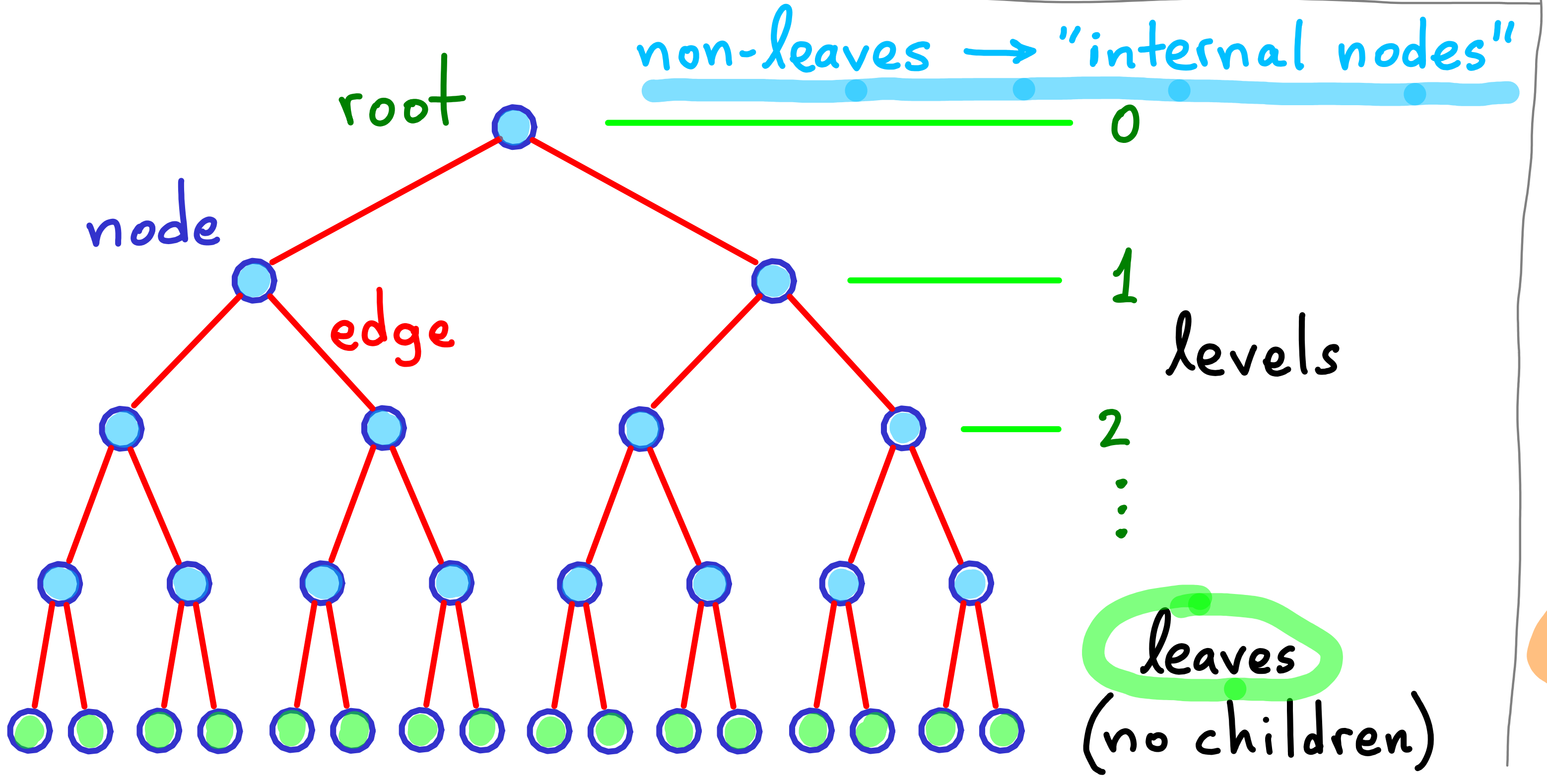
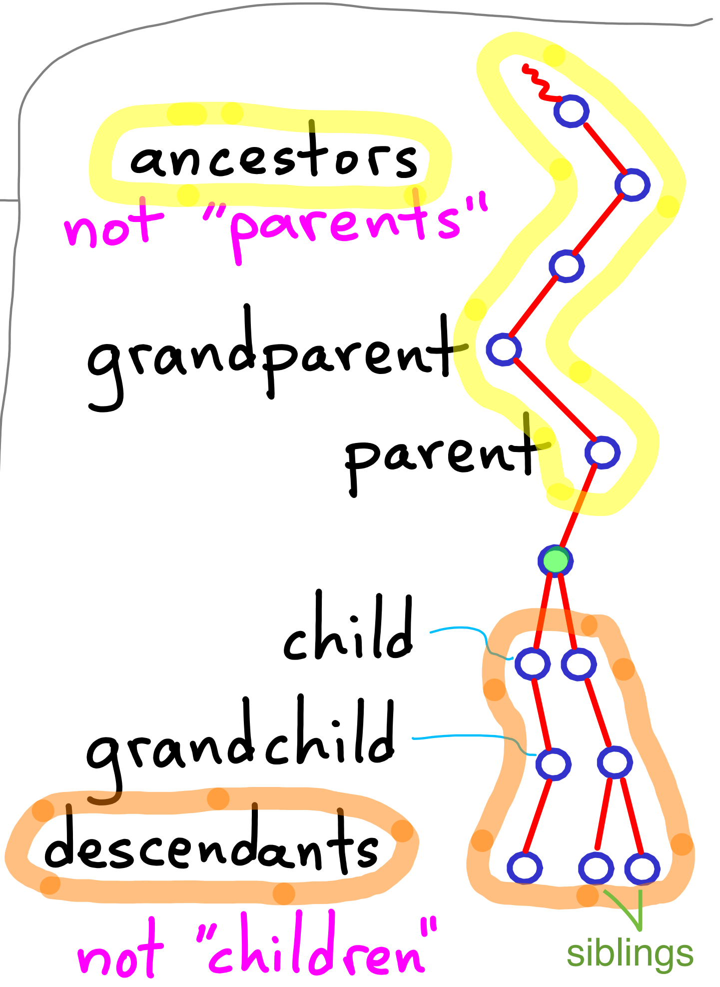
BINARY TREES

