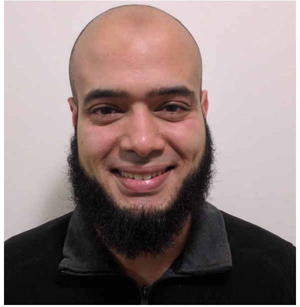
Data Structures

Binary Tree Formulas

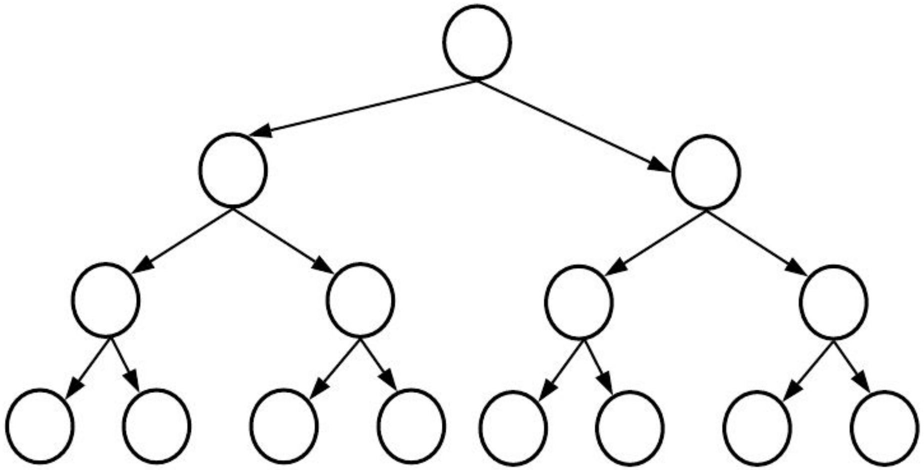
Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Perfect Tree: Find the # of nodes from the height!

- Each level (0-based) has 2^{level} nodes (*2 * previous level nodes*).
- For N levels: $2^0 + 2^1 + 2^2 + 2^3 + \ldots\ldots + 2^{level} = 2^{levels} - 1 = 2^{h+1} - 1$ nodes



Leve 0: 1 node (2^0)

Leve 1: 2 node (2^1)

Leve 2: 2*2 = 4 nodes (2^2)

Leve 3: 4*2 = 8 nodes (2^3)

Perfect Tree: Find the # of nodes from the height!

- We can derive this mathematically
- First recall the **power rule for logarithms**

$$\log_b(M^n) = n\log_b M$$

- Also recall $\mathbf{\log_2 2 = 1}$

$$n = 2^{h+1} - 1$$

$$n + 1 = 2^{h+1}$$

$$\lg(n+1) = h + 1$$

$$h = \lg(n+1) - 1$$

Facts

- We can derive *upper and lower* bounds of a normal tree from the perfect tree
  - This is because we know every level is complete; i.e. you CAN'T have fewer levels!
- **In any binary tree:**
- Each level has a max of 2^h nodes
- For L levels,   No more than $2^L - 1$ nodes
- For N nodes, the min # of levels is: ceil(log(N+1))
  - 1 Node ⇒ 1, 3 Nodes ⇒ 2, 7 Nodes ⇒ 3, 15 Nodes ⇒ 4   (these are in perfect cases)
- For M leaves, the min # of levels is: ceil(log M) + 1
  - 1 leaves ⇒ 1, 2 leaves ⇒ 2, 4 leaves ⇒ 3, 8 leaves ⇒ 4, 16 leaves ⇒ 5

The logarithm

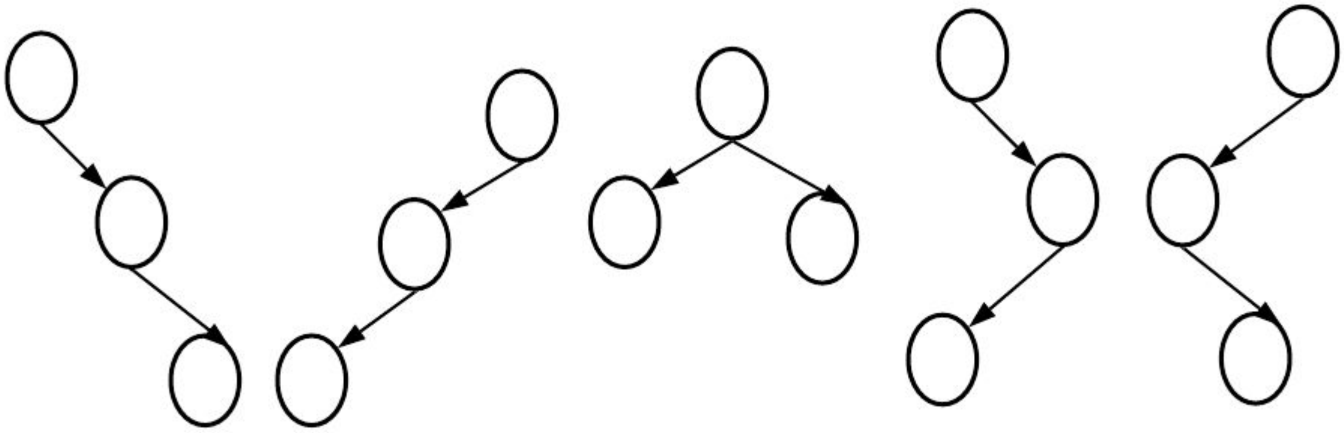
- Observe how the log has a very small value
- This means, we can have a tree of 1 million nodes, but its height can be:
  - ~ 1 million with degenerate tree
  - ~20 only if it is perfect or complete
- In balanced trees (e.g. AVL / red-black), we put constraints that help us have such controlled height , rather than allowing the uncontrolled chaos of a very deep and unbalanced tree

		Log	Number
2^0	= 1	0	1
2^1	= 2	1	2
2^2	= 4	2	4
2^3	= 8	3	8
2^4	= 16	4	16
2^5	= 32	5	32
2^6	= 64	6	64
2^7	= 128	7	128
2^8	= 256	8	256
2^9	= 512	9	512
2^{10}	= 1024	10	1024
2^{11}	= 2048	11	2048
2^{12}	= 4096	12	4096
2^{13}	= 8192	13	8192
2^{14}	= 16384	14	16384
2^{15}	= 32768	15	32768
2^{16}	= 65536	16	65536
2^{17}	= 131072	17	131072
2^{18}	= 262144	18	262144
2^{19}	= 524288	19	524288
2^{20}	= 1048576	20	1048576

Img src

How many unlabeled binary trees of 3 nodes?

- All 5 trees below can be drawn using just 3 nodes
- So, in general, for any number of nodes n, how many unlabeled binary trees are there?



How many **unlabeled** binary trees of n nodes?

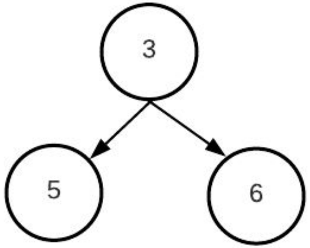
- The answer is a very interesting mathematical number!
- The Catalan Number (wiki has a lot of facts)
  - You don’t need to know why

$$C_n = \frac{1}{n+1}\binom{2n}{n} = \frac{(2n)!}{(n+1)!\,n!}$$

How many labeled binary trees of n nodes?

- Given a single tree of n nodes, we can label it in n! ways!
- So the answer is **Catlan**(n) * n!

$$\frac{1}{n+1}\binom{2n}{n} \times n! = \frac{(2n)!}{(n+1)!}$$



"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."