Tree Dynamic Programming

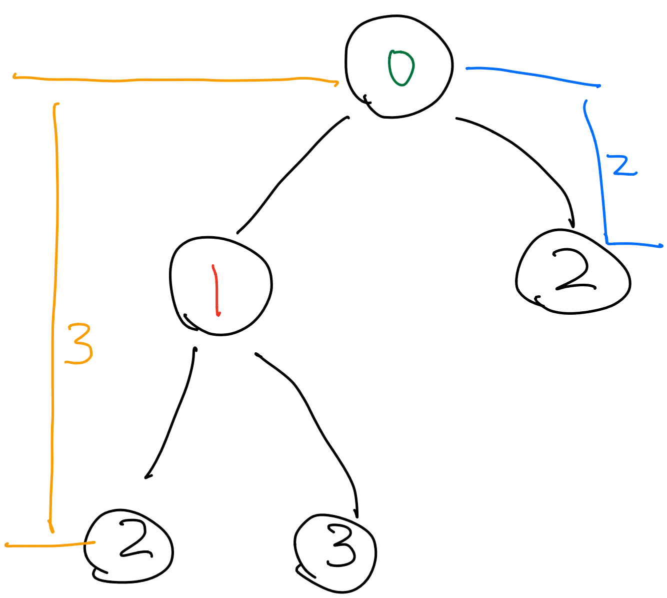
Setting: A tree roots at 0. ⇒ Compute dp[i]

Trick: for each node i, we store two values

highest[i]

secondhighest[i]

Ex 1.



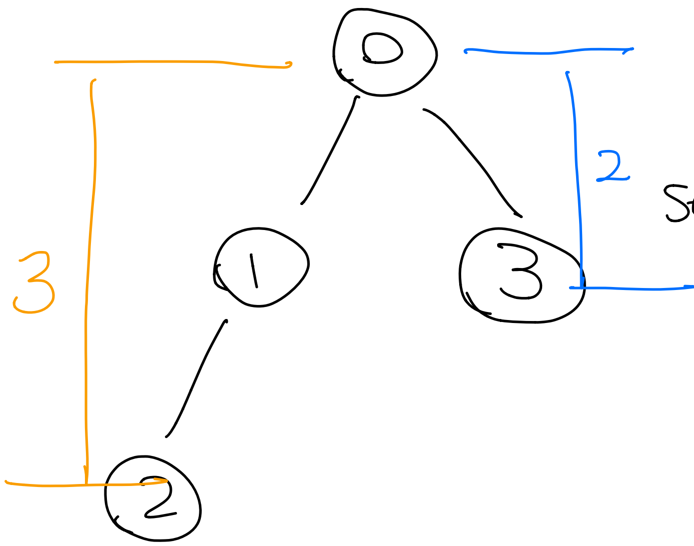
highest[0] = 3

secondhighest[0] = 2 (not 3!)

highest[1] = 2

secondhighest[1] = 2

Ex 2.



highest[0] = 3

second highest[0] = 2

Intuition: $O(n)$

Let dp[i] be the height of a tree rooted at i.

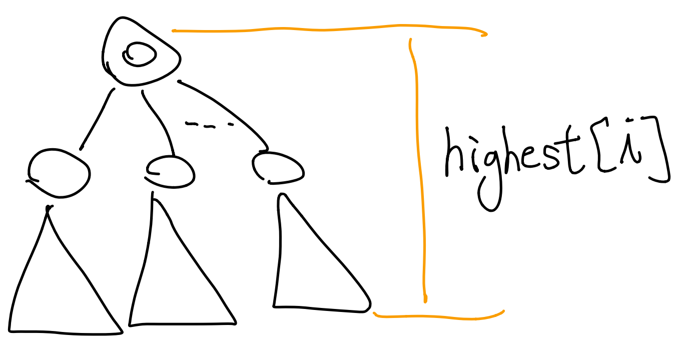
And assume we have already computed

highest[i] $i \in [0, n)$

second highest[i]

