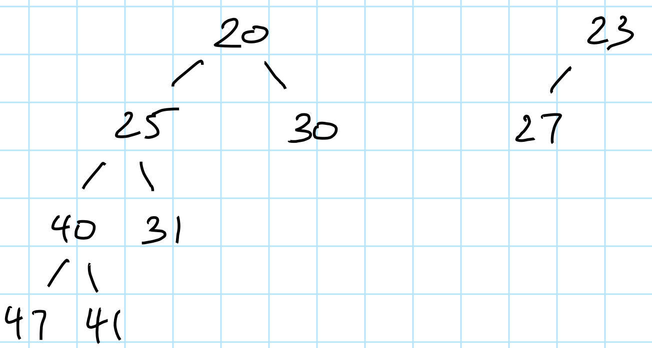
Lecture 13 - Oct 02

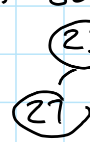
- Assignment 4 due Oct 14, but you can do Problems 1, 2 now
- Onboarding Exam <u>for CS 340</u> must be done by Oct. 8.
  (It might <u>not</u> be enough if you have done an Onboarding Exam for some other course)
- For Onboarding exam, test proctortrack with same environment as you would use in the actual exam, and have your Photo ID at hand.
- Onboarding exam may take 8 hrs to be evaluated.
  If you do not pass, you have to retake it.

<u>Example 28</u> (Skew heap merge)



→ smaller of the two roots is 20

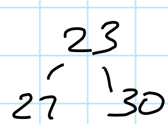
→ use 20 as root and recursively merge its right subtree 30 with



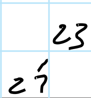
then swap the two children of 20 afterwards

merge of 30 with

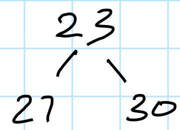
→

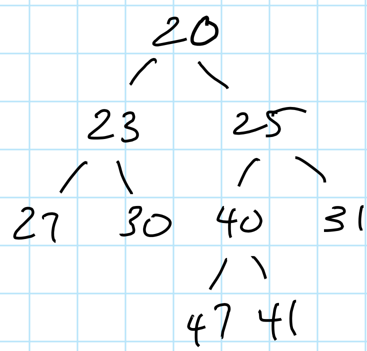


here we do <u>not</u> swap the two children of 23, since the root (23) was the maximal node on the rightmost path in



, i.e., no right child existed

→  first becomes the right child of 20, but then the two children of 20 are swapped

→ result: 

