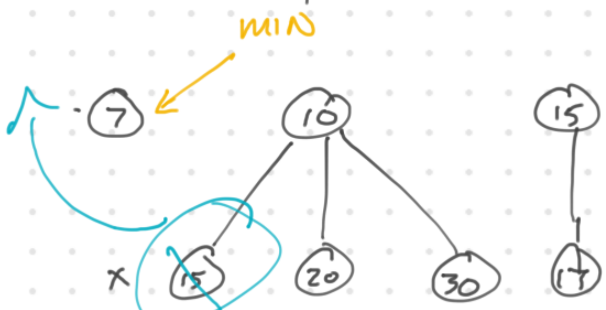
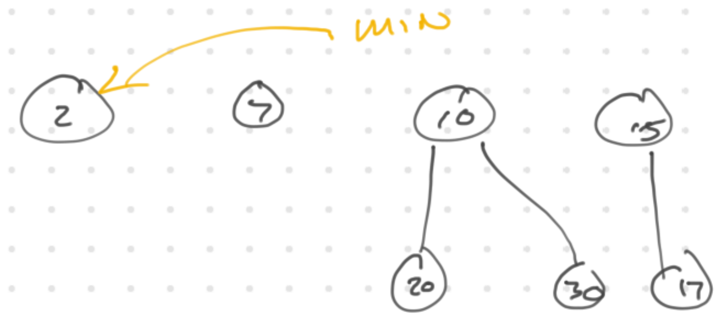
FH

Decreasing a key in FH.

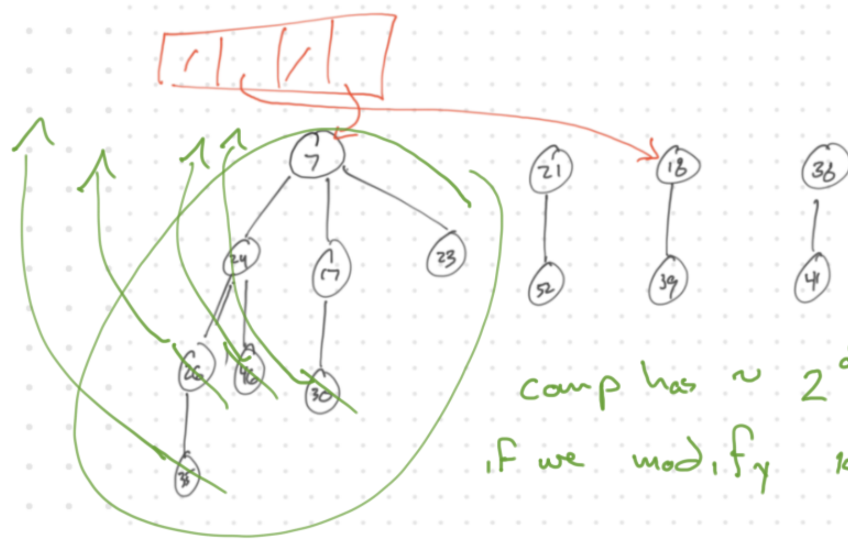


~~Re~~ change key to 2

we can delete x from list of children of x.parent + insert x into the list of roots w/ the new key value