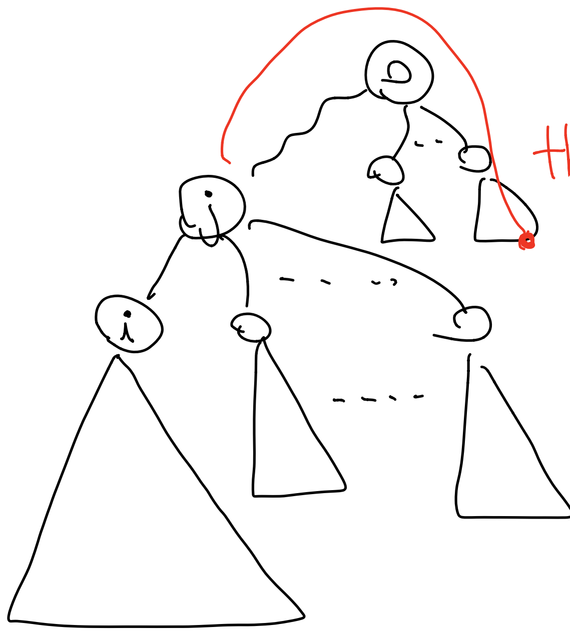
in advance by DFS

It will be easy to know dp[0]



How about dp[i] ?

Assume we know the longest path from j to other nodes other than j's descendants

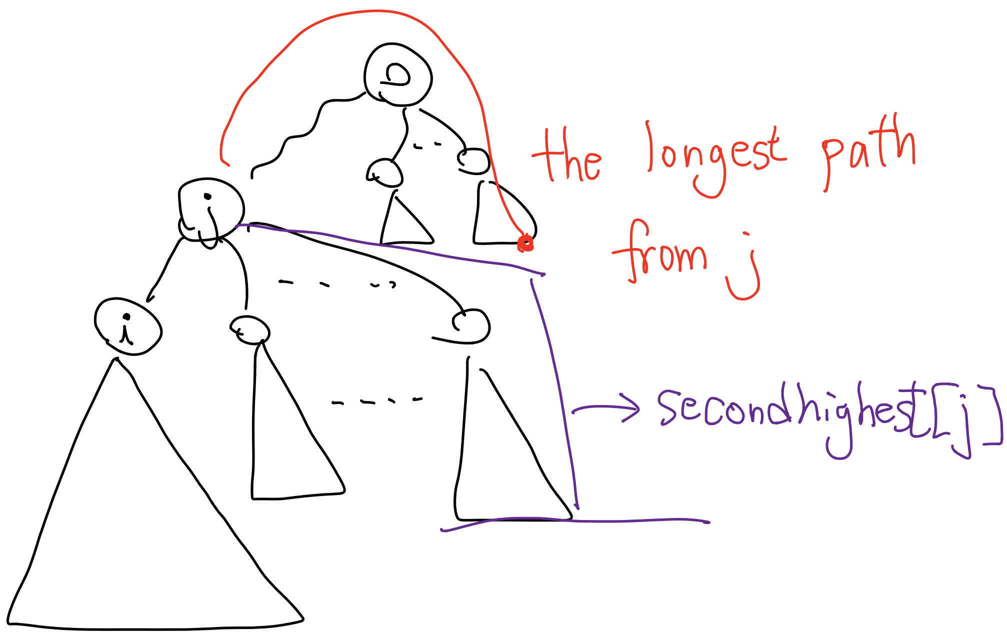


the longest path from j

It will be easy to see:

dp[j] = max(highest[j], the longest path from j)

How about the longest path from i ?

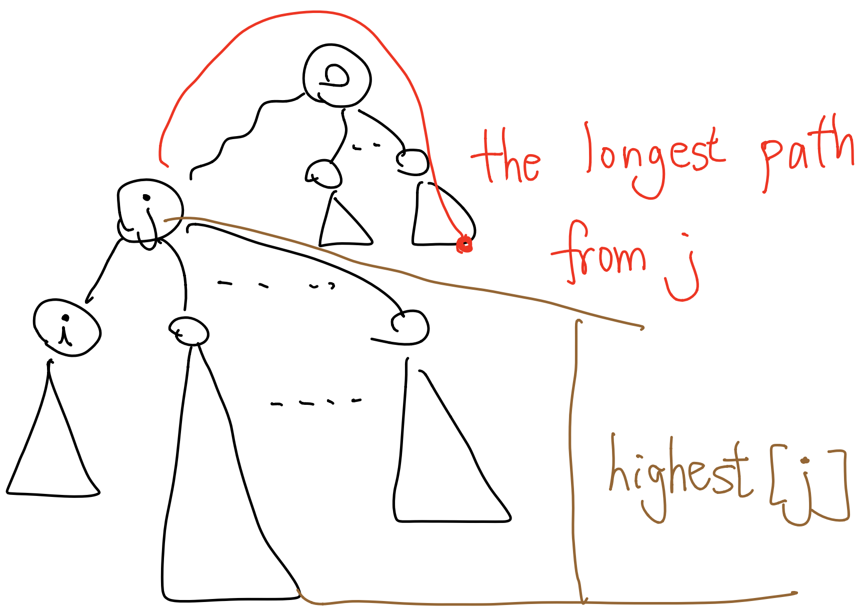


It turns out it is easy to see:

