

★ Fibonacci Heap

→ extract max, delete takes a lot of time

getmax : $O(1)$

meld : $O(1)$

insert : $O(1)$

★ All are amortized times

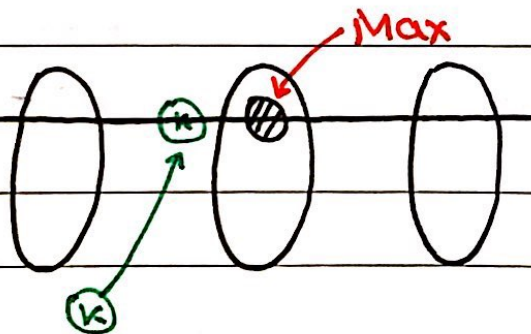
extract max : $O(\log n)$

increase key : $O(1)$

Delete key : $O(\log n)$

Build heap

★ The pointer (head) points to the max rootlist stored in doubly circular linked list



∴ getmax : $O(1)$

Insert : $O(1)$ (compare k with max & adjust pointers)

* $\phi(H) = t(H)$: no. of trees in the heap

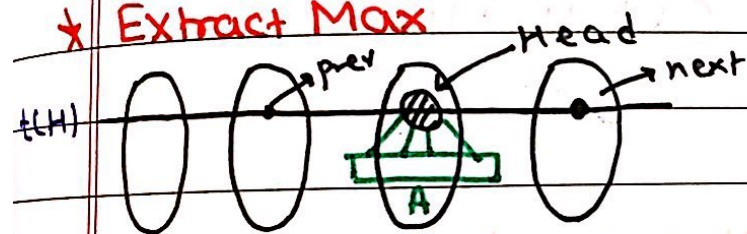
$\phi_0(0) = 0$ (# empty heap)

Insert: $\Delta\phi = t(H_{\text{new}}) - t(H_{\text{old}}) = O(1)$

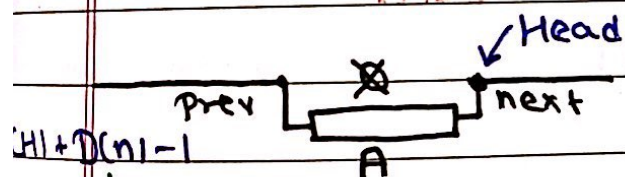
\therefore Insert: $O(1)$

Getmax: $O(1)$

* Extract Max



* Remove the head & maintain the children of head in a circular doubly linked list



* Then merge all siblings with root list in $O(1)$

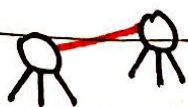
max Degree of a tree in a heap

* Consolidate Fibonacci Heap

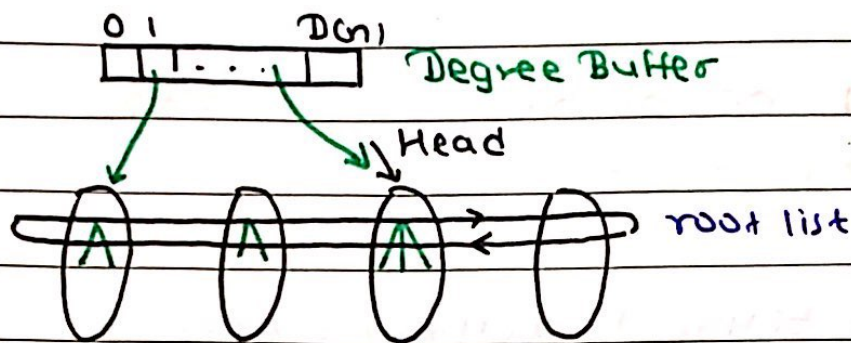
To have almost $D(n)+1$ trees in the heap

PHP $0, 1, 2, \dots, D(n)$: Degrees

→ IF 2 trees have the same degrees



check root values of the two trees



Each ele. of buffer points to almost 1 tree

If clash in buffer merge the two trees

So Consolidate takes $O(t(H) + D(n))$

\therefore Extract max:

$$O(t(H) + D(n)) + cD(n) + 1 - t(H) \\ = O(t(H)) - t(H) + O(D(n))$$

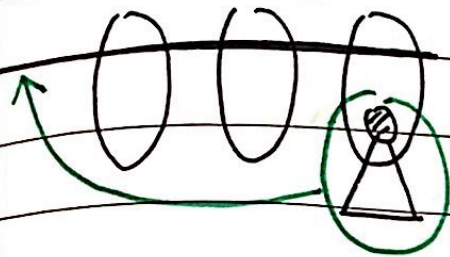
$$\text{Let } O(t(H)) \leq c t(H)$$

$$\text{Let } \phi(H) = \alpha t(H) \text{ where } (\alpha > c)$$

\therefore Extract max:

$$O(t(H) + D(n)) - \alpha t(H) + O(D(n)) \cdot \alpha \\ = c t(H) - \alpha t(H) + O(D(n)) \\ = \underline{\underline{O(D(n))}} \rightarrow \text{extract max}$$