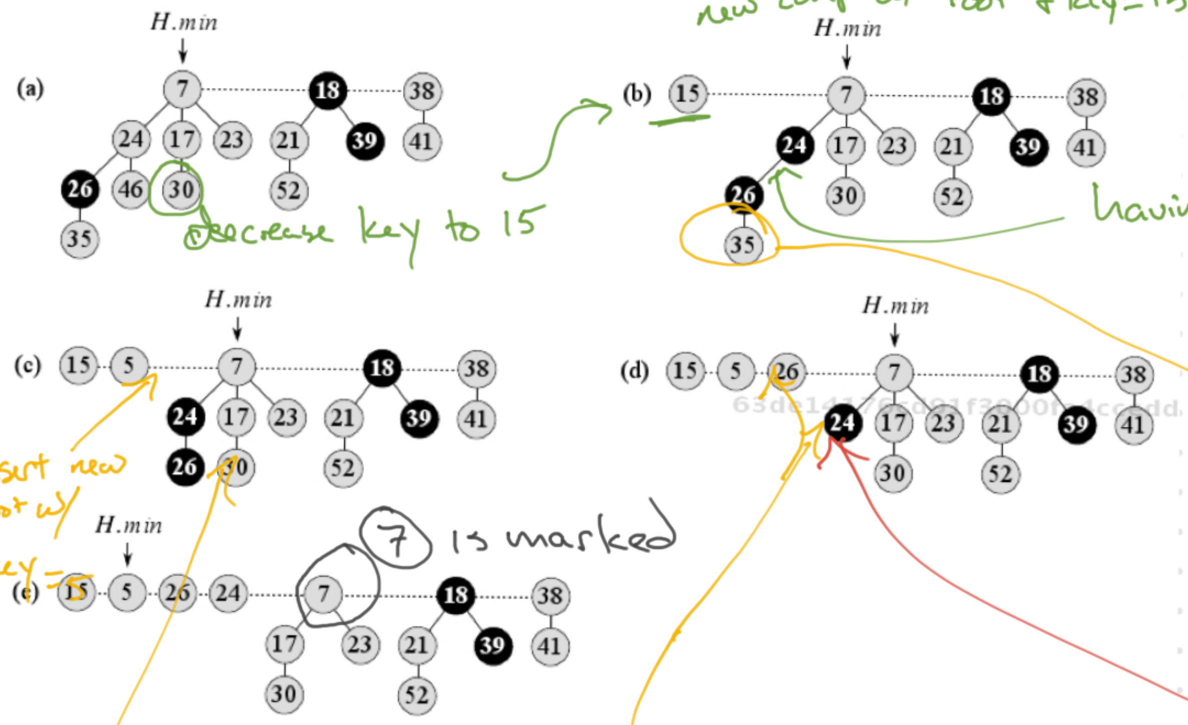
Here is where the marks come into play



comp has ~ 2^{deg} # of vert if we modify key what's going on with the marked vertices

- at some point, x was inserted (as a root) marked as ~~t~~ FALSE
- later on, x was made the child of another node (still marked false)
- still later, children of x are cut away - we're ok if it happens once, but at that point, we mark the vertex to TRUE, + we don't cut away a neighbor in the future, - Instead, if we return to the vertex, we move it to roots + remark it FALSE

new comp w/ root + key=15



decrease key to 15

having deleted this node's child, it gets marked

insert new root w/ key=5

7 is marked

Subsequently, we change this node's key to 5

This node has had a child deleted + is marked

26 was marked, so it becomes a root of a new comp + is unmarked

24 is still marked + lost a child, so we move it to be a root by itself

```
FIB-HEAP-DECREASE-KEY(H, x, k)
  if k > x.key
      error "new key is greater than current key"
  x.key = k
  y = x.p
  if y ≠ NIL and x.key < y.key
      CUT(H, x, y)
      CASCADING-CUT(H, y)
  if x.key < H.min.key
      H.min = x
```

check we are really reducing key

update + take the parent y

at very end, need to check if the min has moved + update accordingly.

```
CUT(H, x, y)
  remove x from the child list of y, decrementing y.degree
  add x to the root list of H
  x.p = NIL
  x.mark = FALSE
```

cut node x out of the tree

+ make a new component of H

w/ root x (+ set mark(x) = FALSE)

```
CASCADING-CUT(H, y)
  z = y.p
  if z ≠ NIL
      if y.mark == FALSE
          y.mark = TRUE
      else CUT(H, y, z)
          CASCADING-CUT(H, z)
```

take next parent up

if for the first time, we find a node marked FALSE, we change to TRUE + terminate

+ ow, we cut the vertex + proceed to it's parent.