① If highest[i]+1 == highest[j]:

the longest path from i =

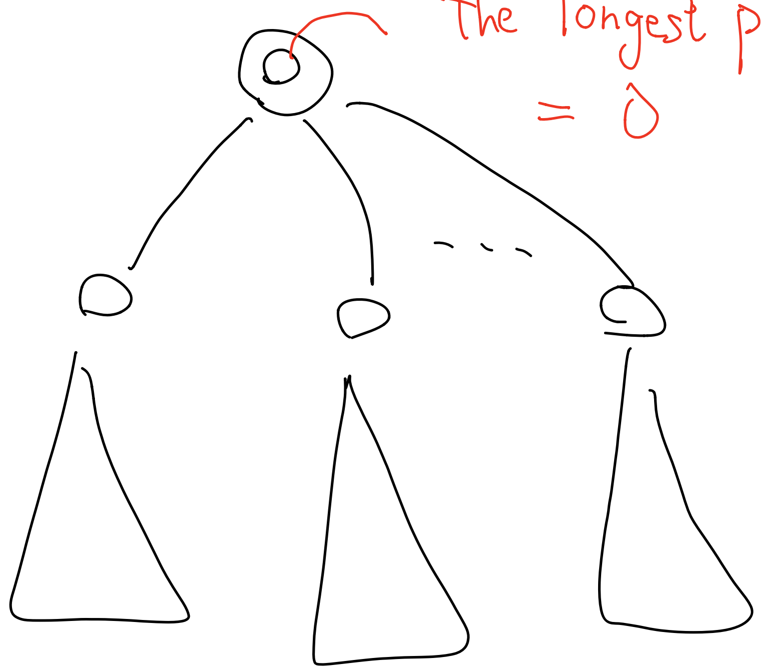
max(secondhighest[j], the longest path from j) + 1



② If highest[i]+1 < highest[j]:

the longest path from i =

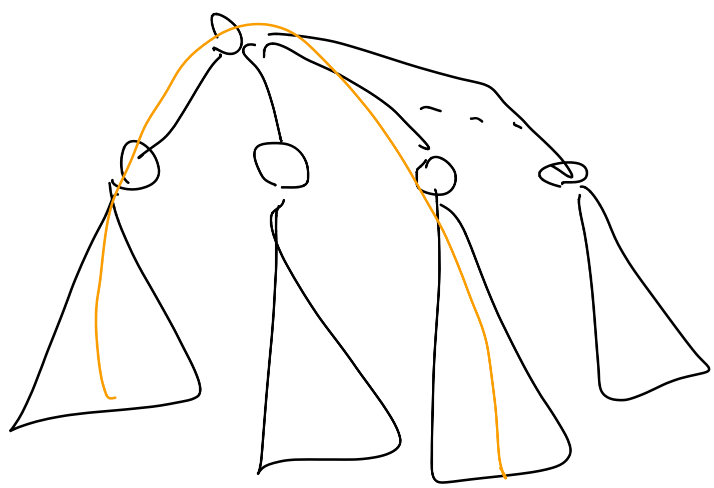
max (highest[j], the longest path from j) + 1



the longest path from ◯

= ∂

Assume: there is only one longest path



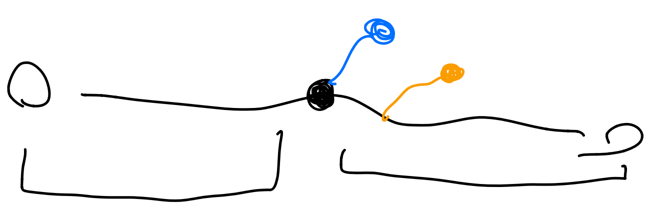
The path contains odd nodes

When we choose the middle point, the height of it is $\frac{1}{2}(\#\text{ of nodes} - 1) + 1$

If we choose ●, it will have

$\frac{1}{2}(\#\text{ nodes} - 1) + 1 + 1$

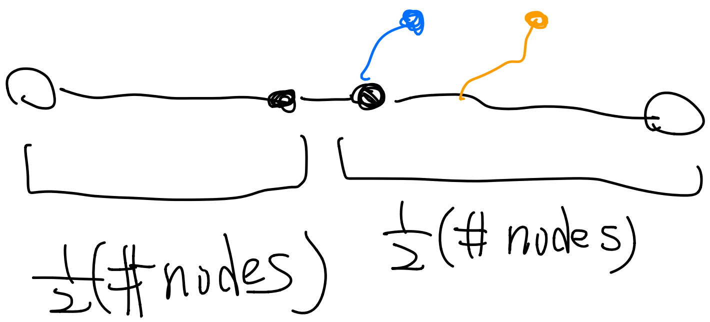
which is greater



$\frac{1}{2}(\#\text{ nodes} - 1)$ $\frac{1}{2}(\#\text{ nodes} - 1)$

Choose ● will be even longer.

