Order Statistics Doubt Clearing

I/P A n elements

partition A

O/P Kth order statistics (Kth smallest element)

Kth index in sorted order

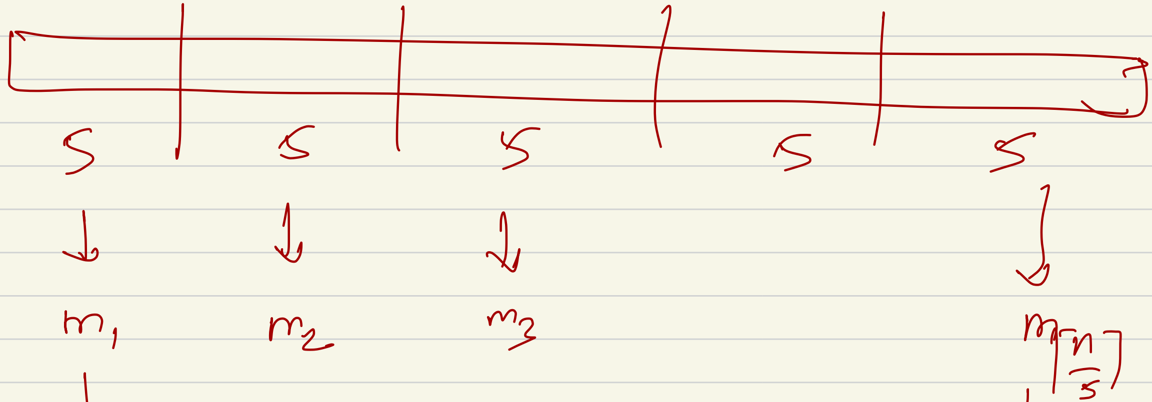
Quick Sort partition

Choose a pivot x

$\leq x$ | x | $> x$

O(n) partition

atleast some fraction of the original array



median of these $\lceil \frac{n}{5} \rceil$ medians

$\downarrow$ pivot x

ensures that each partition has atleast $\left(\frac{3n}{10} - 6\right)$ elements

$$T(n) \leq T\left(\frac{7n}{10}+5\right) + T\left(\frac{n}{5}+1\right) + O(n)$$

solve

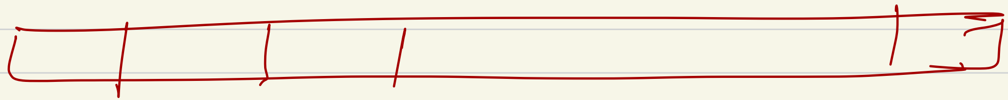
$$\boxed{T(n) = O(n)}$$

- Groups of 5 elements $\Rightarrow$ $T(n) = O(n)$

worst case

- 3 elements $\Rightarrow$ $T(n) = O(n)$?

- 7 elements



$\lceil \frac{n}{3} \rceil$ groups

3   3   3   ...   1-3

m_1 m_2 — - - $m_{\lceil \frac{n}{3} \rceil}$

median of $\lceil \frac{n}{3} \rceil$ medians

pivot x

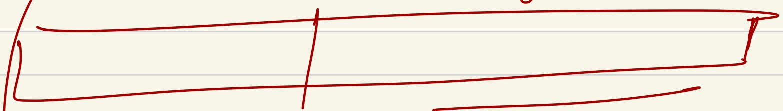
x : # of elements $< x$ = $\frac{n}{6}$

each group will have one element less than this

$\frac{n}{3}$ elements $< x$

$$\boxed{T(n) \le T\left(\frac{2n}{3}\right) + T\left(\frac{n}{3}\right) + O(n)}$$

