Date:

Black Board

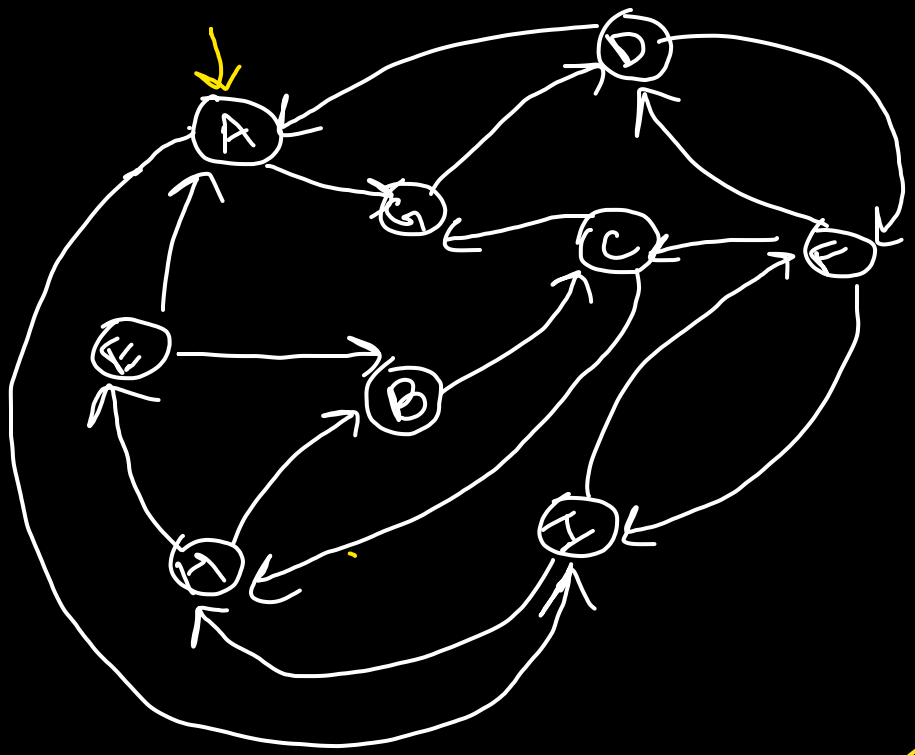
Data Structures

Lecture # 27

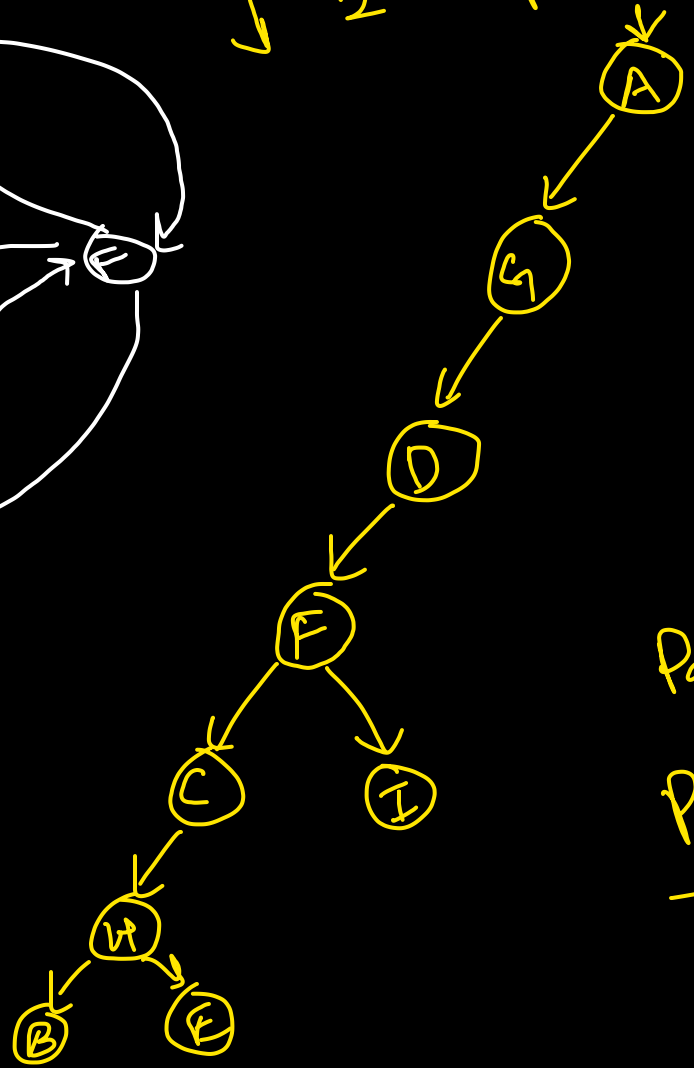
Topics:

√ **Breadth First Search**

Graph Traversal



A
1 hop from A
2 hop ....



