HOW BIG IS $d(n)$?

We need to find an upper bound on $d(n)$, the max degree of all root nodes.

> *Determine the minimum number of nodes possible in a tree with root of degree k.*

- Consider any node x of *degree* k with minimum possible nodes. How do we get there?

HOW BIG IS $d(n)$?

We need to find an upper bound on $d(n)$, the max degree of all root nodes.

> *Determine the minimum number of nodes possible in a tree with root of degree k.*

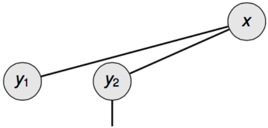
- Consider any node x of *degree* k with minimum possible nodes. How do we get there?
- Let $y_1, y_2, \ldots, y_k$ be the children in order that they are attached (during consolidate).

HOW BIG IS $d(n)$?

We need to find an upper bound on $d(n)$, the max degree of all root nodes.

> *Determine the minimum number of nodes possible in a tree with root of degree k.*

- Consider any node x of *degree* k with minimum possible nodes. How do we get there?
- Let $y_1, y_2, \ldots, y_k$ be the children in order that they are attached (during consolidate).

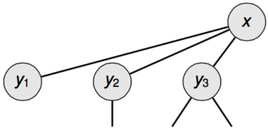


HOW BIG IS $d(n)$?

We need to find an upper bound on $d(n)$, the max degree of all root nodes.

> *Determine the minimum number of nodes possible in a tree with root of degree k.*

- Consider any node x of *degree* k with minimum possible nodes. How do we get there?
- Let $y_1, y_2, \ldots, y_k$ be the children in order that they are attached (during consolidate).

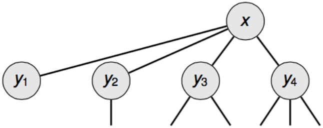


How big is $d(n)$?

We need to find an upper bound on $d(n)$, the max degree of all root nodes.

> *Determine the minimum number of nodes possible in a tree with root of degree k.*

- Consider any node x of *degree* k with minimum possible nodes. How do we get there?
- Let $y_1, y_2, \ldots, y_k$ be the children in order that they are attached (during consolidate).

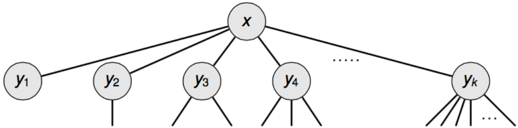


How big is $d(n)$?

We need to find an upper bound on $d(n)$, the max degree of all root nodes.

Determine the minimum number of nodes possible in a tree with root of degree k.

- Consider any node x of *degree* k with minimum possible nodes. How do we get there?
- Let $y_1, y_2, \ldots, y_k$ be the children in order that they are attached (during consolidate).

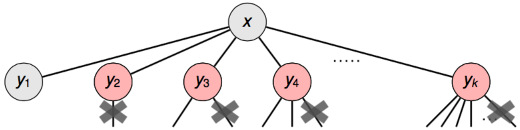


HOW BIG IS $d(n)$?

We need to find an upper bound on $d(n)$, the max degree of all root nodes.

> *Determine the minimum number of nodes possible in a tree with root of degree k.*

- Consider any node x of *degree* k with minimum possible nodes. How do we get there?
- Let $y_1, y_2, \ldots, y_k$ be the children in order that they are attached (during consolidate).
- And after we have *cut* as many nodes as possible?

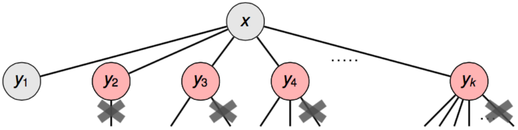


How big is $d(n)$?

We need to find an upper bound on $d(n)$, the max degree of all root nodes.

Determine the minimum number of nodes possible in a tree with root of degree k.

- Consider any node x of *degree* k with minimum possible nodes. How do we get there?
- Let $y_1, y_2, \ldots, y_k$ be the children in order that they are attached (during consolidate).
- And after we have *cut* as many nodes as possible?



