graph representations

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





Adjacency matrix		Adjacency List
O(1)	(u,v) in E?	O(n) [O(n_v)]
O(n)	All neighbors of u?	O(n_u)
O(n^2)	Space?	O(m+n)

2·# edges = sum of # neighbors

2m = Σ $_{u \text{ in } V}$ n_u

$$\Longrightarrow\ 2|E| = \sum_{u\, \in V} \deg(u)$$

$$\sum_{u\, \in V} \deg(u) = \deg(u_1) + \deg(u_2) + \cdots + \deg(u_n)$$
$$= n_{u_1} + n_{u_2} + \ldots + n_{u_n}$$

Breadth First Search (BFS)

Build layers of vertices connected to s

$L_0 = \{s\}$

Assume $L_0,..,L_j$ have been constructed

L_{j+1} set of vertices not chosen yet but are connected to L_j

Stop when new layer is empty

Use linked lists

Use CC[v] array

$O(m+n)$ BFS Implementation

BFS(s)

CC[s] = T and CC[w] = F for every $w \neq s$

Set $i = 0$

Set $L_0 = \{s\}$

While L_i is not empty

$L_{i+1} = \emptyset$

For every u in L_i

For every edge (u,w)

If CC[w] = F then

CC[w] = T

Add w to L_{i+1}

i++

return cc

O(m+n) BFS Implementation

BFS(s)

Array

Input graph as Adjacency list

$CC[s] = T$ and $CC[w] = F$ for every $w \neq s$

Set $i = 0$

Set $L_0 = \{s\}$

While L_i is not empty

Linked List

$L_{i+1} = \emptyset$

For every u in L_i

For every edge (u,w)

If $CC[w] = F$ then

$CC[w] = T$

Add w to L_{i+1}

i++

return cc

Version in KT also computes a BFS tree

All the layers as one

BFS(s)

CC[s] = T and CC[w] = F for every $w \neq s$

Set i = 0

Set L_0= {s}

While L_i is not empty

  $L_{i+1} = \emptyset$

  For every u in L_i

    For every edge (u,w)

    If CC[w] = F then

      CC[w] = T

      Add w to L_{i+1}

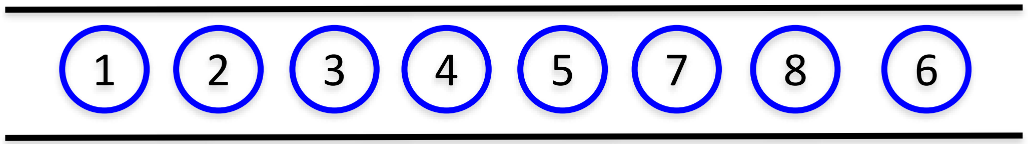
  i++

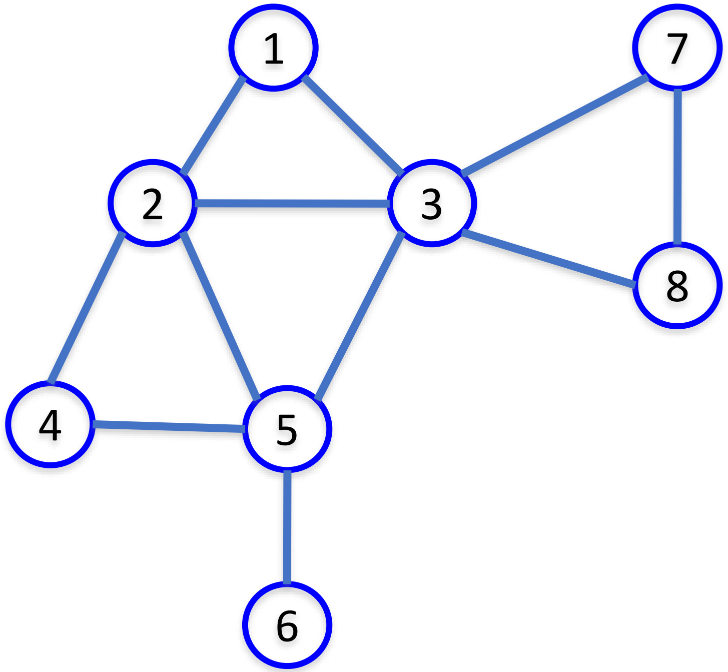
return cc

All layers are considered in first-in-first-out order

Can combine all layers into one queue: all the children of a node are added to the end of the queue

An illustration





Queue O(m+n) implementation

BFS(s)

CC[s] = T and CC[w] = F for every w≠ s

Intitialize Q= {s}

While Q is not empty

Delete the front element u in Q

For every edge (u,w)

If CC[w] = F then

CC[w] = T

Add w to the back of Q

O(n)