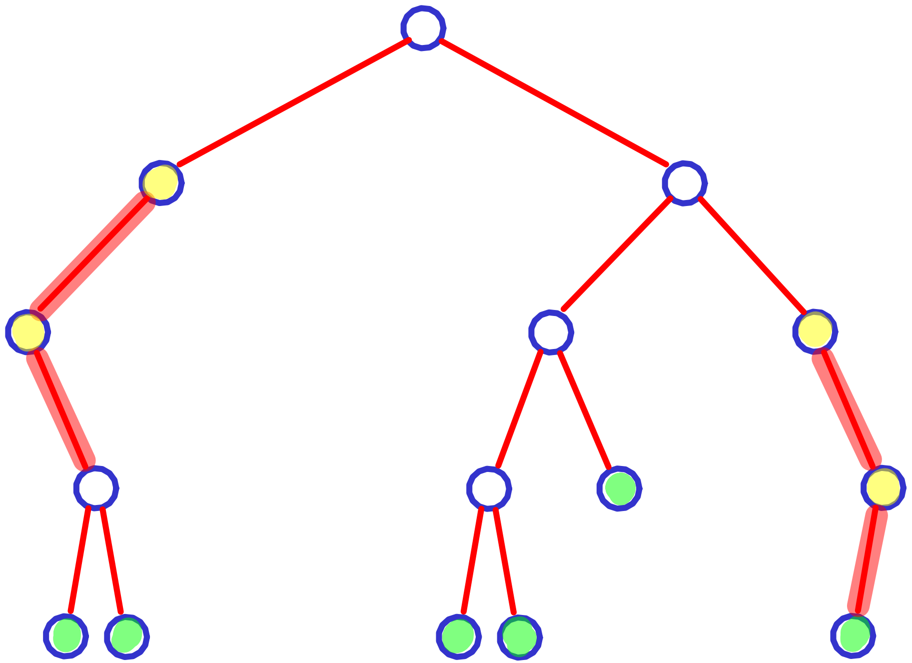
- if non-empty, ∃ root at level 0.
- every node at level i connects to ≤ 2 children at level $i+1$.
- every non-root node has 1 parent





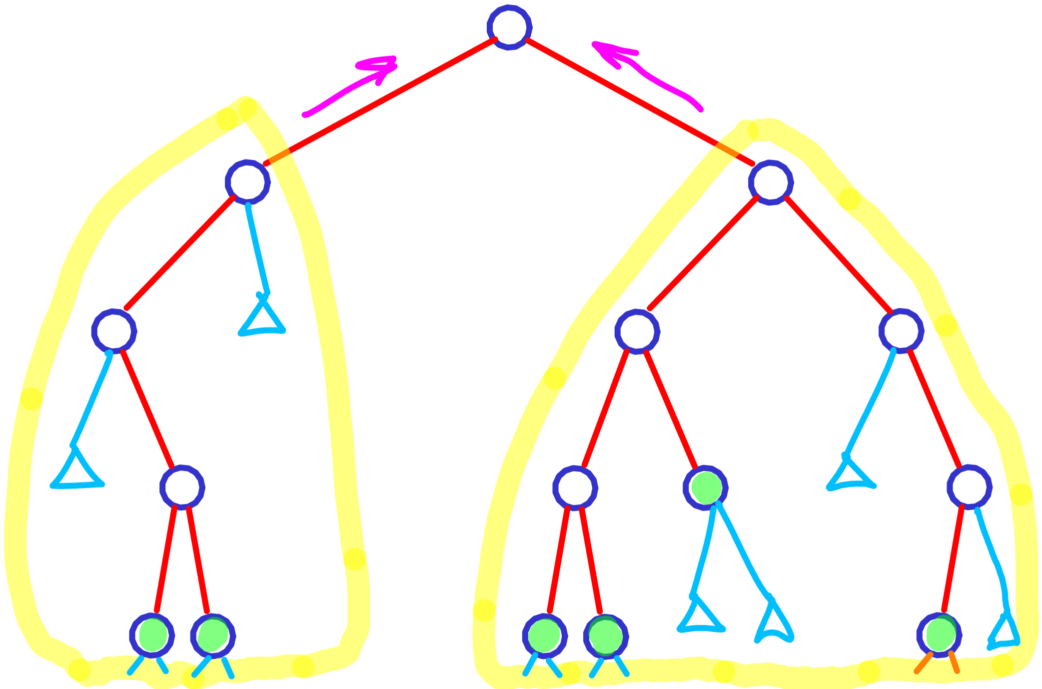
BINARY TREES

- if non-empty, $\exists$ root at level 0.
- every node at level i connects to ≤ 2 children at level $i+1$.
- every non-root node has 1 parent



