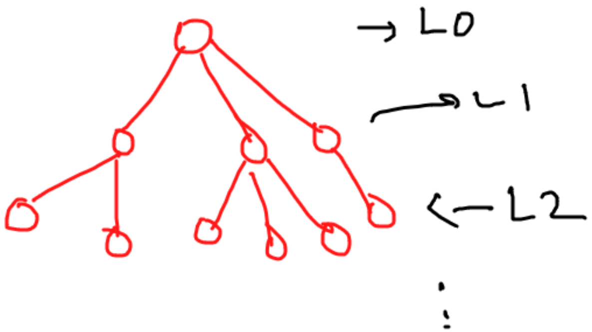
Tree

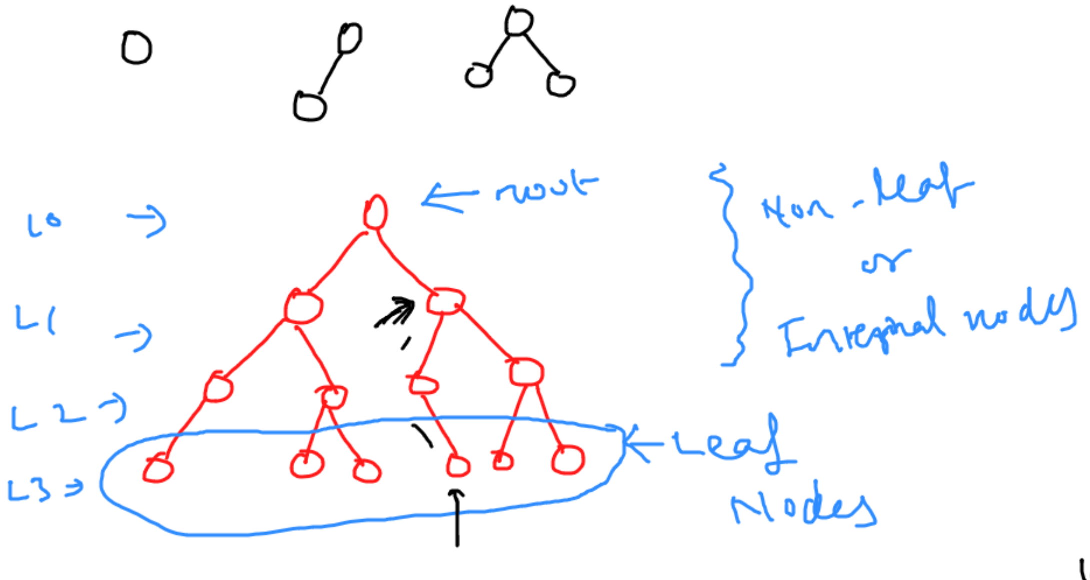


Parent Child Hierarchy.

Types

- Unary → max 1 child node
- Binary → max 2 child nodes
- Ternary → max 3 child nodes
- n-ary → max n child nodes

Binary Tree



Height & Depth

$D(v) = 1$

$H(v) = 2$

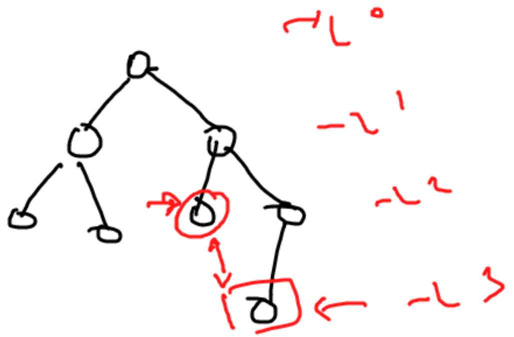
$H(v) \neq D(v)$

Depth

Length of path (no. of edges) from the root to the given node.

Height

Length of path (no. of edges) of the given node from the deepest node.