par[F] = D

par[H] = C

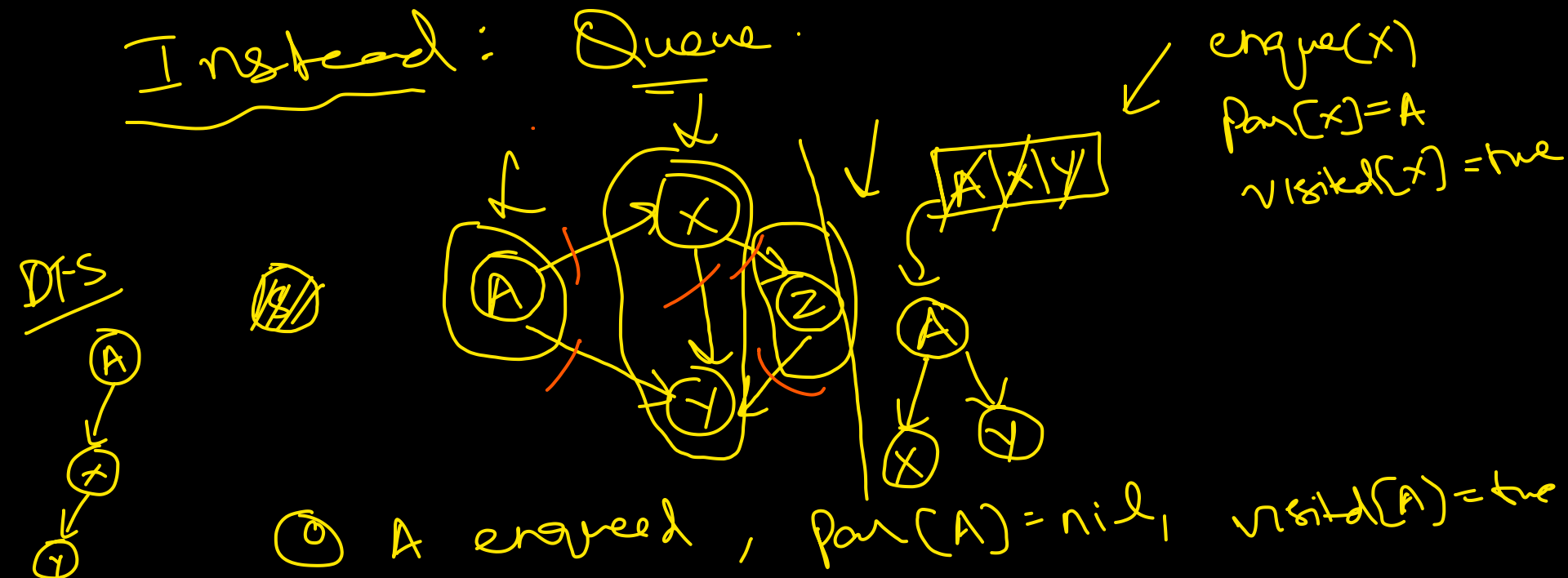
dfs()
{explore(G, x)

Breadth First Search

s
↳ 1 hop from s → 2 hops from s ....

→ Non_recursive ⟹ Not using stack

Instead: Queue.



① A enqueed, par(A)=nil, visited[A]=true

① → A extracted

→ x, y enqued — par[x] = par[y] = A

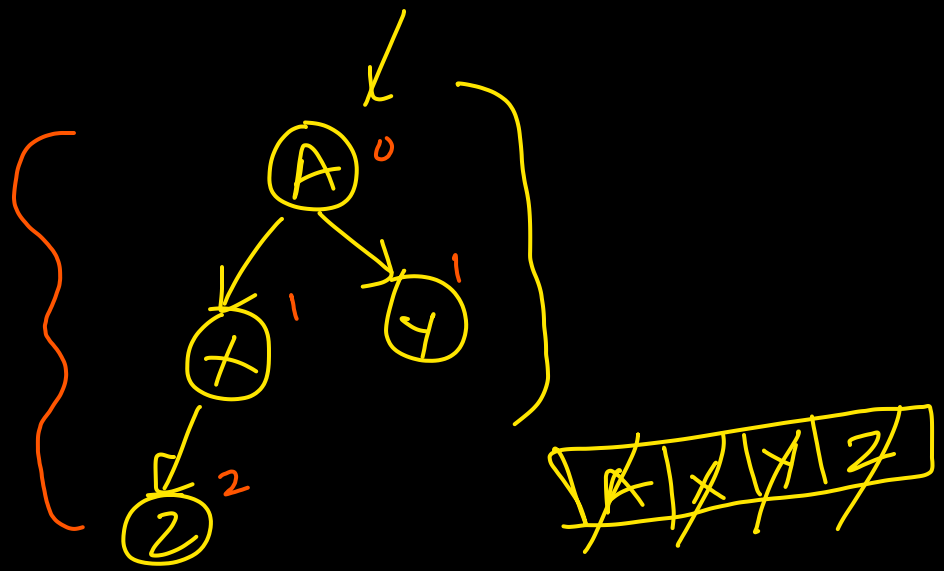
visited[x] = visited[y] = true

② → x extracted

→ z enqued, par[z] = x, visited[z] = true

③ → y extracted

④ → z extracted



BFS $(G=(V,E), s)$

Create Q

For each $x \in V$:
    visited[x] = false

O(|V|)

visited[s] = true
par[s] = nil
Q.enqueue(s)