BINARY TREES

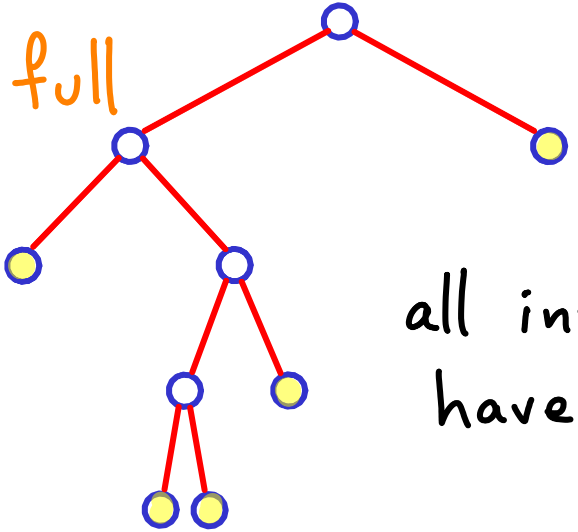
- if non-empty, ∃ root at level 0.
- every node at level i connects to 2 subtrees at level $i+1$.
- every subtree root has 1 parent
  (except global root)



[subtrees can be empty]

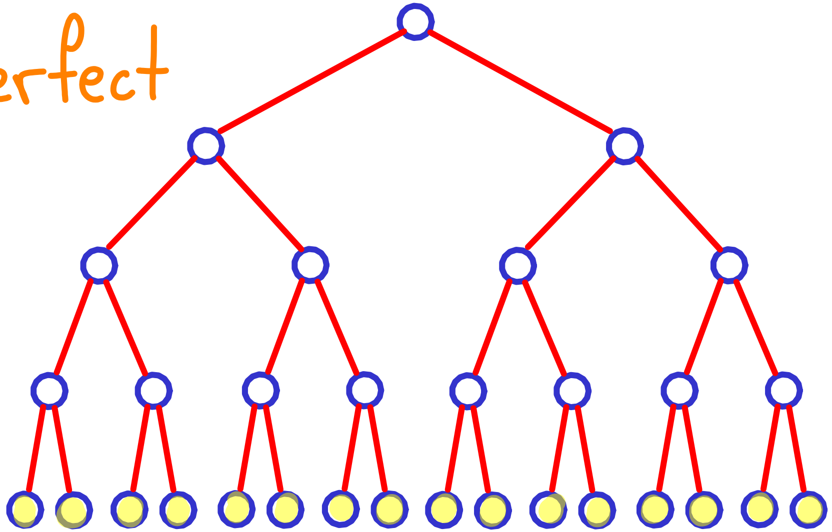
leaves : 2 empty subtrees

full



all internal nodes have 2 children

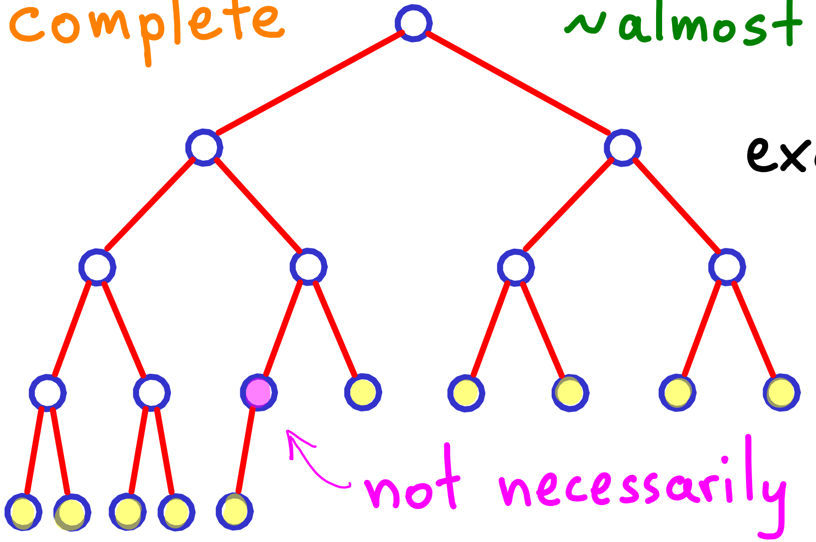
perfect



full & all leaves on same level

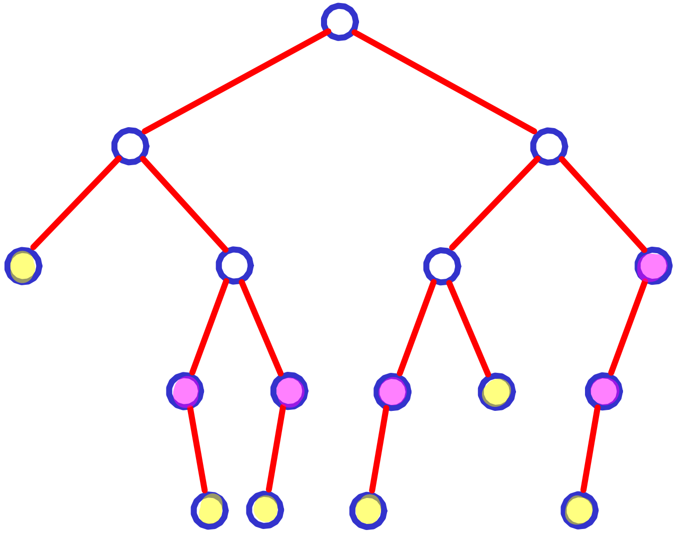
complete

~almost perfect

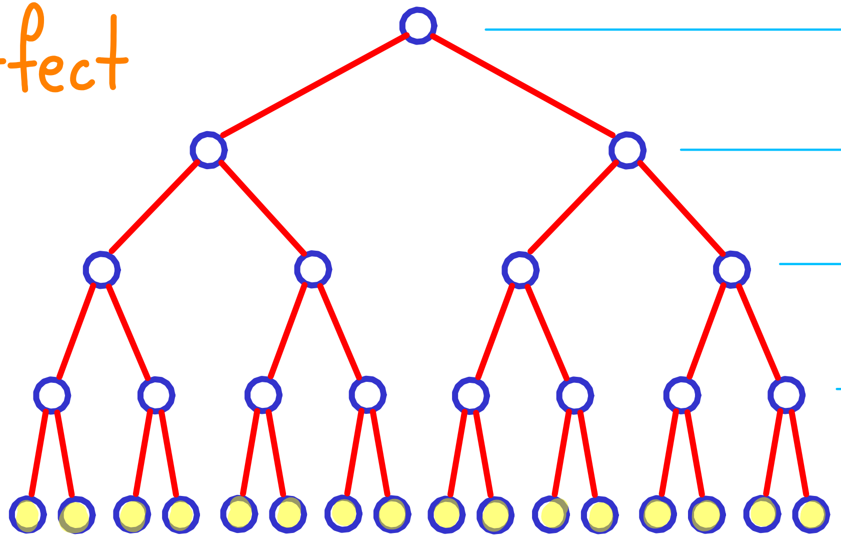


exception: last level (partially) filled in from left to right

not necessarily a full tree



perfect



1 $\quad 2^0$

2 $\quad 2^1$

4 $\quad 2^2$

8 $\quad 2^3$

16 $\quad 2^4 = 2^{L-1} = \#\text{leaves} = \ell$

#nodes per level ?

↳ 2^i for level i

$i : \{0, 1, 2, \ldots, L-1\}$

$\#\text{levels} = L$