Height(root) = Height(Tree)

Depth(Deepest node) = Depth(Tree)

$\boxed{\text{Height(Tree)} = \text{Depth(Tree)}}$

(1) O ← H(T) = 0 (Zero)

1 node = root node

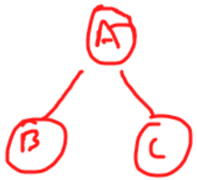
H(T) = 0

(2) No node → H(T) = -1

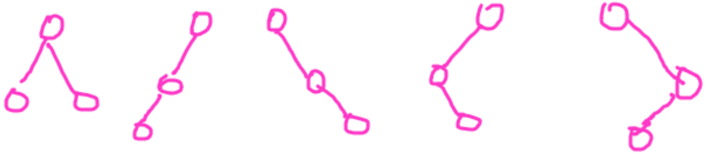
No. of unlabelled binary trees possible with 'n' nodes

$\boxed{N = 3}$

Unlabelled



Labelled



← for $N = 3$, $\Rightarrow$ 5 unlabelled Binary trees

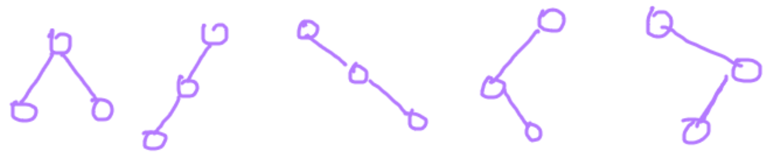
$$\text{No. of unlabelled binary trees} = \frac{{}^{2n}C_n}{n+1}$$

, where n = no. of nodes.

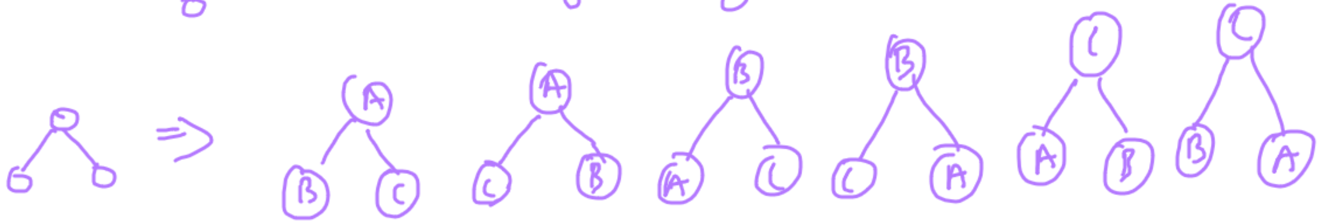
$$\frac{{}^{6}C_3}{4} = \frac{\cancel{6} \times 5 \times \cancel{4}}{\cancel{3 \times 2 \times 1} \times \cancel{4}} = 5$$

