Sep 11th

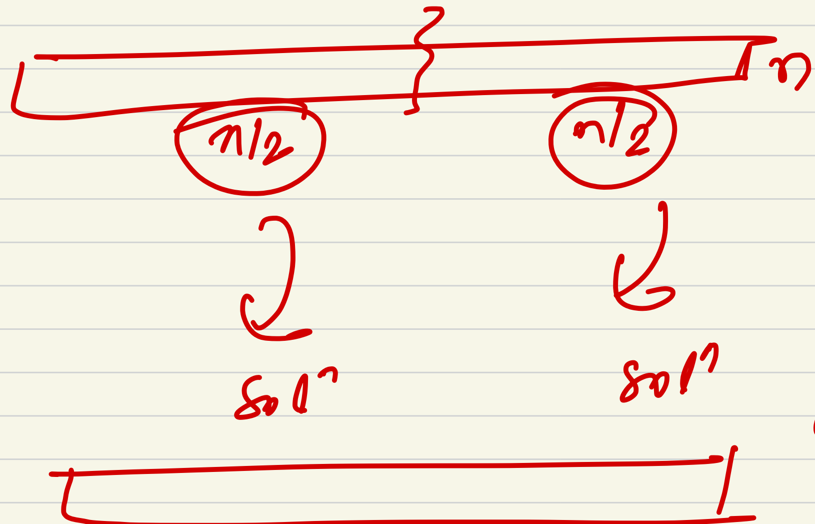
# Algo-1

## Divide & Conquer

Algo-Design

Problem instance size $n$

Merge Sort




Divide

Conquer

Combine

Sol$^n$ for the overall problem

Quick Sort is not a stable sorting algo

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 6' | 6'' | 3 | 2 | 1 | 8 | 3 | 9 | 11 |

1 2

pivot

6' < 6''  (final sorted array)

Sorting beforehand

[Comparison based sorting
lower bound - $\Omega(n \log n)$]

$P_x$ ] $P_x$ X, Y , index in sorted array (X)

index in sorted array (Y)

[0 ... n/2] n/2 $P_y'$

$P_x$ : sorted as per X

size n

$P_y$ : sorted as per Y

$P_x''$ $P_y''$

$P_x'$ $P_y'$

n/2

$P_x$

middle

0 n/2 conquer n/2 n/2 + 1 n

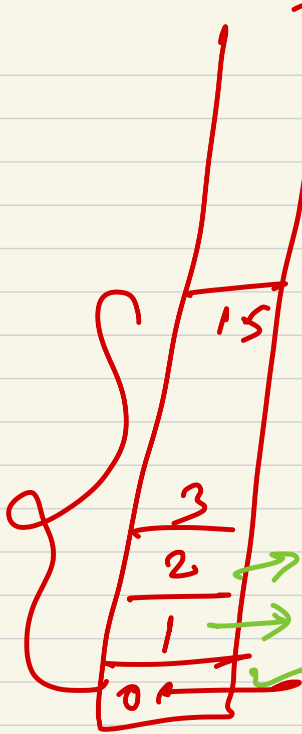
closest pair closest pair

15 next points

lower most

15n comparisons at most

sorted as per Y




left/right

left/right

sorted array

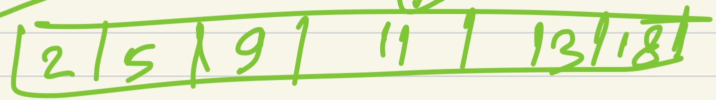
at most 15n comp.

