CMSC351-2023-10-06

Friday, October 6, 2023 8:26 AM

HEAPS AND HEAPSORT

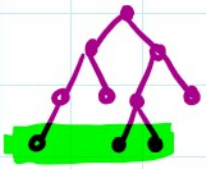
(1) Complete Binary Trees

(A) defn: A complete binary tree is a BT in which every level is full except the lowest level doesn't have to be, but in the lowest level all the nodes are to the left.

ex

COMPLETE!

ex

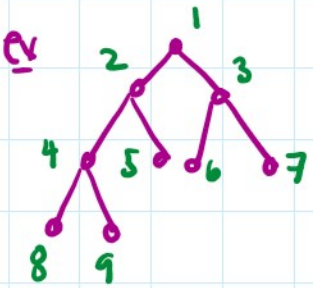


NOT COMPLETE!

(B) Index notes

- Firstly we 1-index the nodes, top to bottom, left to right:

ex



note: Some resources 0-index!

- If a node has index i, its children will have indices:
  Left: $2i$
  Right: $2i+1$
- If a node has index i, its parent will have index $\lfloor i/2 \rfloor$
- if a CBT has n nodes, then the node with largest index that has children has index $\lfloor n/2 \rfloor$
- The nodes w/ children have indices $1, 2, \ldots, \lfloor n/2 \rfloor$

(C) Level notes

- The leftmost node on level k has index 2^k.

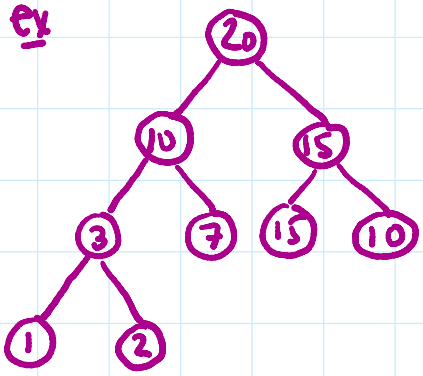
(c) Level notes

- The leftmost node on level k has index 2^k.
- A node w/ index i is on level $\lfloor \lg i \rfloor$.
- If we have n nodes then max level is level $\lfloor \lg n \rfloor$.
- Given a node w/ index i, the number of levels btw that node's level and the max level (inclusive) is $\lfloor \lg n \rfloor - \lfloor \lg i \rfloor + 1$

② Max Heap:

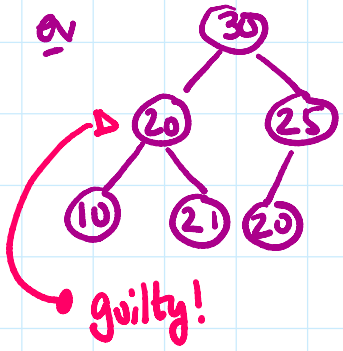
defn: a max heap is a CBT in which each node's key is $\geq$ the keys in its children.

ex



MAX HEAP!

ex



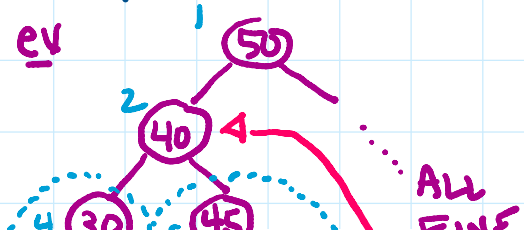
NOT MAX HEAP!

③ Converting a CBT to a Max Heap

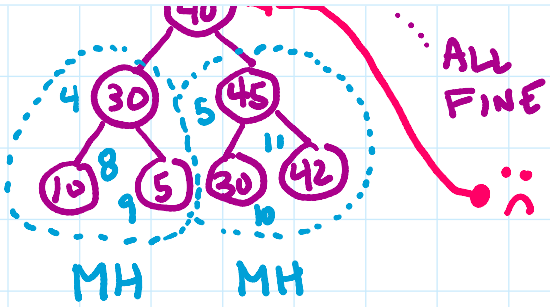
(A) Given a CBT, how can we algorithmically convert it to a MH?
We'll have two fctns/algorithms.

(B) maxheapify (aka swapkeydown)
sps we have a node s.t. both child subtrees are max heaps but perhaps the node's subtree isn't:

ex



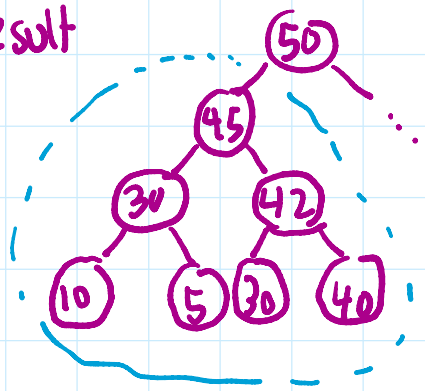
maxheapify checks if that key is less than either of the child keys. If so, it swaps that key w/ the



If SD, it swaps that key w/ the largest ⊛ child key and recursively calls itself on that child
⊛ if child keys =, pick either.

↳ ex we'd swap index 2 w/ index 5 (keys 40↔45)
and then calls itself on index 5
which then swaps index 5 w/ index 11 (keys 40↔42)

Result



Now: entire subtree rooted at the original node same is a MH.

Time Complexity?

Best case: $\Theta(1)$ if child keys not larger.
Worst case: must swap to the bottom of the heap.
each swap is $\Theta(1)$.
at most $O(\lg n)$ swaps blc number of levels is $O(\lg n)$

(c) Convert to max heap: go to the last node w/ children, which is node $\lfloor n/2 \rfloor$.
We call max heapify on nodes:
$\lfloor n/2 \rfloor, \ldots, 3, 2, 1$
We go up blc that way when we call it on a node, we know its child subtrees are MHs!
Time: BC: $\lfloor n/2 \rfloor$ obs. each $\Theta(1)$ so $\Theta(n)$

Now, we know its child subtrees are MMS!

Time: BC: $\lfloor n/2 \rfloor$ ops. each $\Theta(1)$ so $\Theta(n)$

WC: $\lfloor n/2 \rfloor$ ops. each $O(\lg n)$ so $O(n \lg n)$