①

RECURSIVE ALGO Ⓐ

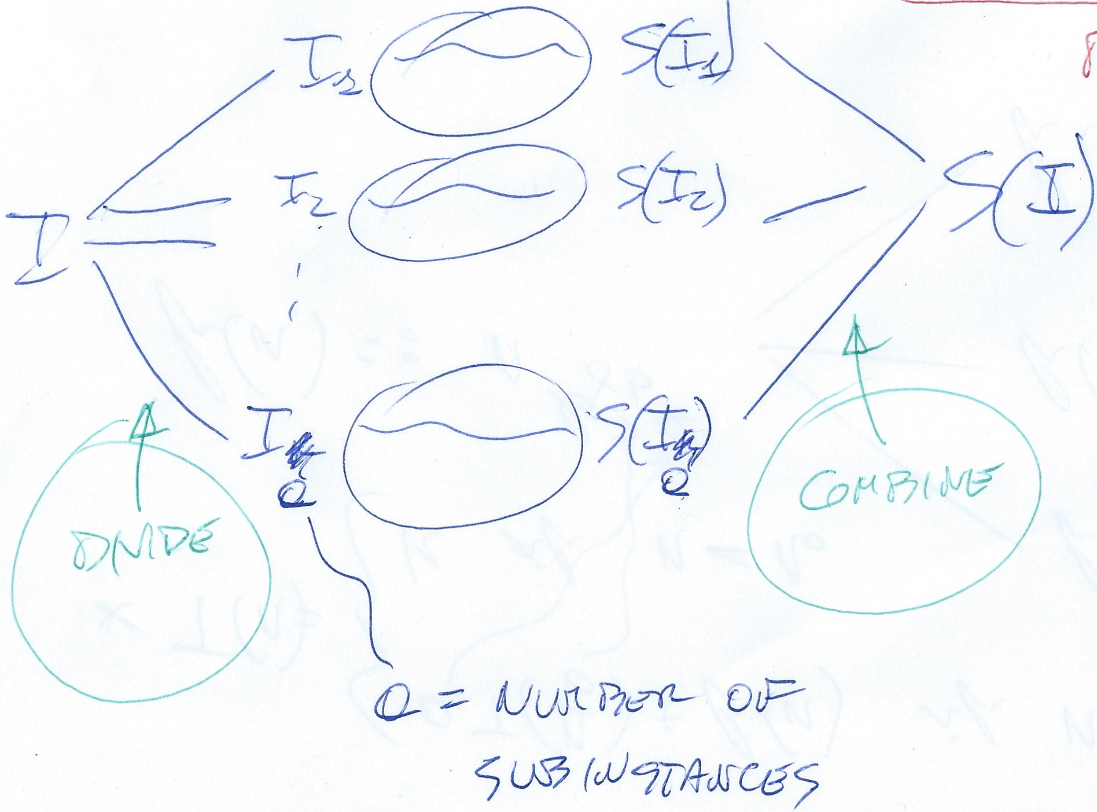
IF → BASE CASES

REC CALLS → SMALLER INSTANCES

COMPLEXITY?

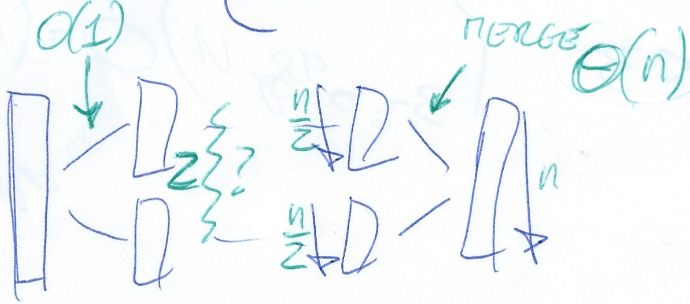
$$T_A(\underbrace{n}_{\text{INPUT SIZE}}) = \begin{cases} \underbrace{k \quad \text{if } n < c}_{\text{BASE CASE}} \\ \underbrace{f(n)}_{\text{OTHER}} + \underbrace{a \cdot T_A\left(\frac{n}{b}\right)}_{\text{REC. CALLS}} \quad \text{if } n > \end{cases}$$