Let n = total #nodes. notice: n is odd

- how many levels ?

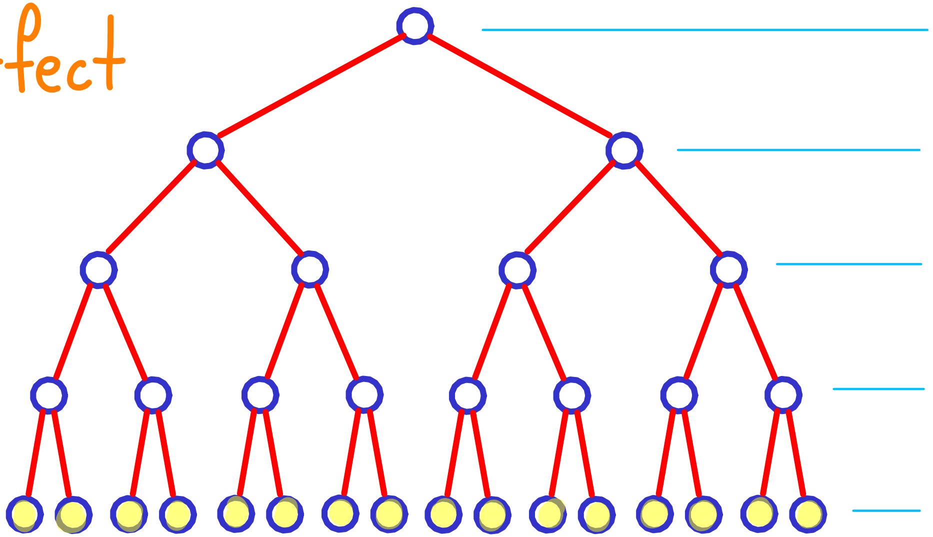
$$n = \sum_{i=0}^{L-1} 2^i = 2^L - 1$$

For $L \geq 2$: Hypothesis: $\sum_{i=0}^{L-2} 2^i = 2^{L-1} - 1$

Base case trivial for $L = 1$

$$\sum_{i=0}^{L-1} 2^i = 2^{L-1} + \sum_{i=0}^{L-2} 2^i = 2^{L-1} + 2^{L-1} - 1$$

perfect



1 2^0

2 2^1

4 2^2

8 2^3

16 $2^4 = 2^{L-1}$ = #leaves = ℓ

#nodes per level ?

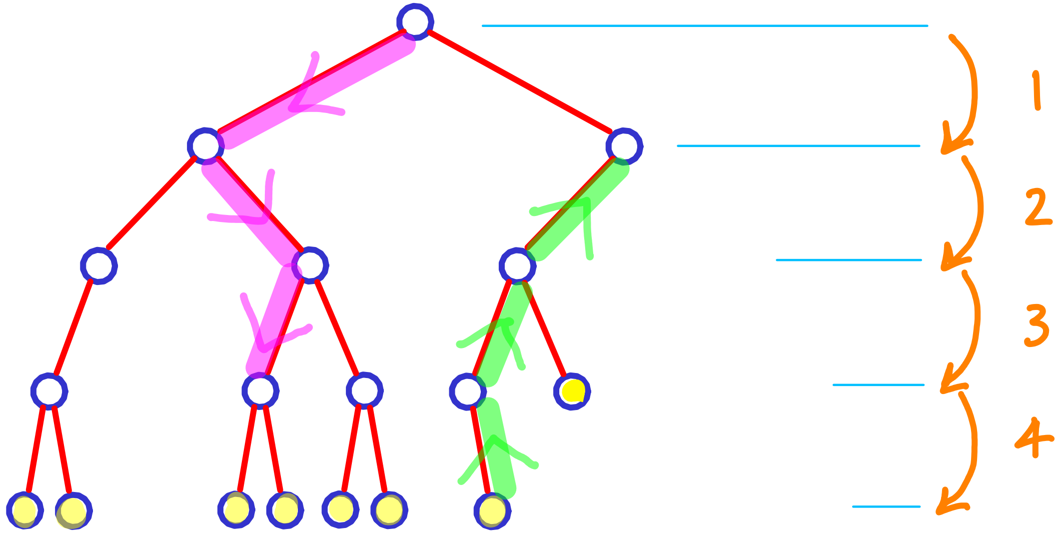
↳ 2^i for level i

$i : \{0, 1, 2, \dots, L-1\}$

#levels = L

Let n = total #nodes. notice: n is odd

- how many levels? $n = \sum_{i=0}^{L-1} 2^i = 2^L - 1 \rightarrow L = \log_2(n+1)$
- how many leaves? $n = 2 \cdot 2^{L-1} - 1 = 2\ell - 1 \rightarrow \ell = \lceil \frac{n}{2} \rceil = \frac{n+1}{2}$
- how many internal nodes? $\lfloor \frac{n}{2} \rfloor = \frac{n-1}{2}$ (n − #leaves)