leftmost

compare with other side

c.n

Binary Search




X=2

sorted array

Search

sorted array

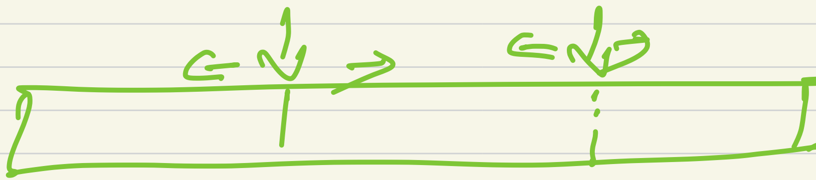
unsorted array

number X

are there two elements in the array that will add up to X

one of the sub-array

Ternary Search



$T(n) = T\left(\frac{n}{3}\right) + c$

$O(\log_3 n)$

$O(n \log n)$ algo — $\{O(n)$ algo$\}$

Binary Search

Merge Sort

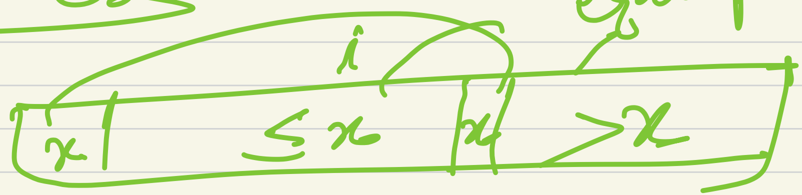
Quick Sort

Worst Case $\Theta(n^2)$

Average Case $O(n \log n)$ $\Theta(n \log n)$

Best Case

equal partition every time




$$T(n) = T(i) + T(n-i) + cn$$

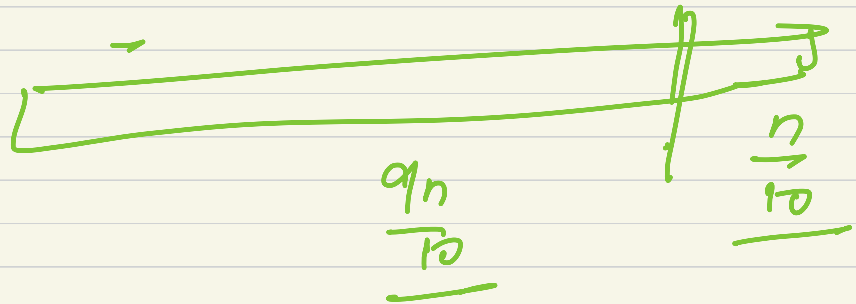
$$T(n) = 2T\left(\frac{n}{2}\right) + cn$$

$\rightarrow \Theta(n \log n)$ Merge Sort

$$T(n) = \underbrace{T(n-1) + T(n-2)}$$

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

<u>Skew</u>




$\Theta(n^2)$

$O(n \log n)$

$O(n \log n)$

Avg case $\Theta(n \log n)$

$$n \qquad\qquad cn$$

$$\frac{n}{10} \qquad \frac{9n}{10} \qquad\qquad cn$$

$$\frac{n}{100} \qquad \frac{9n}{100} \qquad \frac{9n}{100} \qquad \frac{81n}{100} \qquad\qquad cn$$

height of recursion tree

$$\cancel{\frac{n}{1000}} \qquad\qquad \le cn$$

$$\le cn$$

$$\le cn \cdot \underline{\text{Height}}$$

$$\log_{\frac{10}{9}} n$$

$$\left[ cn \log_{\frac{10}{9}} n \right]$$

Two pair of Jugs

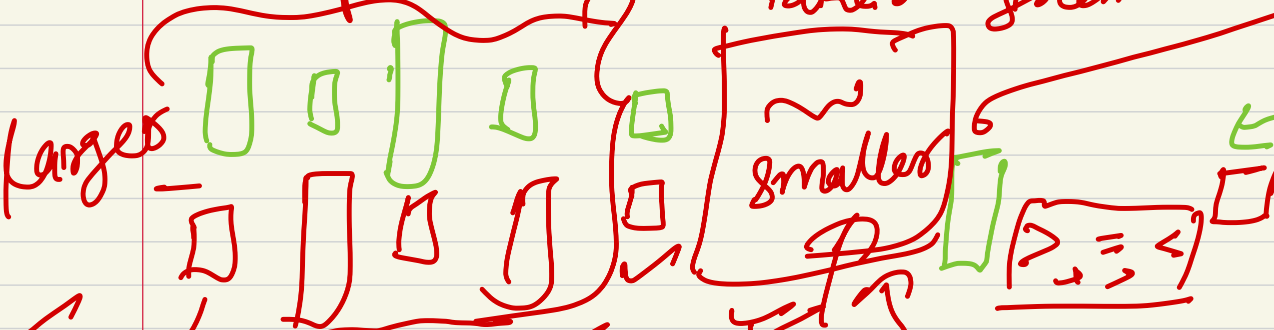
$n$ Green

$n$ Red ~~Red~~

Operation

take a green

Take a red




Match these pair of Jugs with eachother

Quick Sort

$$\boxed{T(n) = T(n-i) + T(i-1) + O(n)}$$

An array of n elements

$a_0$ — — $a_{n-1}$

3 5 12 9 2 1 6

$O(n\log n)$ time

Dominating value

atleast $\frac{n}{2}$ elements have that value

$O(n^2)$

oper^n: → Compare two elements for equality

[ 3 2 2 1 2 2 4 ]

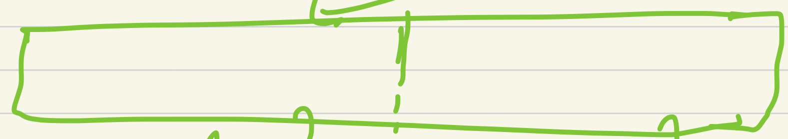
Find the dominating value

4 times

# Divide & conquer

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

✓ Divide :



✓ Conquer .

A : L R

- dom. elem X
- No dom. elem

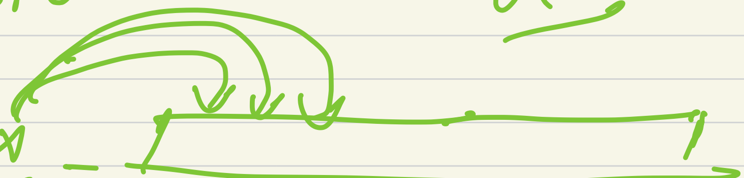
- dom. elem. Y
- No dom. elem.

✓ Combine : —

Recurrence Rel$^n$

O(n) time

Case 1: L gives dom. X
R doesn't



n > n/2 elem

Case 2 L doesn't
R gives Y

Case 3 - Neither X nor Y give
→ No dom.

Yes

Case 4. Both L & R