No. of labelled binary trees possible with 'n' nodes :-

N = 3



A, B, C



⇒ 6 BT possible for each structure

Total labelled binary tree = $6 * 5 = 30$

For each structure, labelled binary trees = $n!$

no. of unlabelled BTs (no. of structures) = $\frac{2nC_n}{n+1}$

$$\boxed{\text{Total labelled BTs} = n! * 2nC_n/(n+1)}$$

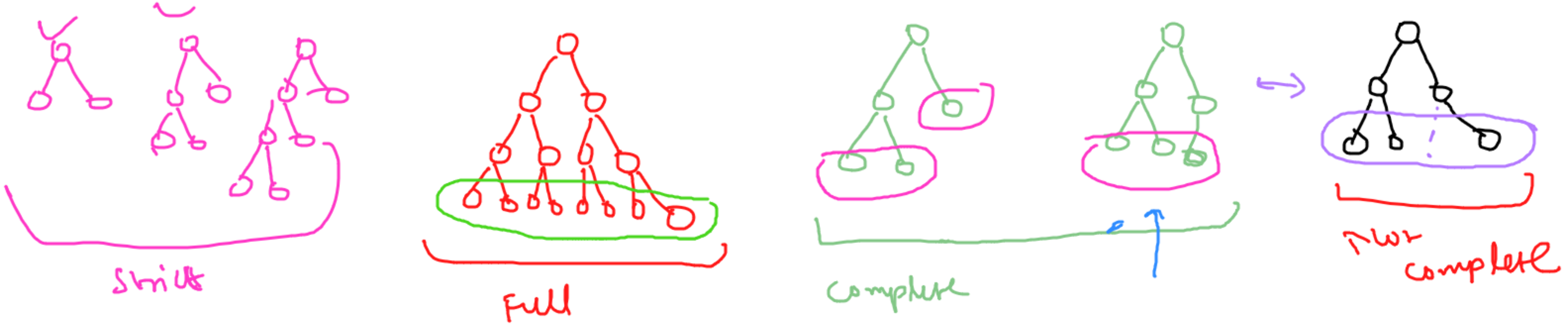
Types of Binary Trees

(1) Strict → either 0 child node or 2 child nodes

(2) Full → every node will have 2 child nodes, all leaf nodes are on same level

(3) Complete → All leaf nodes should be either on last level or last but one level.

nth level or n-1th level, where n → last level.



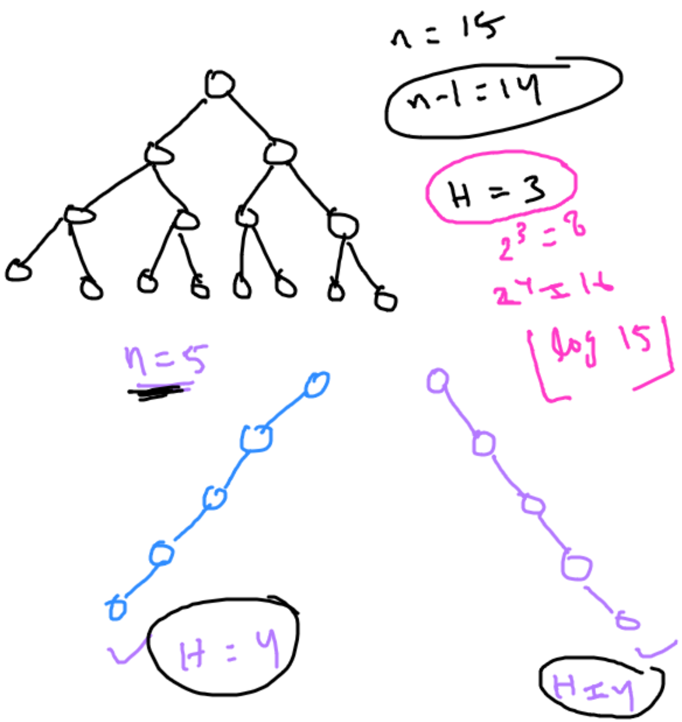
I. All full BT are complete BT but

II. All complete BT are not Full BT.

'N' nodes

Max Height = $N - 1$ (skewed BT)

Min Height = $\lfloor \log N \rfloor$ (Full BT)

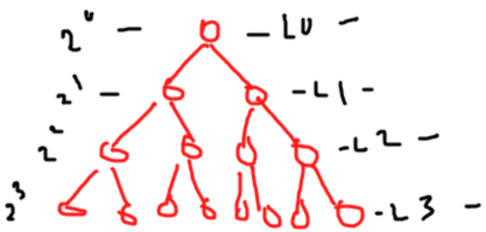


skewed Binary Tree

Given height H

max $N = 2^{h+1} - 1$ (Full)

min $N = H + 1$ (skewed)

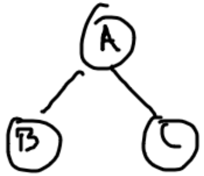


$$2^0 + 2^1 + 2^2 + 2^3 + \cdots + 2^h$$

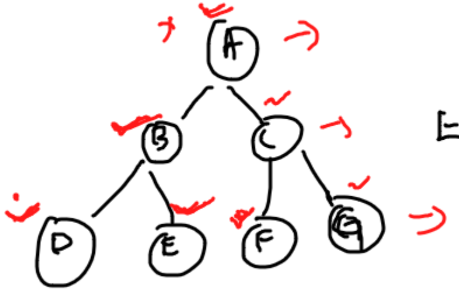
$$\Rightarrow \quad a\frac{(1-r^n)}{1-r} = \frac{1-2^{h+1}}{1-2} = 2^{h+1} - 1$$