depth of tree
⇕
height of tree : h

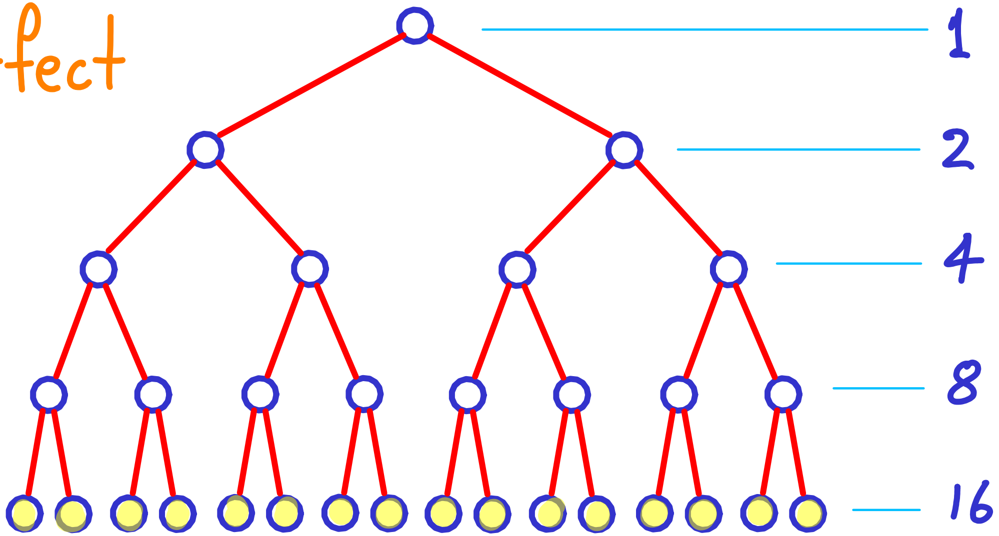
h = number of "steps" (edges) from root to deepest leaf

lowest level

- depth of node = #steps from root
- height of node = #steps from deepest descendant
- definitions hold for any tree

$\boxed{h = L - 1}$

perfect



2^0

2^1

2^2

2^3

$2^4 = 2^h$ = #leaves = ℓ

#nodes per level ?

↳ 2^i for level i

$i : \{0, 1, 2, \ldots, L-1\}$

#levels = L

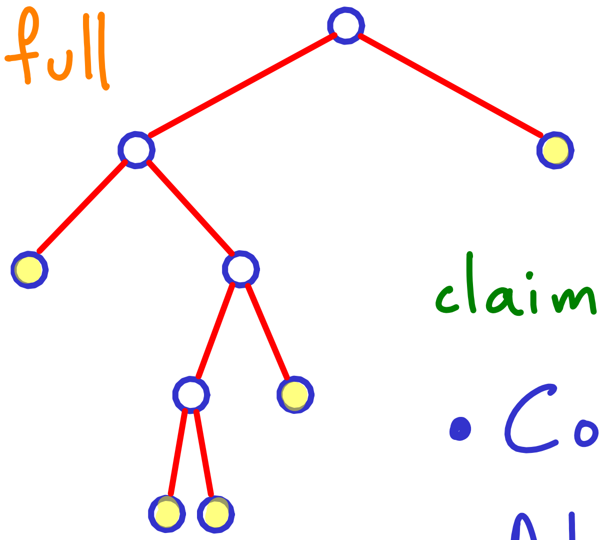
$\boxed{h = L-1}$

Let n = total #nodes. notice: n is odd

- how many levels? $n = \sum_{i=0}^{h} 2^i = 2^{h+1} - 1 \rightarrow h+1 = \log_2(n+1)$
- how many leaves? $n = 2 \cdot 2^{L-1} - 1 = 2\ell - 1 \rightarrow \ell = \left\lceil \frac{n}{2} \right\rceil = \frac{n+1}{2}$
- how many internal nodes? $\left\lfloor \frac{n}{2} \right\rfloor = \frac{n-1}{2}$ (n − #leaves)

	Exact summary	Approximate summary
ℓ vs h	$\ell = 2^h$ $h = \log_2 \ell$	$\leftarrow$ $\leftarrow$
ℓ vs n	$\ell = \lceil \frac{n}{2} \rceil = \frac{n+1}{2}$ $\ell = \#\text{internal nodes} + 1$	within ± 1 $\ell \approx \frac{n}{2}$ $\ell \approx \#\text{internal nodes}$
h vs n	$h+1 = \log_2(n+1)$ $n = 2^{h+1} - 1$	$h \approx \log_2 n$ $h = \Theta(\log n)$ within factor of 2 $n \approx 2^h$ $n = \Theta(2^h)$

full



ℓ = # internal nodes + 1 = $i+1$

Induction on i

To prove for $i \geq 1$, hypothesis:
claim holds for full trees with $< i$ internal nodes.