getting



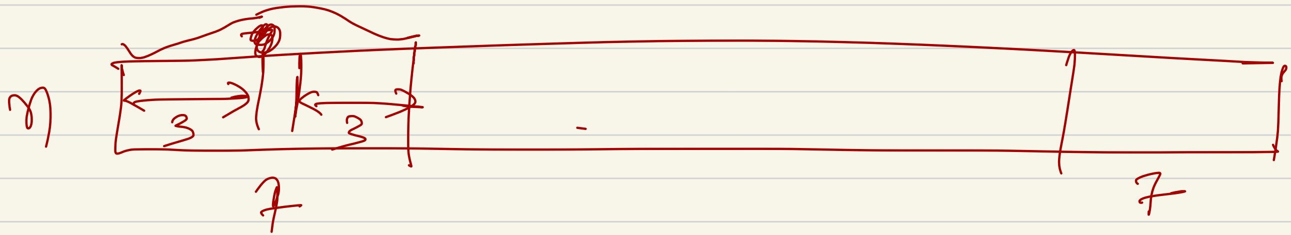
atleast $n/2$ atmost $\frac{2n}{3}$

$T(n) = O(n \log n)$

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + O(n)$$

$T(n) = O(n \log n)$ $\quad T(n) = T\left(\frac{99n}{100}\right) + T\left(\frac{n}{100}\right) + O(n)$

Groups of size 7



m_1 — — - $m\left(\frac{n}{7}\right)$ medians

median of $\frac{n}{7}$ medians

$=$ pivot x

How many elements are less than x? $\rightarrow$ $\frac{n}{14}$ medians

For each of the median, 3 elements $< x$ $\Rightarrow$ $\frac{4n}{14}$ elements $< x$

$\frac{2n}{7}$ elements $< x$

$\Rightarrow$ atmost $\frac{5n}{7}$ elements in any of the subarray.

throwing away more elements !!

Algo 2 : $T(n) \le T\left(\frac{5n}{7}\right) + T\left(\frac{n}{7}\right) + \boxed{O(n)} \le bn$

$\left(\frac{n}{7}\right)$

$\frac{6n}{7}$ ~ throwing away $\frac{n}{7}$ every time

Algo 1 : $T(n) \le \underbrace{T\left(\frac{7n}{10}\right) + T\left(\frac{n}{5}\right)}_{\left(\frac{9}{10}n\right)} + \boxed{O(n)} \le cn$

$\left(\frac{n}{5}\right)$

$\boxed{b > c}$

throwing away $\frac{n}{10}$ every time

$O(n)$

1. Find the median of all groups

$5 \rightarrow \frac{n}{5}$ medians each of 5 elements

$\left[5 \log 5\right] * \frac{n}{5}$

vs

$\log 5$

7 $\quad \frac{n}{7}$ medians each of 7 elements

$\log 7$

$\left[7 \log 7\right] * \frac{n}{7}$

$\log 9$

5 is the optimal

Why Order Statistics

ith Smallest elements

Sorting

Searching

array A of n elements

$a_1 \leq a_2 - - a_n$

find out an key 'x'

Order Statistics

Give me the ith smallest elements

Can you give me a worst-case $O(n\log n)$ quickSort algo?

→ You can find median in $O(n)$ time

→ Find median & use that as pivot & then do partitions

→ $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$

Can you give me Largest i elements in the array in their sorted order? / just the bag?

$O(n)$ time

x' $> x'$

approp. index

order statistics gives a linear time algo

A of n elements

Top i largest elements in sorted order?

If i need sorted o/p

— Sorting & take the last i → $O(n \log n)$

— Use Heaps → $O(n + i \log i)$

$O(n)$ → construct heap

Find i largest element → $\{i \log i\}$

Order Statistics ↔

If we don't need sorted o/p → $O(n)$ + $i \log i$

Finding max. & min. simultaneously

$\frac{3n}{2}$ Comparisons compared to $2n-2$

You are given an array of n elements

You've to find the second largest elements.

What is the min. number of comparisons you need?

→ largest element = $n-1$ comparisons

largest elements

Think like a tournament

n teams playing

comparison is a match

one team wins

another looses → out of the tournament

for a winner,

other n-1 teams have to loose

⇒ atleast n-1 games

2nd largest how many comparisons?

$= n + \log n - 2$

$n + \log n - 2$??

