Trees

-collection of nodes, where one of the node is the root node.

-siblings – children of same parent

-descendants – the children and the children of the children

-degree of a node- number of children (direct)

-lever (pochvat ot 1) we count nodes(number of nodes)

-height (we count edges) -pochvame on 0

-forest – a collection of trees is called a forest

Binary Tree

-tree of degree two (every node can have maximum 2 children) (deg(T) =2)

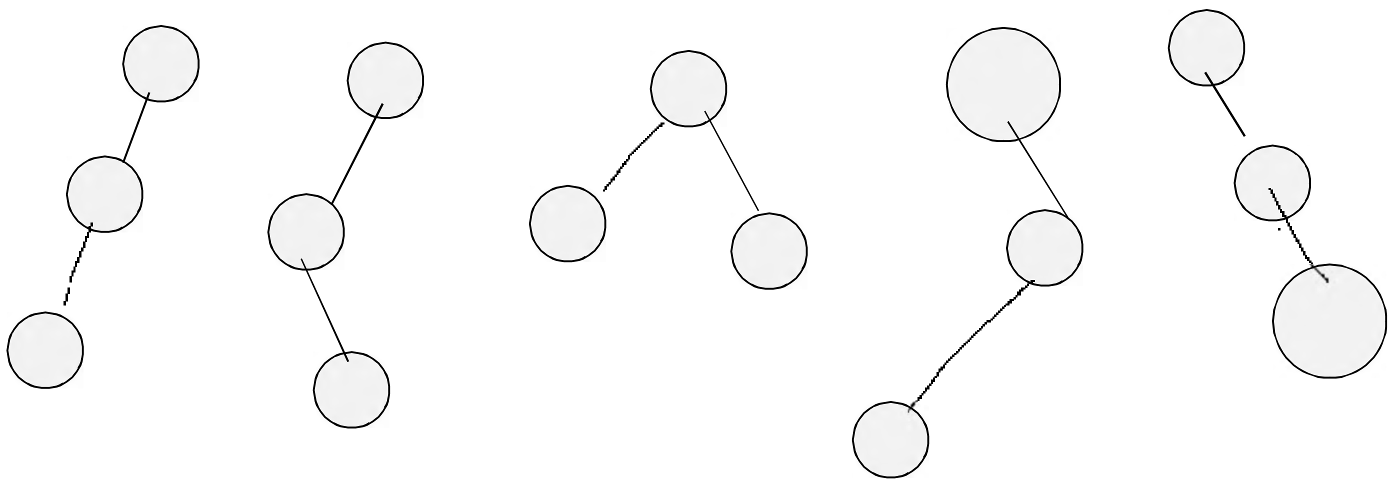
-we have left and right child

-every node can have {0, 1 ,2 } children

Number of binary trees for given set of nodes

If suppose some number of nodes are given like 3 nodes. Then, using 3 nodes how many different binary trees can we generate?

1.Unlabelled nodes(n =3) If we have 3 nodes, we can generate 5 trees (T(3)=5)



T(4)=14

Catalan' Formula: $T(n) = 2nC_n / n + 1$ (gives how many different trees)

T(5)=2*5C5/6=$10C_5$/6= 42

Maximum height possible is height

Number of threes with maximum height with 3 nodes are 4= 2^2

N=4 -> 8= 2^3

N= n -> 2^{n-1}

One more formula for Catalan:

N	0	1	2	3	4	5	6
T(n)= 2nCn/n+1	1	1	2	5	14	42	132

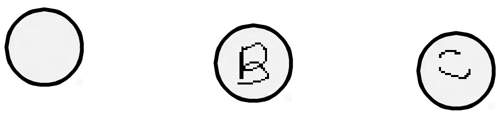
T(6)=1*42+1*14+2*5+5*2+14*1+42*1=123

T(6)=T(0)*T(5)+T(1)*T(4)+T(2)*T(3)+T(3)*T(2)+T(4)*T(1)+T(5)*T(0)

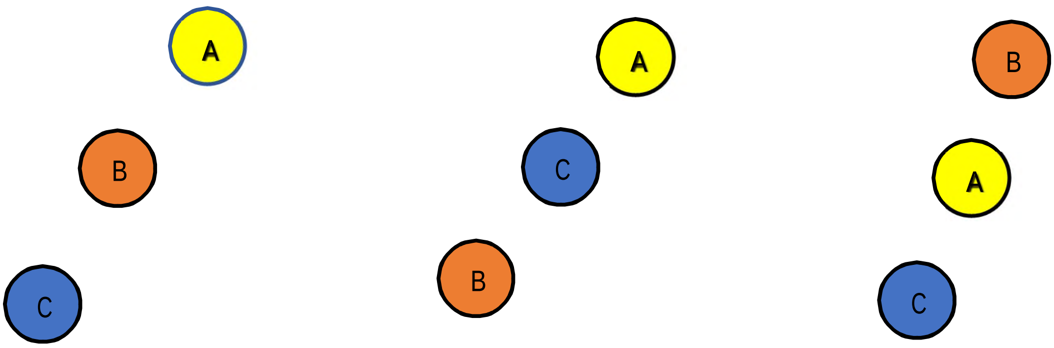
T(n)= $\sum_{i=1}^{n} \quad T(i-1)^{\wedge} * T(n-i)$

2. Labelled nodes

If we have labelled nodes, how many binary trees can we generate?