base case trivial

$\ell = 1 \quad i = 0$

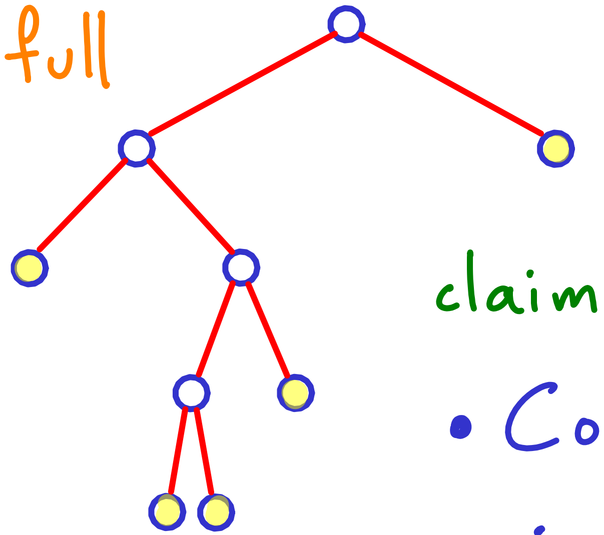
$n = 1$

- Consider any full tree with i internal nodes.
- At lowest level, $\exists$ leaf x.
- x has a sibling y. (parent p of x has 2 children)
- Remove x & y : p becomes a leaf.
- New tree has $i-1$ internal nodes & is full.
  $\hookrightarrow$ has i leaves by hypothesis.
- Put x & y back. $+2$ leaves
- $p \rightarrow$ internal : -1 leaf

$\left.\begin{array}{l} +2 \text{ leaves} \\ -1 \text{ leaf} \end{array}\right\}$ $i+1$ leaves

$\square$

full



base case trivial

$\circ$

$\ell=1 \quad i=0$

$n=1$

$\ell = \underline{\#\text{internal nodes}} + 1 = i+1$

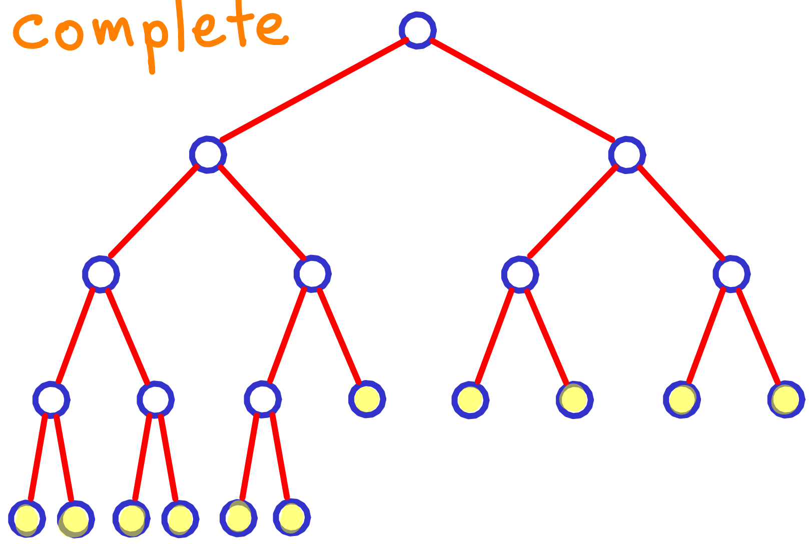
Induction on i

ALTERNATE PROOF

To prove for $i \geq 1$, hypothesis: claim holds for full trees with $< i$ internal nodes.

- Consider any full tree with i internal nodes.
- $i \geq 1$ so the root has 2 non-empty subtrees.
- Left subtree has x internal nodes, right subtree has y.
- $x+y=i-1$ (excluding root) $\rightarrow$ $x \leq i-1$ & $y \leq i-1$
- Subtrees are full, so by hypothesis, left subtree has $x+1$ leaves & right subtree has $y+1$.
- Combined, they have $x+y+2 = (i-1)+2 = i+1$ leaves $\square$

complete

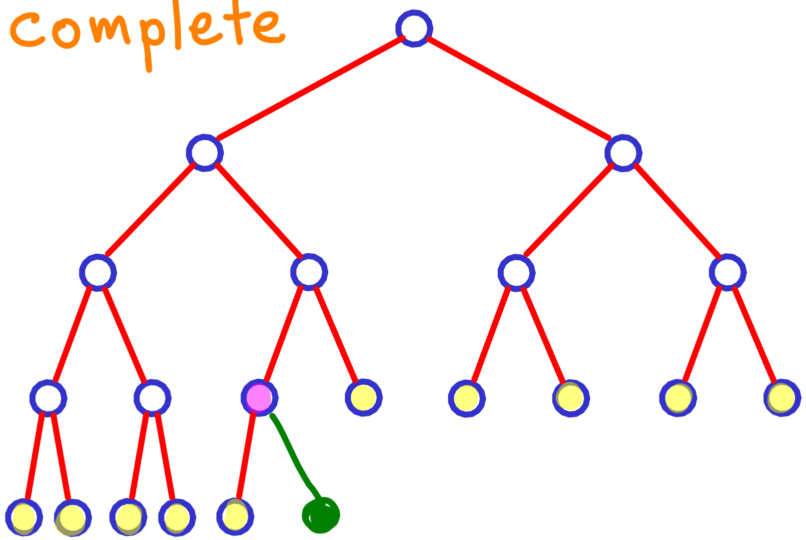


n is odd, tree is full.

$\ell = \#\text{internal nodes} + 1$

$$\ell = \left\lceil \frac{n}{2} \right\rceil = \frac{n+1}{2}$$

complete



n is even, tree is not full.