O(1)

Repea
onc

$\Sigma_u\ O(n_u) = O(\Sigma_u\ n_u) = O(m)$

O(1)

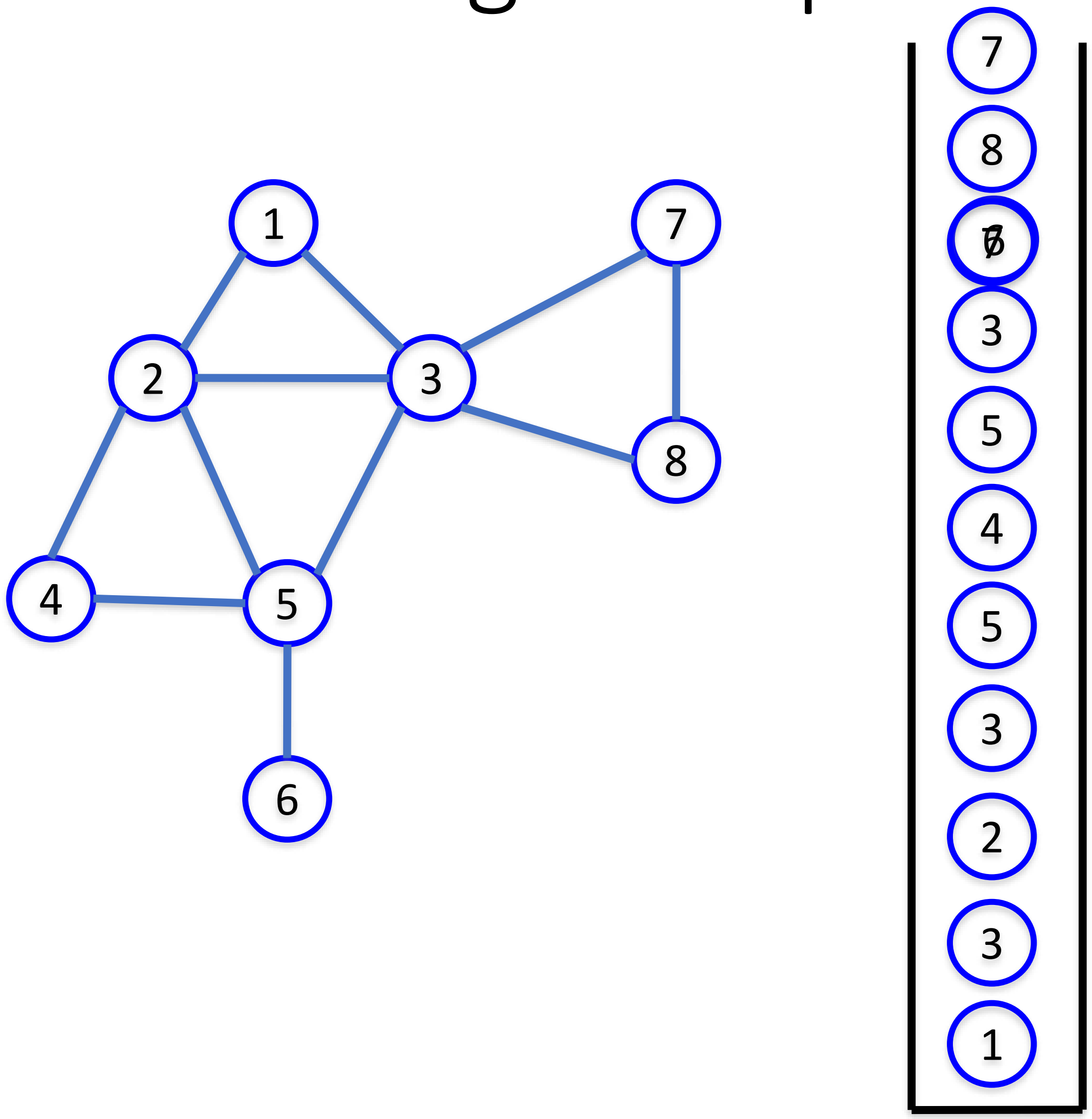
$O(n_u)$

O(1)

Implementing DFS in O(m+n) time

Same as BFS except stack instead of a queue

A DFS run using an explicit stack



DFS stack implementation

DFS(s)

CC[s] = T and CC[w] = F for every w≠ s

Intitialize Ŝ = {s}

While Ŝ is not empty

Pop the top element u in Ŝ

For every edge (u,w)

If CC[w] = F then

CC[w] = T

Push w to the top of Ŝ

Same O(m+n) run time analysis as for BFS

Reading Assignment

Sec 3.3, 3.4, 3.5 and 3.6 of [KT]