Tree Traversals

(1) Inorder → L Root R
(2) Preorder → Root L R
(3) Post order → L R Root
(4) Level order → Levelwise

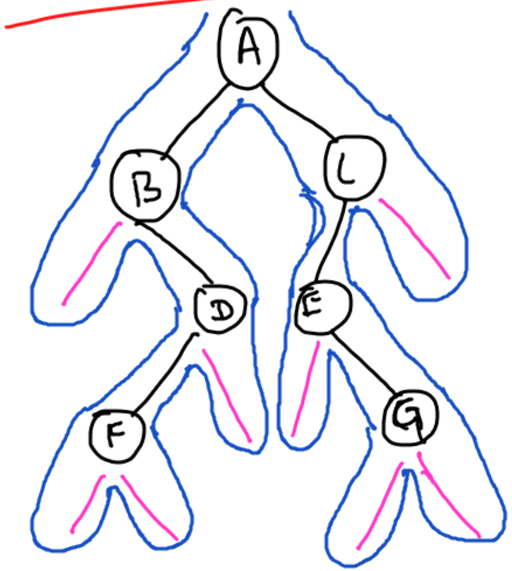


Inorder = B A C
Preorder = A B C
Postorder = B C A
Level order = A B C



Level order ⇒ A B C D E F G
Inorder ⇒ D B E A F C G
Postorder ⇒ D E B F G C A
Preorder ⇒ A B D E C F G

Example: Traversal



Preorder – A B D F C E G

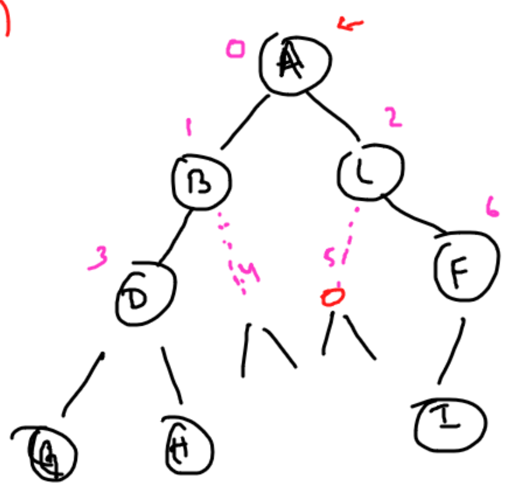
Inorder – B F D A E G C

Postorder – F D B G E C A

Tree Representation

(1) Array

(2) Linked List



A	B	C	D			F			
0	1	2	3	4	5	6	7	8	9

Given node index 'i'

index of

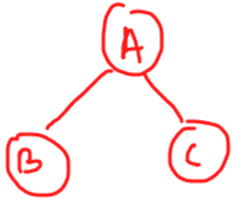
Left child = $2*i+1$

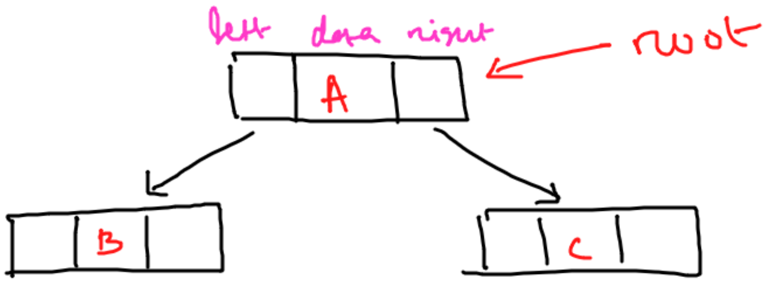
Right child = $2*i+2$

Given node index 'i'

Parent node index = $(i-1)/2$

Linked List Representation





Left subtree

Right subtree