When the path contain even nodes:



Choose :

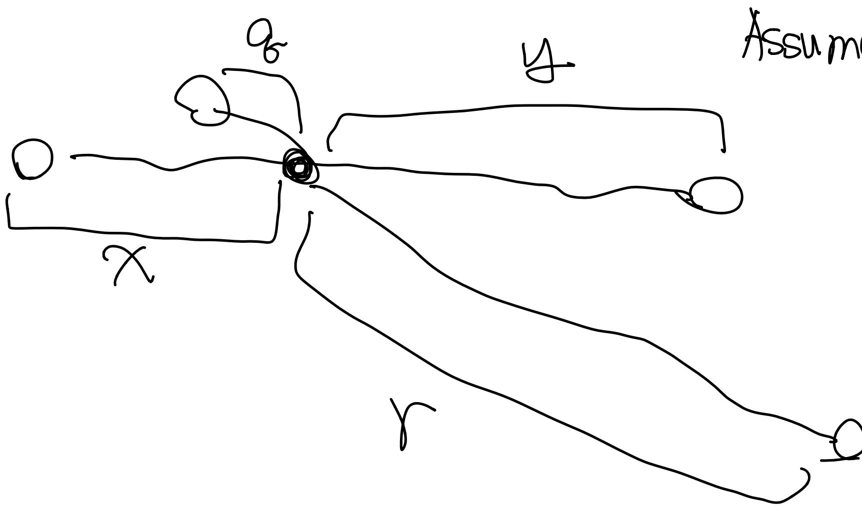
height = $\frac{1}{2}(\#\text{nodes}) + 1$

Choose :

height = $\frac{1}{2}(\#\text{nodes}) + 1 + 1$

Choose will be even larger.

When there are multiple longest path



Assume # nodes on the path $= p$

Assume $x + 1 + y = q + r + 1 = p$

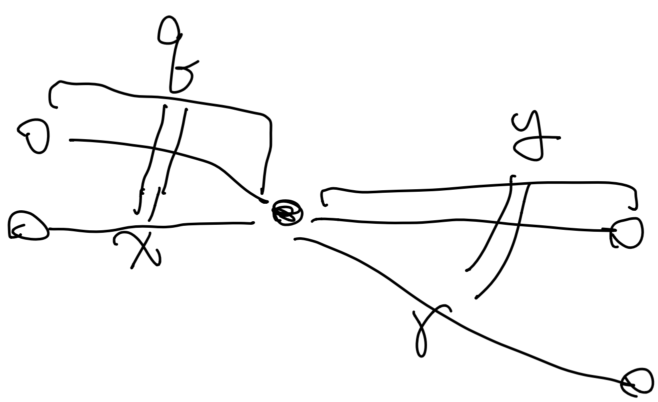
WLOG, we first assume $r > y$

If $r > y$, then $x + r + 1$ can form a even longer path, contradicts that $x + 1 + y$ or $q + r + 1$ are the longest path

So apparently, $r = y$,

if $r = y$, then $x = q$,

The tree should be like this

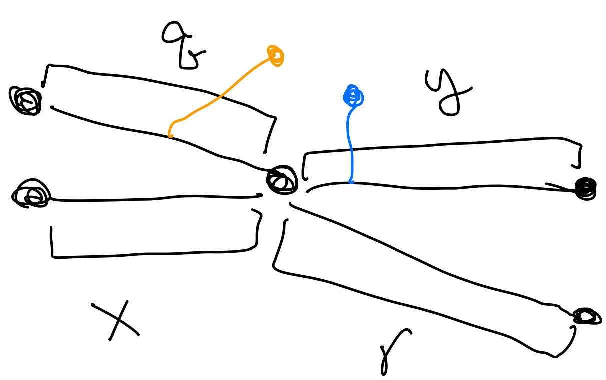


WLOG, $x \le y$ ($q \le r$)

However, if $x = q \le y = r$,

$y + r + 1$ can form a longer path!

It should look like this instead.



where $x = q = y = r$

$\Rightarrow$ We can just find the middle point.

Because choosing or will result a longer height