lecture 14 – Oct 05

insertion into BQs. $T_{worst}(N) = \Theta(\log(N))$

$T_{avg}(N) = \Theta(1)$

The amortized worst case running time of insertion is $O(1)$, i.e., a sequence of N insertions into an initially empty BQ takes $O(N)$ worst-case time.

(without proof) see assignment on amortized cost! (bit-counter)

deleteMin

1) find binomial tree B with smallest root in the BQ

   $T_{worst}(N) = \Theta(\log(N))$

2) $BQ_1 = BQ$ minus B $T_{worst}(N) = \Theta(\log(N))$.

3) BQ_2 = collection of trees resulting from cutting off root from B (→ this is a BQ!)

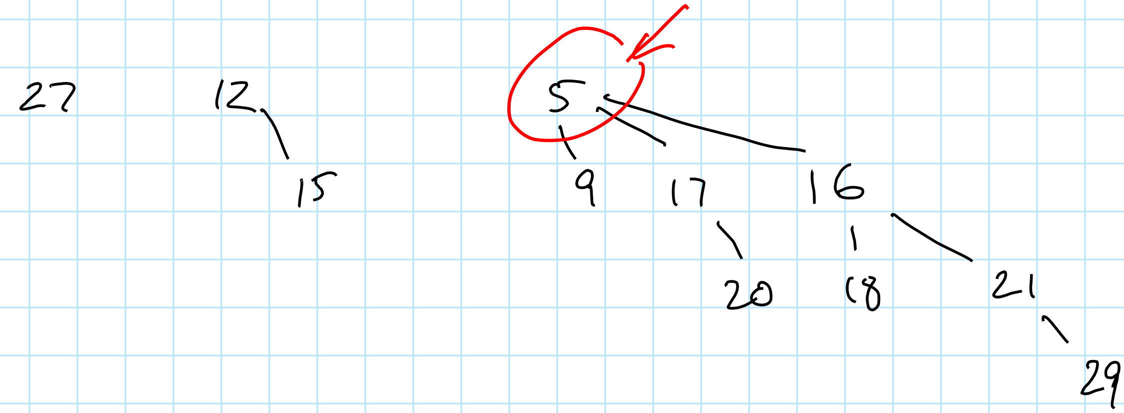
   $T_{worst}(N) = O(\log(N))$

4) $BQ_{result} = merge(BQ_1, BQ_2)$ $T_{worst}(N) = \Theta(\log(N))$

$\Longrightarrow T_{worst}(N) = \Theta(\log(N))$

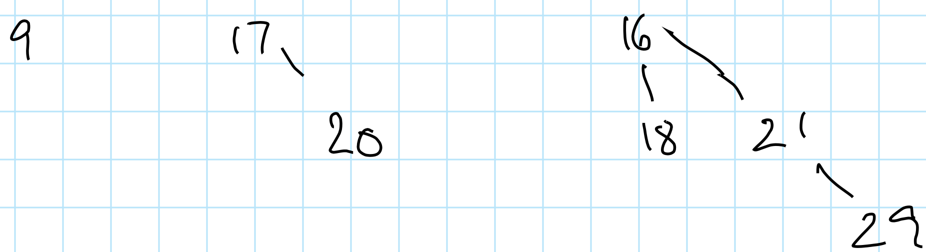
Example 31. deleteMin

BQ:

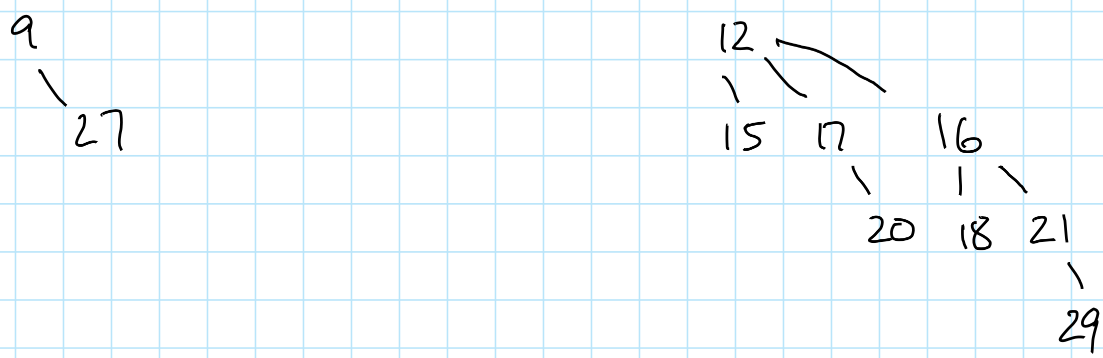


BQ_1

BQ_2



BQ_{result}



Notation for midterm

Omega (N^2) for $\Omega(N^2)$

Theta (N^3) for $\Theta(N^3)$

O(N)

o(N^4) for $o(N^4)$

2^k for 2^k

3. SORTING

problem given: array A of N elements from a set on which a total order is defined (e.g., array of integers)

task: return an array that contains the same elements as A, in increasing order

allowed operations:
- comparisons: $<$ $>$ $==$
- assignments: $=$

$\Rightarrow$ "comparison-based sorting";
in the analysis we count # comparisons

3.1. Insertion Sort

sort playing cards in your hands ...

$N-1$ passes; in pass i, $1 \le i \le N-1$:

- initially elements in positions 0 through $i-1$ are sorted ("the cards you are holding")
- then element in position i is inserted such that positions 0 through i are sorted.

the card you are picking up in this pass

your sorted hand after inserting the card just picked up.

Example 32.

									# comparisons
input		51 |	7	32	11	27	21	29	
after	i=1	7	51 |	32	11	27	21	29	1
	i=2	7	32	51 |	11	27	21	29	2
	i=3	7	11	32	51 |	27	21	29	2
	i=4	7	11	27	32	51 |	21	29	3
	i=5	7	11	21	27	32	51 |	29	3
	i=6	7	11	21	27	29	32	51 |	5
									16

see next lecture!