amortized complexity

```
FIB-HEAP-DECREASE-KEY(H, x, k)
if k > x.key
    error "new key is greater than current key"
x.key = k
y = x.p
if y ≠ NIL and x.key < y.key
    CUT(H, x, y)
    CASCADING-CUT(H, y)
if x.key < H.min.key
    H.min = x
```

O(1)

O(1)

O(1)

O(1)

```
CUT(H, x, y)
remove x from the child list of y, decrementing y.degree
add x to the root list of H
x.p = NIL
x.mark = FALSE
```

O(1)

```
CASCADING-CUT(H, y)
z = y.p
if z ≠ NIL
    if y.mark == FALSE
        y.mark = TRUE
    else CUT(H, y, z)
        CASCADING-CUT(H, z)
```


Conclusion

overall complexity is $O(c)$ where c is # of recursive calls to Cas_Cut

we just need to check how much since at each pass, we do $O(1)$ work, not counting recursive calls

if we do c recursive calls, we do $O(c)$ work here amortized complexity:

$$= O(c) + T(H') + 2m(H') - T(H) - 2m(H)$$

(where H' is new ~~tree~~ FH after updating H)

$$= T(H) + c \qquad\qquad m(H) - c + 2$$

because we do one cut in main code + for all but one recursive call ~~to~~ in cas_cuts requires another call to cut

in reality, in order to make this work out, we want $\Phi(H) = \alpha(T(H) + 2m(H))$

$$\text{amortized complexity} = O(c) + \cancel{T(H)} + \cancel{c} - \cancel{2}(\cancel{m(H)} - c + 2) - \cancel{T(H)} - \cancel{2m(H)}$$

$$= O(c) - (\alpha - 1)c + 4 = \boxed{O(1)}$$

constant depending on implementation of rewriting pointers

FH_delete(& H, x)

FH_DECREASE_KEY $(H, x, -\infty)$ $O(1)$

FH_EXTRACT_MIN (H) $O(D(n))$

$D(n)$ is max deg of a node.

$\Rightarrow$ amortized complexity is $O(D(n)) = O(\log n)$

Proof $D(n) = O(\log n)$

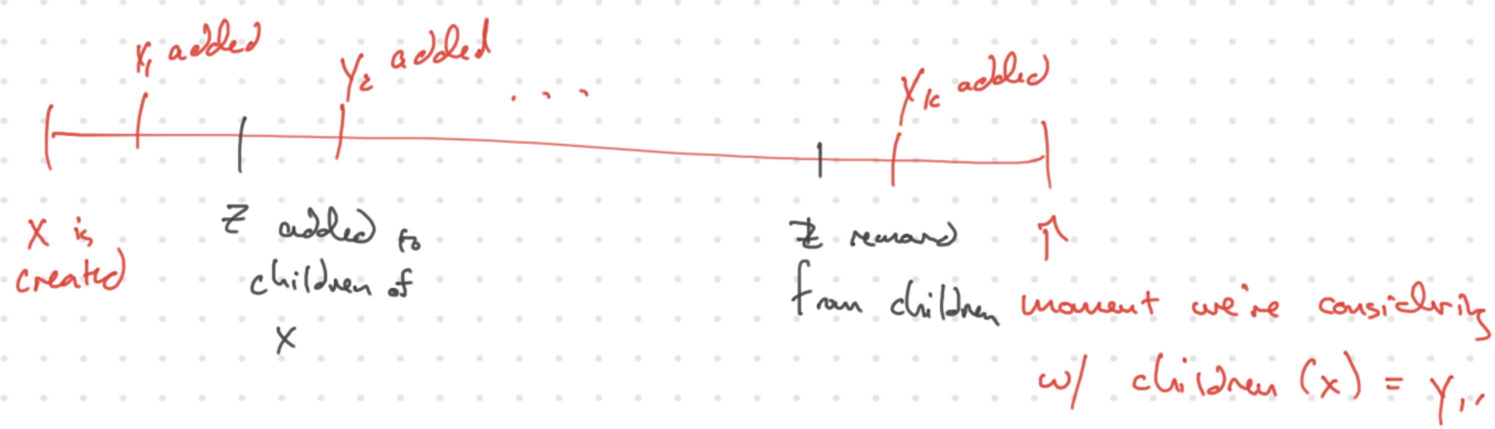
Lem Let x be a node in FH H w/ $x.deg = k$. Let $y_1, y_2, \ldots, y_k$ be ~~neighbor~~ children of x & assume This is the order in which They were linked to x. Then $y_1.deg \geq 0$ & $y_i.deg \geq i-2$ $\forall i$