Observation. $\forall i, 1 \leq i \leq k: \; d_i \geq i-2$

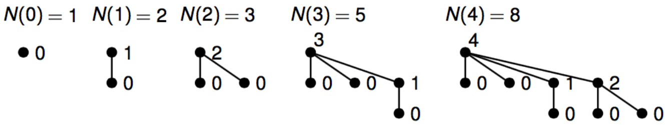
Bounding the number of nodes

Let's determine the *minimum* number of nodes $N(k)$ possible in a *tree* with root of degree k.

Observation.

Bounding the number of nodes

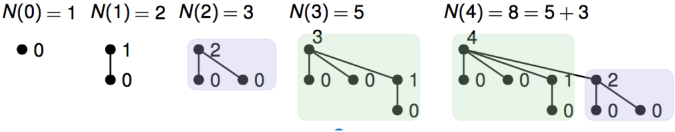
Let's determine the *minimum* number of nodes $N(k)$ possible in a *tree* with root of degree k.



Observation.

Bounding the number of nodes

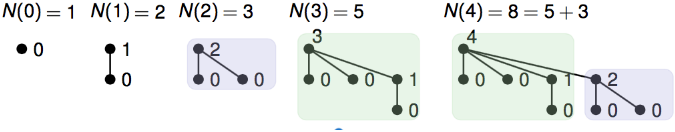
Let's determine the *minimum* number of nodes $N(k)$ possible in a *tree* with root of degree k.



Observation.

Bounding the number of nodes

Let's determine the *minimum* number of nodes $N(k)$ possible in a *tree* with root of degree k.



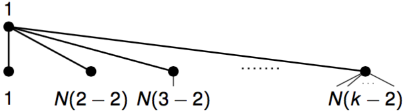
Observation. $N(k) = N(k-1) + N(k-2) = F(k+2)$?

where $F(k+2)$ is the $k+2^{nd}$ Fibonacci number.

N(k) = F(k+2)?

Recall. $\forall i, 1 \leq i \leq k: \;\; d_i \geq i - 2.$

$N(k) =$

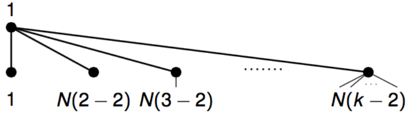


$$
\begin{aligned}
N(k) &= 1 + 1 + N(2-2) + N(3-2) + \cdots + N(k-2) \\
&= \\
&= \\
&= \\
&=
\end{aligned}
$$

N(K) = F(K+2)?

Recall. $\forall i, 1 \leq i \leq k: \;\; d_i \geq i-2.$

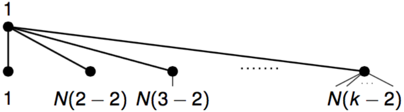
$N(k) =$



$$
\begin{aligned}
N(k) &= 1 + 1 + N(2-2) + N(3-2) + \cdots + N(k-2) \\
&= 1 + 1 + \sum_{j=0}^{k-2} N(j) \\
&= \\
&= \\
&=
\end{aligned}
$$

N(k) = F(k+2)?

Recall. $\forall i, 1 \le i \le k: \;\; d_i \ge i-2.$

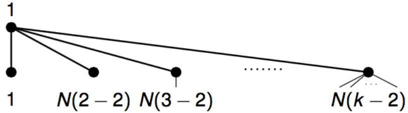
$N(k) =$ 

$$
\begin{aligned}
N(k) &= 1 + 1 + N(2-2) + N(3-2) + \cdots + N(k-2) \\
&= 1 + 1 + \sum_{j=0}^{k-2} N(j) \\
&= N(k-2) + 1 + 1 + \sum_{j=0}^{k-3} N(j) \\
&= \\
&=
\end{aligned}
$$

N(k) = F(k+2)?

Recall. $\forall i, 1 \le i \le k: \; d_i \ge i-2.$

$N(k) =$

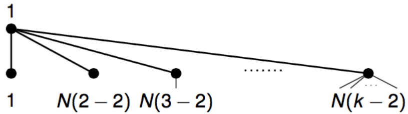


$$
\begin{aligned}
N(k) &= 1 + 1 + N(2-2) + N(3-2) + \cdots + N(k-2) \\
&= 1 + 1 + \sum_{j=0}^{k-2} N(j) \\
&= N(k-2) + 1 + 1 + \sum_{j=0}^{k-3} N(j) \\
&= N(k-2) + N(k-1) \\
&=
\end{aligned}
$$

N(K) = F(K+2)?