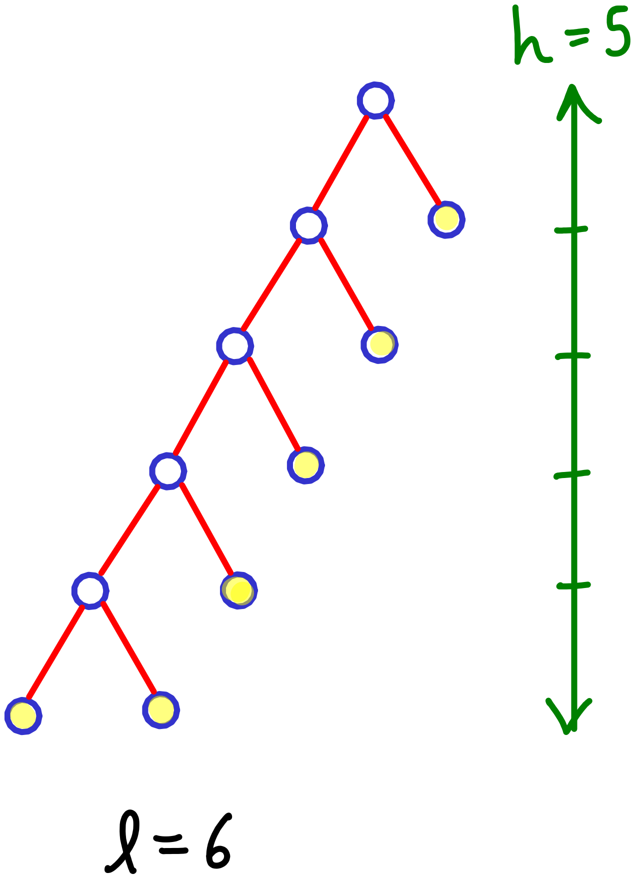
- can add one leaf...
- new tree has n+1 nodes & is full

$$\ell + 1 = \frac{(n+1)+1}{2} \rightarrow \ell = \frac{n}{2} = \left\lceil \frac{n}{2} \right\rceil$$

$\ell = \#\text{internal nodes}$

full

can get $\ell = h+1$



$\ell \neq 2^h$

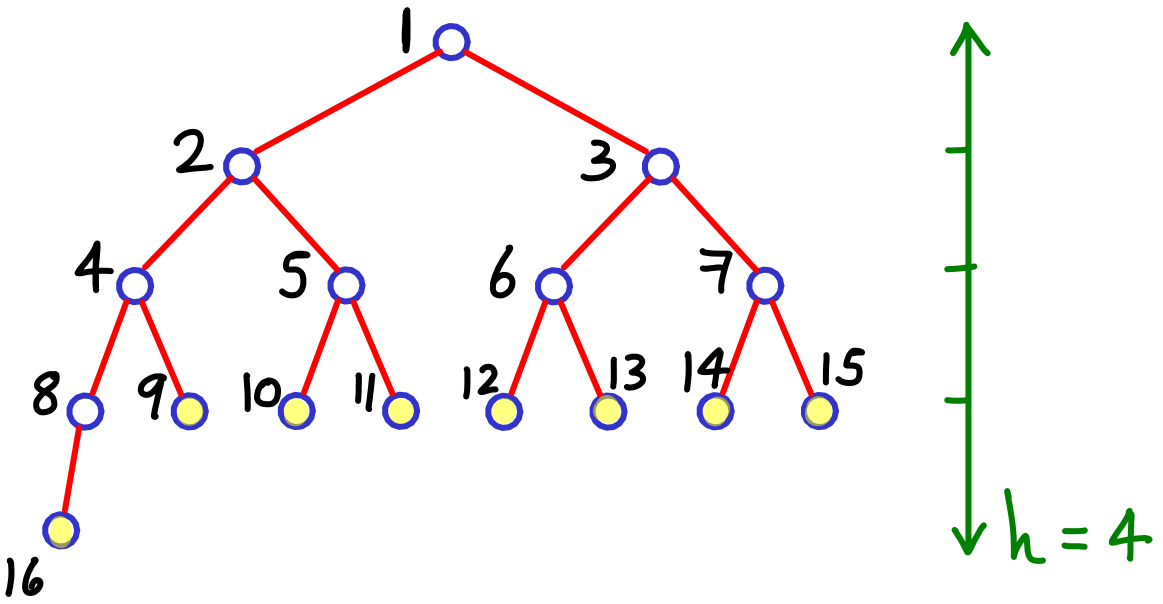
$h \neq \log_2 \ell$

not even $h = \Theta(\log \ell)$

complete

n nodes → what is h?