O(1)

Set up

dist[s] = 0

While !Q.empty()

O(|V|)

$x \leftarrow$ Q.extract()

For each $(x,y) \in E$:

O(|E|)

O(1)

if !visited[y]
(

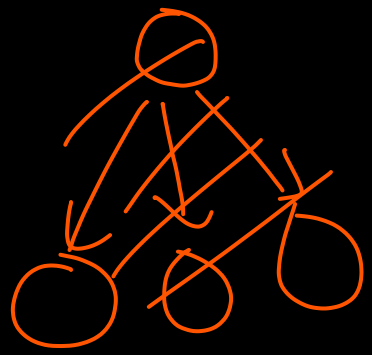
Q. enque(y)
visited(y) = true
Par[y] = x
dist[y] = dist[x] + 1

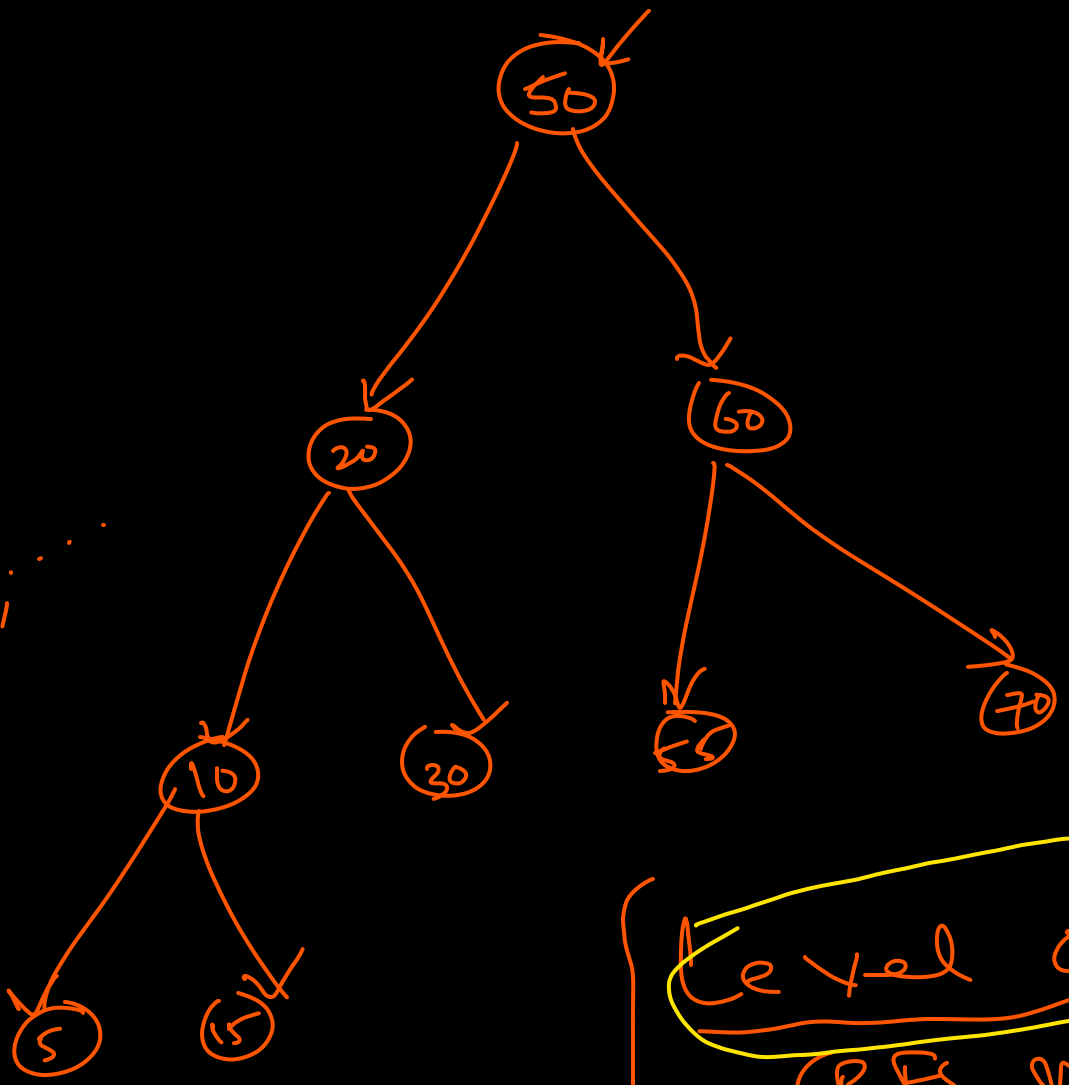
LNR — In order
NLR — Pre order
LRN — post order

recursive.

specialized DFS for BST

do not need visited array





6

{LNR}

5, 10, 15, 20, 30, 50, ...

Level order
(BFS you need)

50, 20, 60, 10, 30, 55, 70, 5, 15

BFS Running Time:

$$O(|V| + |E|)$$

$|V| = n$

$|E| = m$

$O(n + m)$