2. Mergeable heaps

2018年9月18日 21:20

	Binary heap	Binomial heap
$insert$	$\Theta \log n$	$O \log n$
max/min	$\Theta 1$	$O \log n$
$extract\text{-}max/min$	$\Theta \log n$	$O \log n$
$union$	$O(n_1 + n_2)$	$O(\log(n_1 + n_2))$

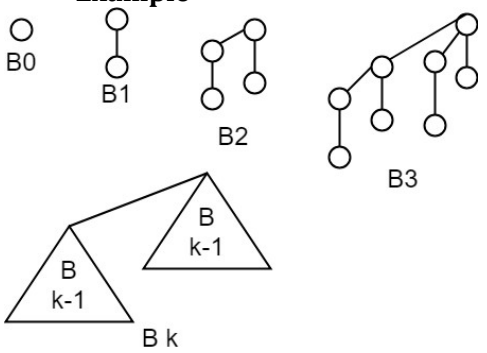
Binomial Heap

Binomial tree

Base case: B_0 one node

Induction process: B_n two copies of B_{n-1} are add an edge connect two left-top-most node.

Example



Properties

For a binomial tree B_k.

#nodes = 2^k, height = k, #children of the root = k.

the root has subtrees $B_0, B_1, \dots, B_{k-1}$

For level k of a n-height binomial tree, the #nodes on the level is $\binom{n}{k}$

Binomial Forest

A collection of binomial trees of distinct sizes.

Because every decimal number has a unique binary representation, for a binomial forest of n nodes:

$n = \sum_0^{k-1} b_i 2^i$, $b_i \in \{0,1\}$. $b_i = 1 \Rightarrow$ create a copy of B_i tree

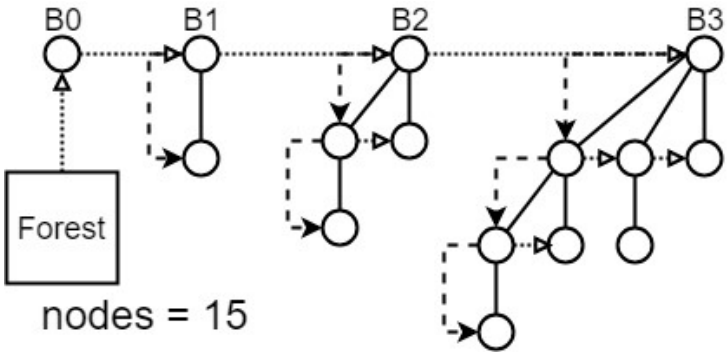
$k =$ #trees in a binomial forest of size n

Since the maximum capacity for a k-1 digits binary number is $2^k - 1 \geq n$, $\arg\min k \leq ceil(\lg(n+1))$

Min-Binomial Heap

A binomial forest where the keys are stored at the nodes such that min-heap property satisfied (i.e. key of node ≤ key of its children)

Implementation



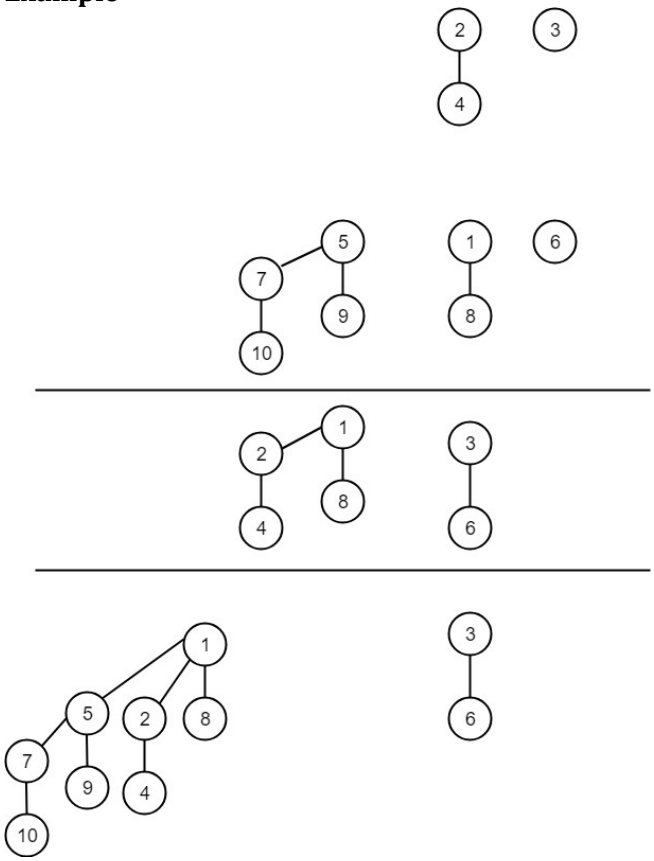
Each node contains

- Key
- Degree (the node is the root of a B_(degree) tree)
- *parent
- *left child
- *right sibling

The trees are linked by each root as a node in a linked list from small tree to large

min()
Go through all the roots of the trees, take $O(k) \Rightarrow O(\log n)$

union$(\boldsymbol{H_1}: \boldsymbol{MBH}, \boldsymbol{H_2}: \boldsymbol{MBH})$
Represents each forest as a binary representation $b_k b_{k-1} \dots b_0$, then do binary sum on it.
Merge tree with the same size by comparing the root, the smaller one becomes the parent of the larger one
Example



The runtime is just the comparison in each merging process, which is at most $\max(k_1, k_2) \Rightarrow O(\log(n_1 + n_2))$

insert$(\boldsymbol{e}: \boldsymbol{element\ \bar{w}\ key})$

$union\big(H, make\_MBH(e)\big)$

extract_min( )
Take out the min element from its tree, makes the remaining of the tree into a forest, then union with the rest trees in the original forest as a forest.