Intuition: h doesn't change much compared to perfect tree



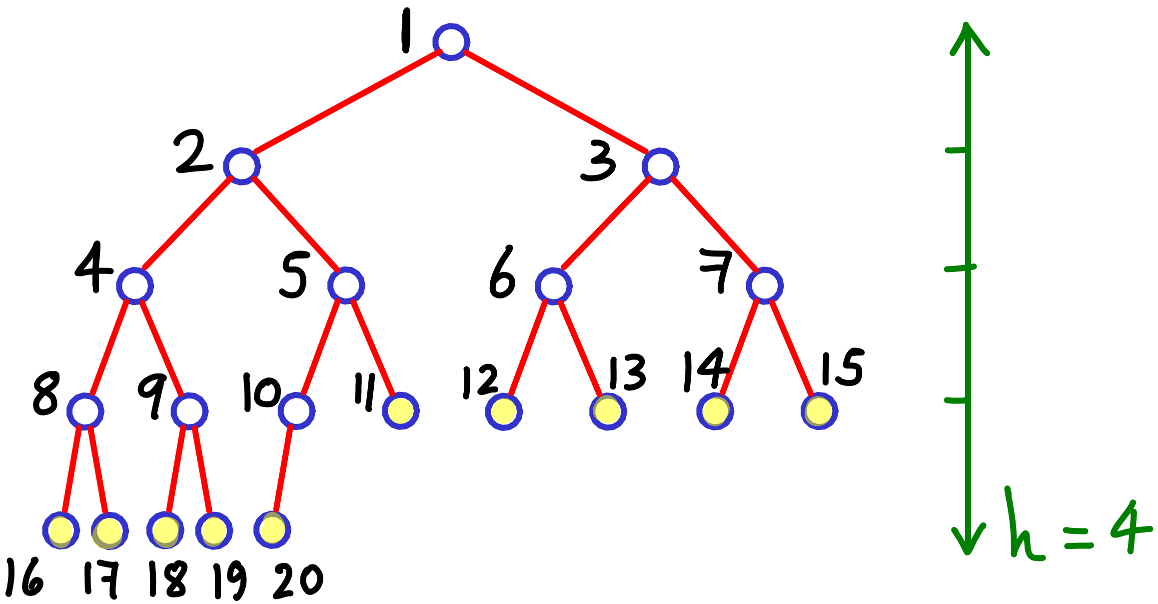
If n is a power of 2, we get:

$$h = \log_2 n$$

complete

n nodes → what is h?

Intuition: h doesn't change much compared to perfect tree



for $2^h < n < 2^{h+1}$

$$h < \log_2 n < h+1$$

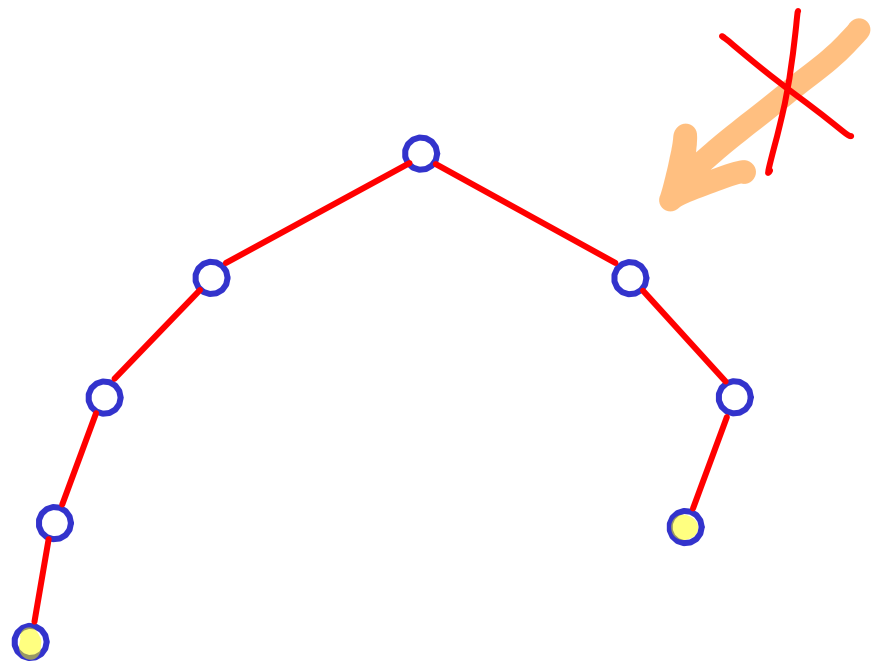
$$h = \lfloor \log_2 n \rfloor$$

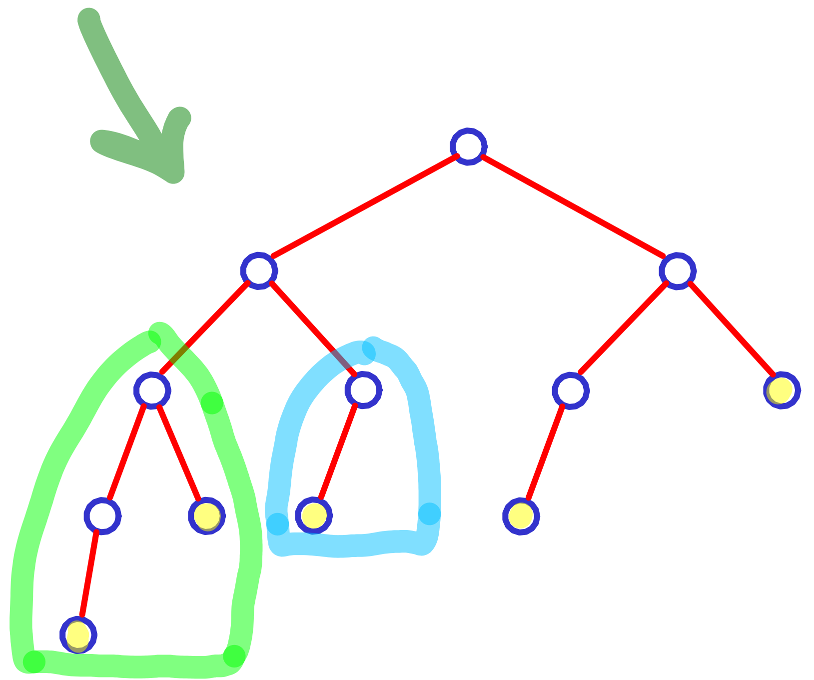
balanced binary trees

no unique definition

common: depth of left & right subtrees differ by ≤ 1

(recursively)





balanced binary trees

no unique definition

common: depth of left & right subtrees differ by ≤ 1
(recursively)

implies

doesn't imply

also common: depth is $O(\log n)$