Again we have five trees, but for each tree we can have 3! different trees.



A->B->C B->C->A

A->C->B C->A->B T(n)=2nCn/n+1*n!

B->A->C C->B->A shapes filling

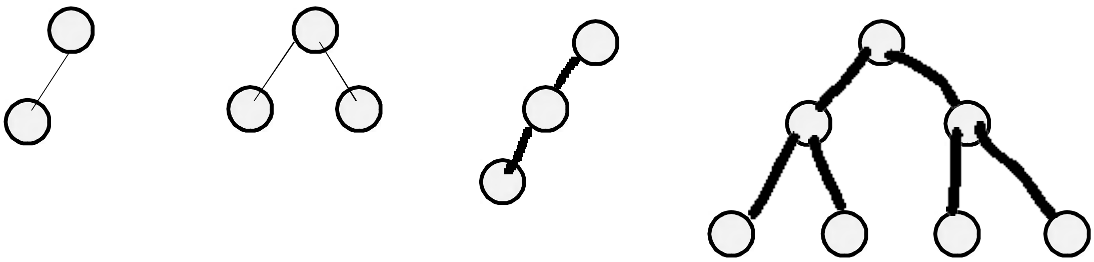
Height vs Nodes in Binary Tree

If we know the height of a binary tree, what the minimum and maximum number of nodes could be?

If nodes are given what could be minimum height and what could be maximum height?

Height= 1 Height=2

Min(2) max (3) min(3) max(7)



<u>Minimum nodes</u> formula=height + 1 (n=h+1)

<u>Maximum node</u>: $1+2+2^2+2^3=15$ (this is for a tree with height 3)

The given formula is GP series: $a+ar+ar^2+......+ar^k=a(r^{k+1}-1)/r-1$

$1+2+2^2+...2^h=2^{h+1}-1$ -> formula for maximum number of nodes a=1,r=2

If nodes are given and we want to calculate the height:

maxHeight h=n-1;

minHeight $h=\log_2(n+1)-1$

<u>Relationship between leaf nodes and non-leaf nodes</u>

There is a relationship between nodes with degree 2 and degree 0.

(deg(2)-number of nodes with degree 2)

deg(0)=deg(2)+1<-always true for binary trees

<u>Strict(Complete) Binary Tree</u>

In a strict binary tree a node can have either 0 or 2 children

In a complete bin tree a node can have either 0 or 2 children , and when represented as array there should not be any blank space between elements.

<u>Height in Strict Binary Tree</u>

If height is given:

Min nodes: n=2h+1 (we start from the bottom of the tree and we count , + 1 because of the root)

Formula form max node is the same: $n=2^{h+1}$ -1

Leaf nodes are 1 + non-leaf nodes <-always true in strict binary trees

n-ary Trees

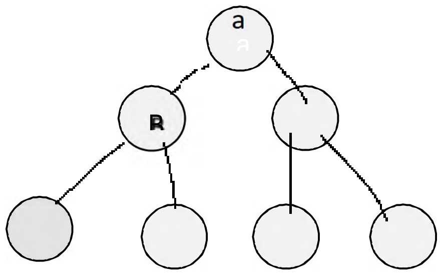
3-ary trees-> {0,1,2,3}

4-ary trees->(0,1,2,3,4} , degrees of nodes

Representation of Binary Trees

1.Array representation

ABCDEFG



T

A	B	C	D	E	F	G
1	2	3	4	5	6	7

element	index	Left child	Right child
A	1	2	3
B	2	4	5
C	3	6	7
	i	2*i	2*i+1

Parent->i/2(floor)

2.Linked representation

```
Struct Node
{
        Node *left;
        T data;
        Node *right;
}
```

left	data	right

If there are n nodes, we always have n+1 null pointers in the leafs.

Full vs Complete Binary

Full-a binary tree of height h having maximum number of nodes

If represented in an array all cells are filled with values.

Complete- no blank spaces in between the array, but at the end we can have free cells.

A complete binary tree of height h will be a full binary tree up to height h-1 and in the last level elements will be completed form left to right without having blank space. We say complete if is suitable for array representation. full binary tree->complete tree.