* Increase Key:

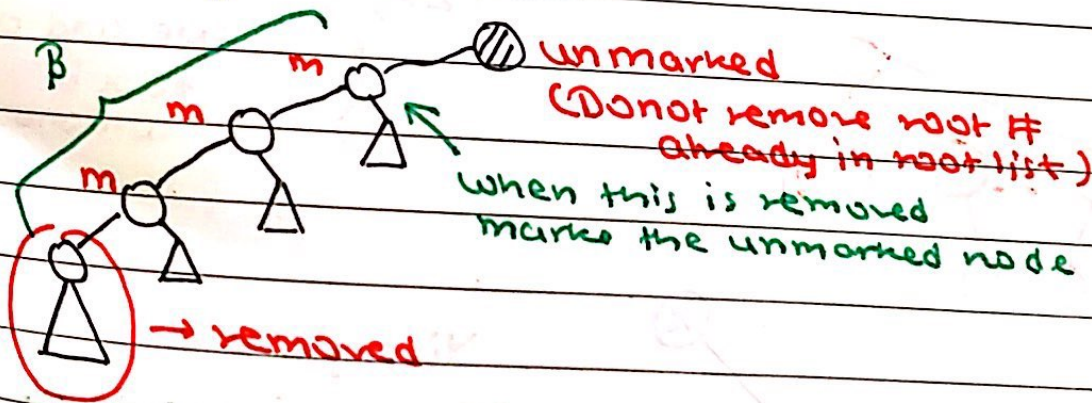


* Donot use check parent & swap # worst case $O(h) = O(n)$

add  to the root list

* If a child is lost mark the parent (by m)
If any parent loses ≥ 2 children move the parent to the root list.


→ Cascading cuts:



* Now consider $\phi(H) = \alpha(t(H) + 2m(H))$
 $m(H) \geq 0 \therefore \phi(H_0) = 0$
 why no 1? Let as ex to reader.
 ↓
 no. of marked nodes

* Increase key:

$$O(\beta) + \alpha(t(H) + \beta + 2 \times (m(H) - (\beta - 1) + 1)) - \alpha(t(H) + 2m(H))$$

node incr not marked 

$$= O(\beta) + 4\alpha - \alpha\beta = c'\beta - \alpha\beta + 4\alpha$$

Set $\alpha = \max(c', c) \therefore$ Insert: $O(1)$

* Merge:

Join 2 Linked list & update max in $O(1)$

→ Consolidate done in Extract max to save the cost of merge.

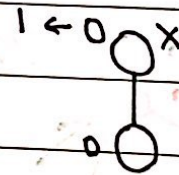
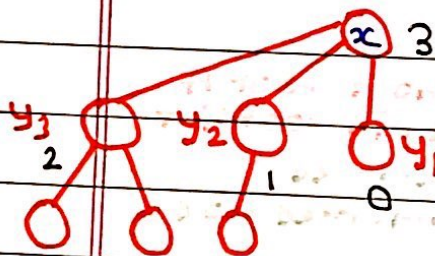
* Delete:

Increase the key to ∞

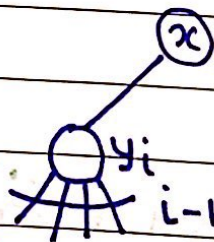
& then extract max

$O(D(n))$

* If $x.\text{degree} = k$ then $\text{size}(x) = S_k$



We add child to x in consolidate.



$y_i.\text{deg} \geq i-2$

(# One child could be lost # marked)

$$\therefore \text{size}(x) \geq 2 + \sum_{i=2}^d S_{y_i.\text{degree}}$$

\downarrow
 y_1 & root

$\therefore S_k$ is monotonically \uparrow w.r.t k

$$\therefore S_{y_i.\text{deg}} \geq S_{i-2}$$

$$\therefore \text{size}(C_k) \geq 2 + \sum_{i=2}^d S_{i-2} \quad (\# d \text{ is } \deg(C_k))$$

Claim: $S_k \geq F_{k+2}$

Base $S_0 = 1 \geq F_2 = 1$

$$S_1 = 2 \geq F_3 = 2$$

TPT $S_k \geq F_{k+2}$

$$S_k \geq 2 + \sum_{i=2}^k S_{i-2}$$

$$\Rightarrow S_k \geq 2 + \sum_{i=2}^k F_i$$

$$\geq 1 + \sum_{i=0}^k F_i = F_{k+2} \quad (\# \text{Property of Fib})$$

$$\therefore n \geq \text{size}(C_k) = S_k \geq F_{k+2} \geq \phi^{k+2}$$

ϕ : golden ratio

$$\therefore k = O(\log n)$$