Recall. $\forall i, 1 \leq i \leq k: \;\; d_i \geq i-2.$

$N(k) =$



$$
\begin{aligned}
N(k) &= 1 + 1 + N(2-2) + N(3-2) + \cdots + N(k-2) \\
&= 1 + 1 + \sum_{j=0}^{k-2} N(j) \\
&= N(k-2) + 1 + 1 + \sum_{j=0}^{k-3} N(j) \\
&= N(k-2) + N(k-1) \\
&= F(k) + F(k+1) = F(k+2)
\end{aligned}
$$

$O(d(n)) \in O(\log n)$

Lemma. For all integers $k \geq 0$, $F(k+2) \geq \varphi^k$ where $\varphi = \frac{(1+\sqrt{5})}{2} = 1.61803\ldots$.

Q. What is φ?

A.

$O(d(n)) \in O(log n)$

Lemma. For all integers $k \geq 0$, $F(k+2) \geq \varphi^k$ where $\varphi = \frac{(1+\sqrt{5})}{2} = 1.61803\ldots$.

Q. What is φ?

A. Solution to $\varphi^2 = \varphi + 1$.

We can prove this by *induction* on k.

Q. Why is this useful?

A.

$O(d(n)) \in O(logn)$

Lemma. For all integers $k \geq 0$, $F(k+2) \geq \varphi^k$ where $\varphi = \frac{(1+\sqrt{5})}{2} = 1.61803\ldots$.

Q. What is φ?

A. Solution to $\varphi^2 = \varphi + 1$.

We can prove this by *induction* on k.

Q. Why is this useful?

A. Shows that *Fibonacci* numbers grow at least *exponentially* fast in k.

Which means...

$O(d(n)) \in O(logn)$

Lemma. For all integers $k \geq 0$, $F(k+2) \geq \varphi^k$ where $\varphi = \frac{(1+\sqrt{5})}{2} = 1.61803\ldots$.

Q. What is φ?

A. Solution to $\varphi^2 = \varphi + 1$.

We can prove this by *induction* on k.

Q. Why is this useful?

A. Shows that *Fibonacci* numbers grow at least *exponentially* fast in k.

Which means...

$$N(k) = F(k+2) \geq \varphi^k$$

$\Rightarrow$

$O(d(n)) \in O(logn)$

Lemma. For all integers $k \geq 0$, $F(k+2) \geq \varphi^k$ where $\varphi = \frac{(1+\sqrt{5})}{2} = 1.61803\ldots$.

Q. What is φ?

A. Solution to $\varphi^2 = \varphi + 1$.

We can prove this by *induction* on k.

Q. Why is this useful?

A. Shows that *Fibonacci* numbers grow at least *exponentially* fast in k.

Which means...

$$N(k) = F(k+2) \geq \varphi^k$$

$\Rightarrow$ number of nodes $n \geq N(k) \geq \varphi^k$.

$\Rightarrow$

$O(d(n)) \in O(logn)$

Lemma. For all integers $k \geq 0$, $F(k+2) \geq \varphi^k$ where $\varphi = \frac{(1+\sqrt{5})}{2} = 1.61803\ldots$.

Q. What is φ?

A. Solution to $\varphi^2 = \varphi + 1$.

We can prove this by *induction* on k.

Q. Why is this useful?

A. Shows that *Fibonacci* numbers grow at least *exponentially* fast in k.

Which means...

$$N(k) = F(k+2) \geq \varphi^k$$

$\Rightarrow$ number of nodes $n \geq N(k) \geq \varphi^k$.

$\Rightarrow$ $\log_\varphi n \geq k$ where k is...$d(n)$.

Therefore,

`Extract_Min` amortized cost of $O(d(n))$ is really $O(\log n)$.