2.6. BINOMIAL QUEUES

<u>insertion:</u> leftist/skew h. $O(\log(N))$

binary h. avg. time $O(1)$

<u>merge:</u> leftist/skew h. $O(\log(N))$

binary h. $O(N)$

⇝ binomial queues insertion: $T_{avg}(N) = O(1)$

merge, deleteMin: $O(\log(N))$

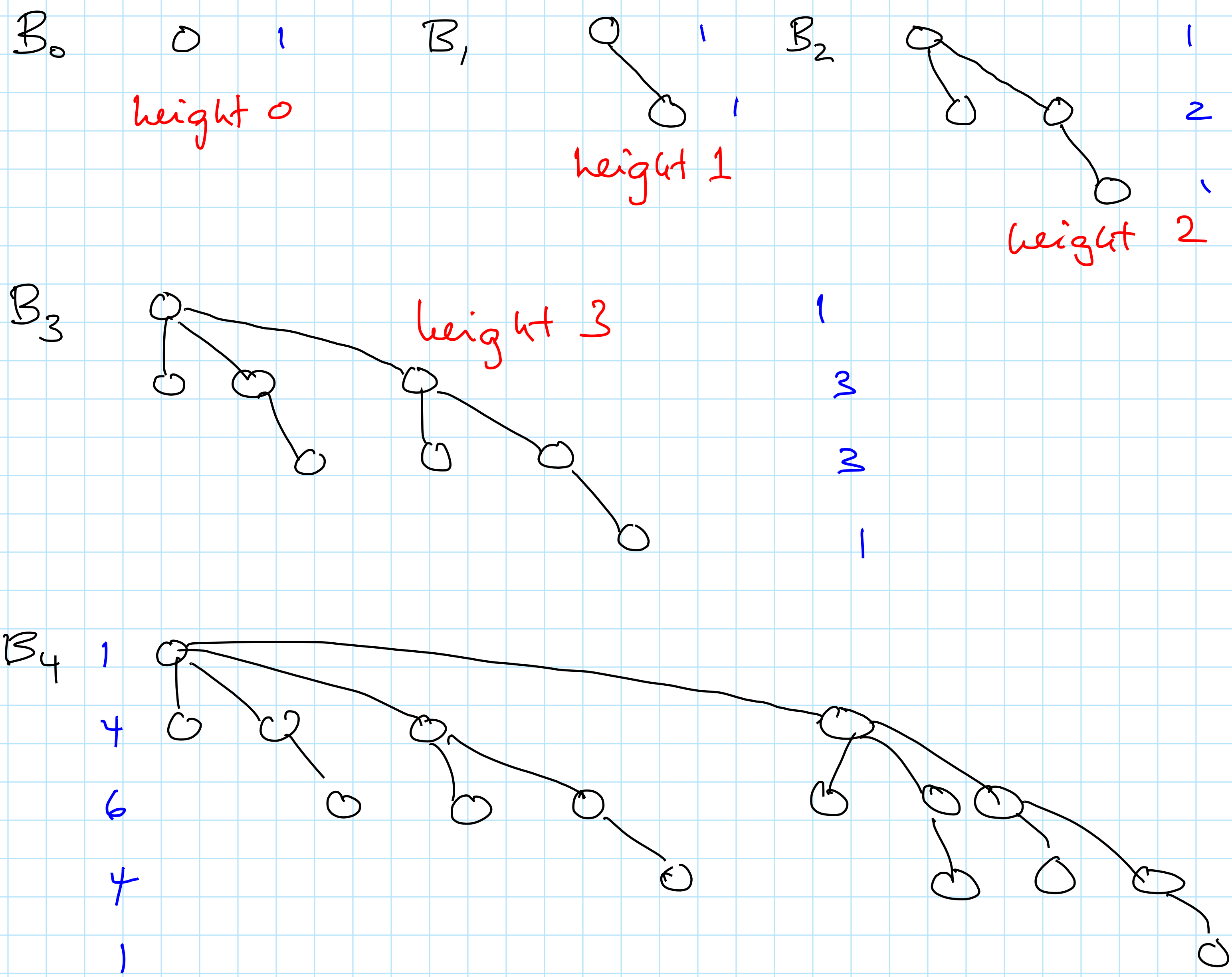
A binomial queue is a special collection of "binomial trees".

<u>Definition 8.</u>

B_0 is the set of trees consisting of one node.

For $k > 0$, B_k

For all $k \in \mathbb{N}$, any tree in B_k is called binomial tree of height k. (note: its height is k)



<u>Fact</u>

- The number of nodes at depth d in a binomial tree of height k is $\binom{k}{d}$
- Binomial trees of height k have exactly $\binom{k}{0} + \binom{k}{1} + \binom{k}{2} + \dots + \binom{k}{k} = 2^k$

(proof by induction)

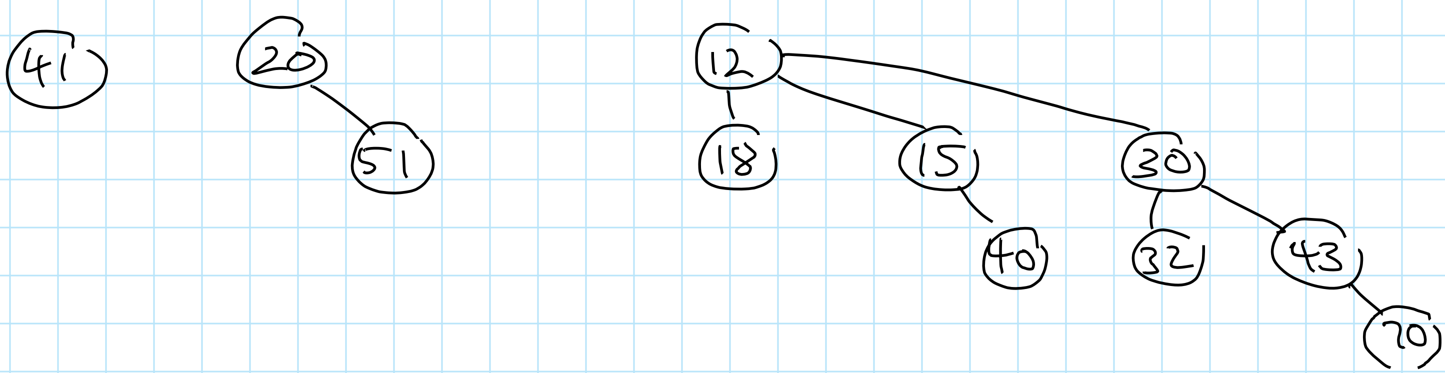
Definition 9. A binomial queue (BQ) of N elements, where $N = 2^{i_1} + 2^{i_2} + \dots + 2^{i_m}$ with $i_1 > i_2 > \dots > i_m$, is a collection consisting of one tree from B_{i_1}, one from $B_{i_2}, \dots$, one from B_{i_m}. Moreover, each of these trees has the heap-order property ($key(n) \geq key(parent(n))$ for all nodes n except the root.)