Pf $y_1.deg \geq 0$ ✓

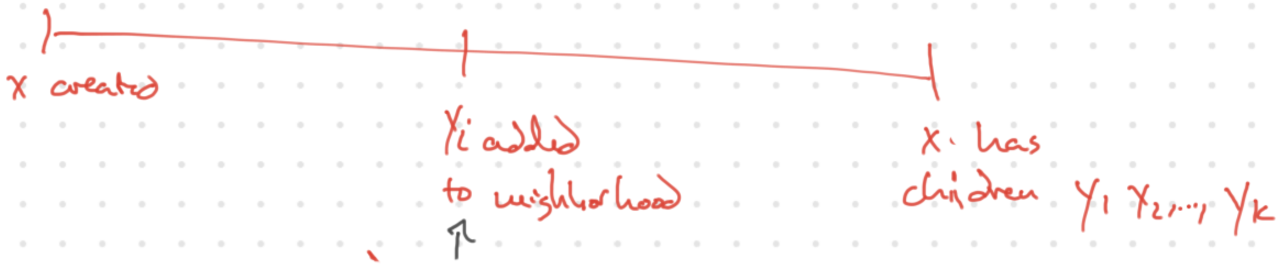
for $i \geq 2$. y_i was attached to x during consolidation & at that moment, ~~deg~~ $x.deg \geq i-1$ (∵ there were the vertices $y_1, y_2, \dots, y_{i-1}$)

$\Rightarrow$ at that moment, $y_i.deg = x.deg \geq i-1$

Q could we say $x.deg = i-1$ at that point?



We can't say $x.deg = i-1$, just $x.deg \geq i-1$ because there could be children (distinct from $y_1, y_2, \dots, y_k$ (like z)) which are later deleted



$$\underbrace{y_i.\deg \geq x.\deg \geq i-1}$$

what happens here to $y_i.\deg$ – y_i could lose a child in this time frame, but if it did, it would have been marked & it happened exactly once.

$\Rightarrow$ at the end, $y_i.\deg \geq (i-1) - 1 = i-2$

as claimed. ∎

1 1 2 3 5 8

F_k := Fibonacci #'s F_0 F_1 F_2 F_3 F_4 F_5

$F_0 = 1$

$F_1 = 1$

$F_k = F_{k-1} + F_{k-2}$

Lem $F_{k+2} = 1 + \sum_{i=0}^{\cancel{k-1}\ k} F_i \quad \forall k \geq 1$

pf induction on k

$k=1 \to F_3 = 1 + \sum_{i=0}^{\cancel{0}\ 1} F_i = 1 + \cancel{1}\ 2$ ✓ $\quad (F_3 = 3)$

$k=2 \to F_4 = 1 + \sum_{i=0}^{\cancel{1}\ 2} F_i = 1 + \cancel{2}$ ✓

$F_4 = \cancel{3}\ 5$

$1+1+1+2 = 5$

assume holds for k correct

$F_{k+1} = 1 + \sum_{i=0}^{k-1} F_i$

$F_{k+2} = F_{k+1} + F_k$

by induction

$= 1 + \sum_{i=0}^{k-1} F_i$

$= 1 + \sum_{i=0}^{k} F_i$ ✓

Lem $\forall\, k \geq 0$, $F_{k+1} \geq \phi^k$ where

$$\phi = \frac{1+\sqrt{5}}{2}$$

pf induction k

$k=0$

$$F_1 \geq \phi^0$$
$$F_1 = 1$$

$k=1$

$$F_2 \geq \phi^1 = 1.6\ldots$$
$$F_2 = 2 \quad ✓$$

in general

$$F_{k+2} = F_{k+1} + F_k$$
$$\geq \varphi^k + \varphi^{k-1}$$
$$= \varphi^{k-1}\underbrace{(\varphi + 1)}$$

