* priority queue : max-heap — insert ~~O(log n)~~ O(1)

remove ~~Min~~ Max O(log n)

hard if every operation O(1)

possible if "amortized" O(1)

均攤

{ cheap insertion usually ⟺ insertion to "small tree" usually

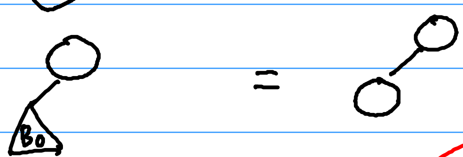
{ expensive insertion sometimes ⟺ insertion to "big tree" sometimes

森林

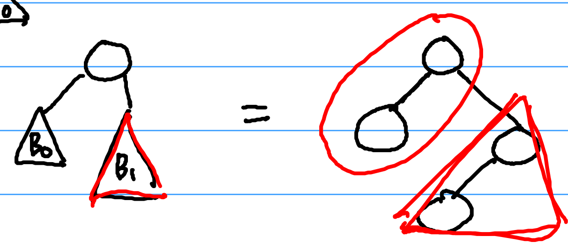
forest

* binomial tree

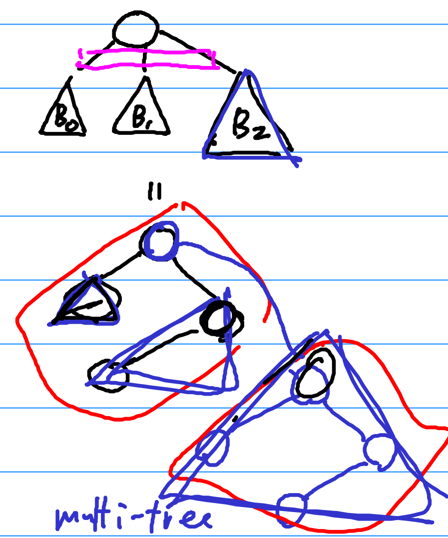
① B_0 = 

② B_1 = 

$\simeq 2 B_0$

④ B_2 = 

$= 2 B_1$

⑧ B_3 = 

$= 2 B_2$

B_k : 2^k nodes

\+ max-tree (multi-tree)

* binomial forest + max-trees

{ at most one B_k (max-tree) per each k }

⇒ can represent any n

n = 10 : 8 (B_3) + 2 (B_1)

n = 17 : 16 (B_4) + 1 (B_0)

getMax O(log n)

removeMax

* binomial heap {}

insert (5) into {} ⇒ {B_0} {(5)}

insert {(7)} into {(5)} ⇒ {B_1} merge (B_0, B_0) {(7)–(5)}

merge

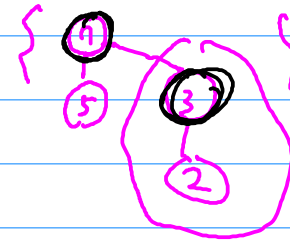
small bin. forest

B_0

insert (3) into {(7)–(5)} ⇒ {B_0, B_1} {(3), (7)–(5)}

insert (2) into {(3), (7)–(5)} ⇒ {B_2}

merge ×2



⋮

B_0

n :

n	insert	B_0	
$\frac{n}{2}$	merge	B_0, B_0	$O(1)$
$\frac{n}{4}$	merge	B_1, B_1	$O(1)$
$\frac{n}{8}$	merge	B_2, B_2	$O(1)$

{B_0, B_1, B_2, B_1, B_0}

$O(n)$

$O(n)$

$O(n)$

⇓

amortized $O(1)$

$n = 15 = (1111)_2 \equiv \{B_0, B_1, B_2, B_3\}$

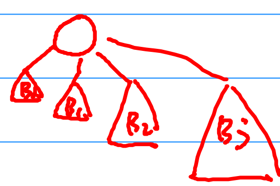
$n = 14 = (1110)_2 \equiv \{\ \ B_1, B_2, B_3\}$

$n = 13$

$n = 12$

⋮

$n = 8$



$\lfloor \log_2 n \rfloor + 1 = O(\log n)$

$(n - 2^k) + (2^k - 1)$

$B_3\ B_2\ B_1\ B_0$

$14 = (1\,1\,1\,0)_2 + (1\,1\,0\,1)_2$

$(\ \ \ 1\ 1)_2$