Example 29. A BQ of 11 nodes

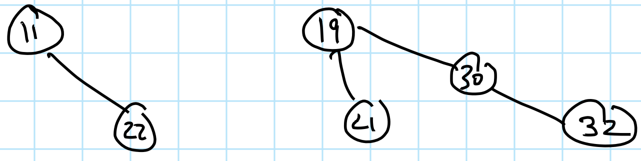
$11 = 8 + 2 + 1$ "1011"

$= 2^3 + 2^1 + 2^0$

$\rightarrow$ one tree from B_0, one from B_1, one from B_3



A BQ of 6 nodes:



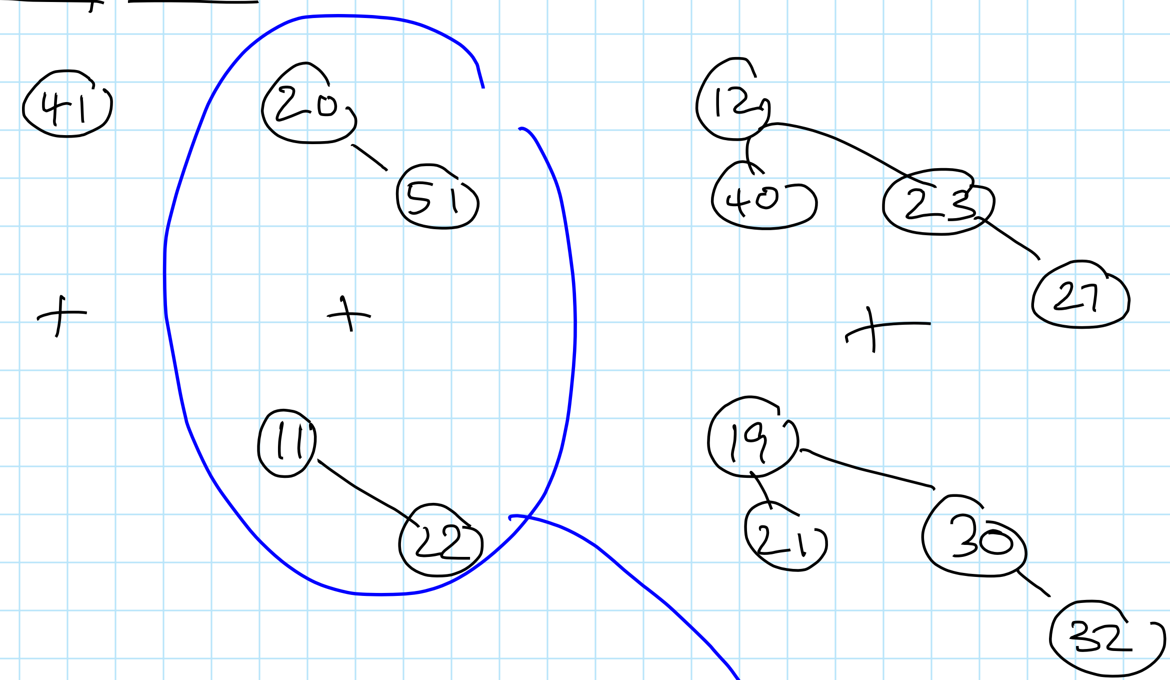
Fact. A BQ with N nodes consists of at most $\lceil \log(N) \rceil$ binomial trees.