You have to play $n-1$ comp. to find the largest

2nd largest → It would have lost to the largest at some point in the tournament

Find all elements that lost to the largest in tournament

x

⇒ $x-1$ comparisons

Find the largest among these

kth' → kn comparisons

$n-1$ + $x-1$

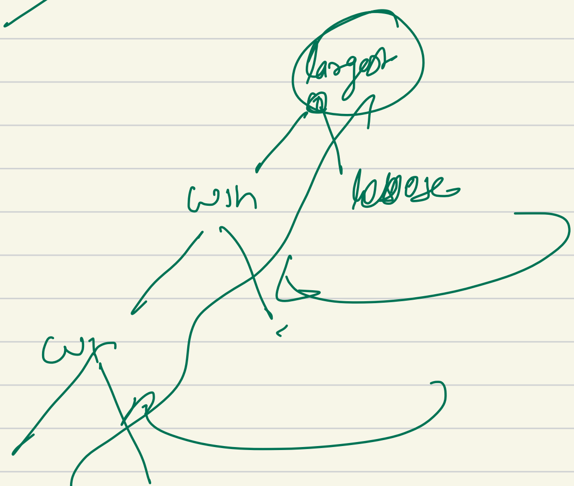
largest   2nd largest

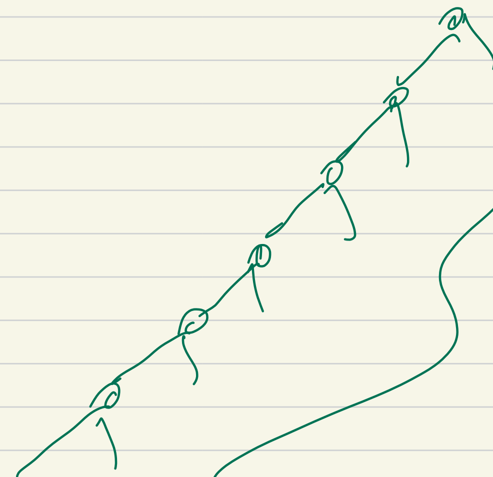
→ # of elements that lost to the largest

2nd

$n+1$ $n-2$

minimize x





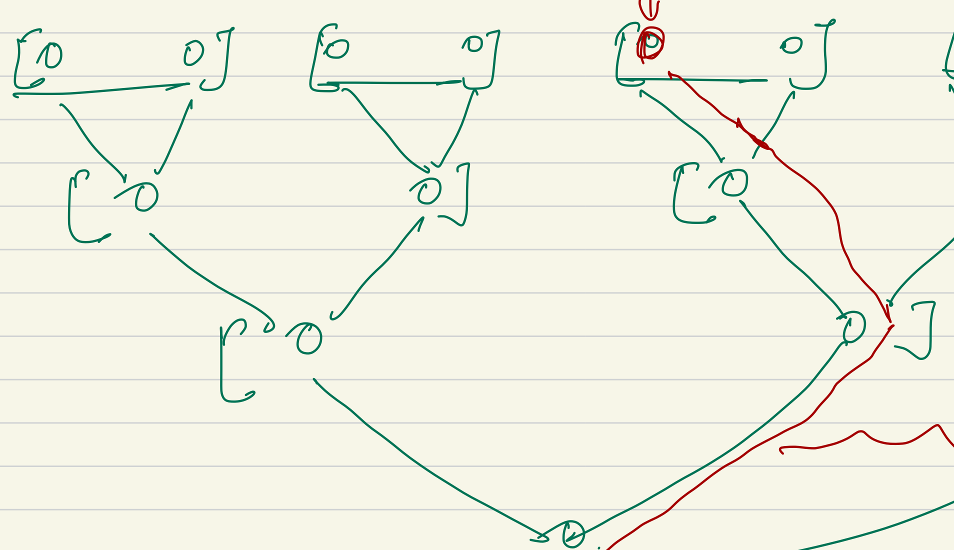
$x = n-1$

naive strategy

$n = 2^m$

winner

take this path from winner (leaf) to root



balanced binary trees

(log n)

log n other elements you're to find largest from

2nd largest

n nodes $\Rightarrow$ $n-1$ edges $\Rightarrow$ games

$x = \log n$

$(n-1 + \log n - 1)$

What is the height of the tree?

$h = O(\log n)$