cl $\phi + 1 = \phi^2$

$$\phi + 1 = \frac{\cancel{2}\,3 + \sqrt{5}}{2}$$

$$\phi^2 = \left(\frac{1+\sqrt{5}}{2}\right)\left(\frac{1+\sqrt{5}}{2}\right) = \frac{1}{4}\cdot\left(1 + 2\sqrt{5} + 5\right) = \frac{1}{4}\left(6 + 2\sqrt{5}\right)$$

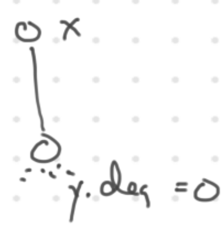
$$= \varphi^{k-1}\left(\varphi^2\right) = \varphi^{k+1} \quad ✓$$

Lem let x be any node of ~~FH~~ H & let $k = x.\deg$. Then size of downtree of x (denote it $\text{size}(x)$) $\geq F_{k+2} \geq \varphi^k$

pf let s_k be the min size of a node x w/ $x.\deg = k$

$s_0 = 1$

$s_1 = 2$

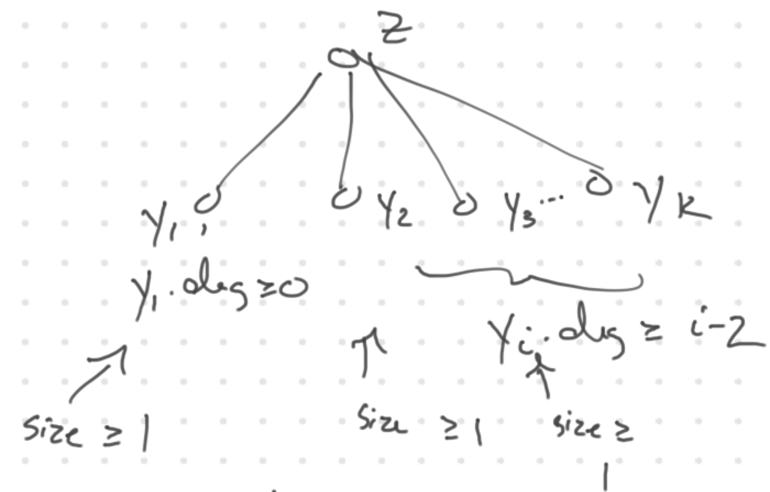


remember s is a lower bound & will always be ≥ 2

$s_1 \geq 2$.

consider some node z with $\text{size}(z) \geq s_k$ where $k = z.\deg$

let $y_1, \ldots, y_k$ be children of z



$$\text{size}(z) \geq 2 + \sum_{i=2}^{k} s_{y_i.\deg}$$

$$\underbrace{\geq 2 + \sum_{i=2}^{k} s_{i-2}}_{\text{holds for all}}$$

by lower bound we proved on ~~deg~~ y_i. deg

we want to prove $s_k \geq F_{k+1}$. $(\forall k)$

holds for $k = 0, 1$

$k \geq 2$, by induction on k, consider

$$s_k \geq 2 + \sum_{i=2}^{k} s_{i-2}$$

$$\geq 2 + \sum_{i=0}^{k-2} s_i \geq \cancel{\Sigma}\ 2 + \sum_{i=0}^{k-1} F_i = 1 + F_{k+1}$$

(by lemma) ✓

Conclusion is $S_k \geq \cancel{\phi}\, F_{k+1} \geq \phi^k$ ∎

ie it's exponential in the deg

+ Therefore $D(n) \leq \lceil \log_\phi n \rceil$

$\Rightarrow D(n) = O(\log n)$