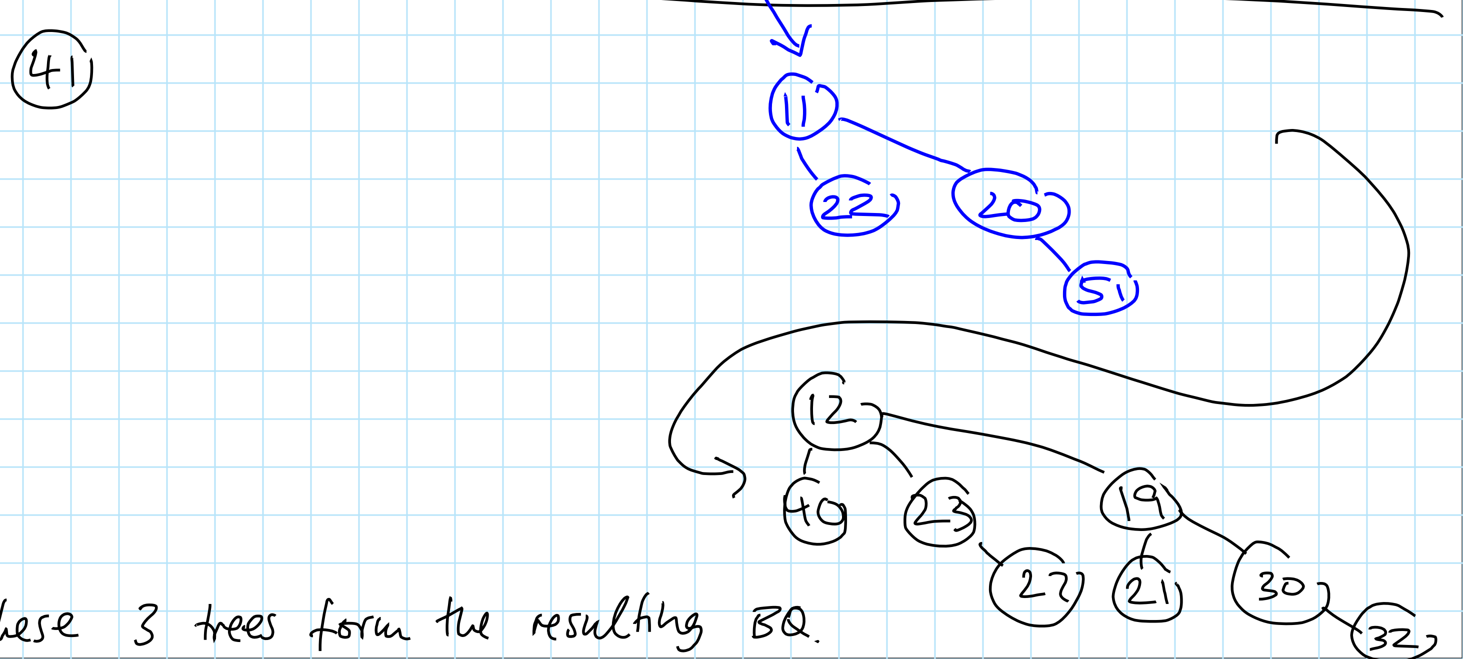
Operations

find Min : check roots of all trees $O(\log(N))$

merge : "add" queues by merging 2 trees of same height into one tree of next larger height (the tree with the larger root becomes a subtree of the other one)

Example 3





These 3 trees form the resulting BQ.

$T_{worst}(N)$: merge two trees: $O(1)$

$O(\log(N))$ many pairs of trees to merge

$\Rightarrow O(\log(N))$

watch out: keep trees in BQ sorted by height.

<u>insert</u> special case of merge $\rightsquigarrow T_{worst}(N) = O(\log(N))$

$T_{avg}(N) = O(1)$

- if a node is inserted into a BQ Q^* and k is minimal such that no tree from B_k occurs in Q^*. Then $T(N) \sim k+1$
- for every k, the probability of a tree in B_k being present in a fixed BQ is $\frac{1}{2}$.

expected (avg.) runtime of insertion?

1	with prob.	$\frac{1}{2}$	(no B_0)
2	–"–	$\frac{1}{4}$	(B_0, no B_1)
3	–"–	$\frac{1}{8}$	(B_0, B_1, no B_2)

$\vdots$

$$\rightsquigarrow \frac{1}{2}\cdot 1 + \frac{1}{4}\cdot 2 + \frac{1}{8}\cdot 3 + \frac{1}{16}\cdot 4 + \dots$$

$$= \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots\right) + \left(\frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \dots\right) + \left(\frac{1}{8} + \frac{1}{16} + \dots\right)$$

$$= 1 + \frac{1}{2} + \frac{1}{4} + \cdots = 2$$

$$= \Theta(1)$$