RECURRENCE RELATION (EQUATION)



MERGE SORT

$$T_{MS}(n) = \begin{cases} k \quad \text{if } n < 2 \\ \Theta(n) + 2 \cdot T_{MS}\left(\frac{n}{2}\right) \end{cases}$$



②

MASTER THEOREM FOR REC. EQ.

$$T(n) = \begin{cases} k & \text{if } n < c \\ f(n) + a\,T\left(\frac{n}{b}\right) \end{cases}$$

$f(n)$ ∷ ? $\cancel{n^{\log_b a}}$ $n^{\log_b a}$

NON REC TERM

RECURSIVE TERM

$f(n) = O\left(n^{(\log_b a) - \varepsilon}\right), \ \varepsilon > 0$

GROWS LESS THAN

$\hookrightarrow$ $\cancel{T(n) = \Theta(f(n))}$

$T(n) = \Theta\left(n^{\log_b a}\right)$

$f(n) = \Theta\left(n^{\log_b a}\right)$

$\hookrightarrow T(n) = \Theta\left(n^{\log_b a} \cdot \log n\right)$

$f(n) = \Omega\left(n^{\log_b a + \varepsilon}\right), \ \varepsilon > 0$

GROWS MORE THAN

$\hookrightarrow T(n) = \Theta(f(n))$

$\boxed{\text{MERGE SORT}}$

$$T_{MS}(n) = \Big\{ \alpha \cdot n + 2\,T_{MS}\left(\frac{n}{2}\right)$$

$f(n)$ a b

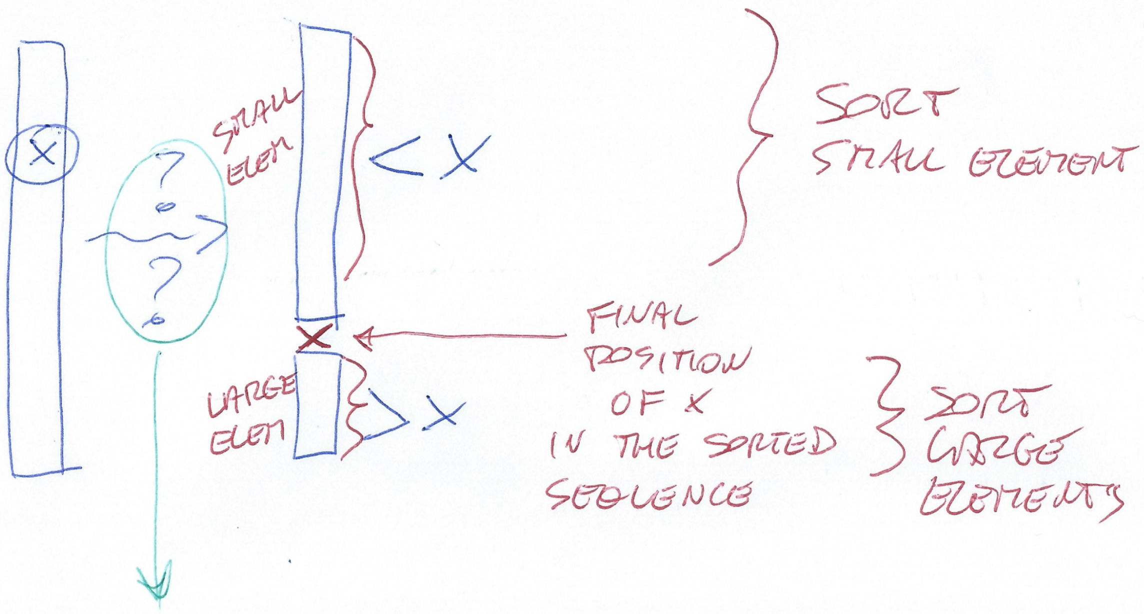
$f(n) :: n^{\log_b a}$

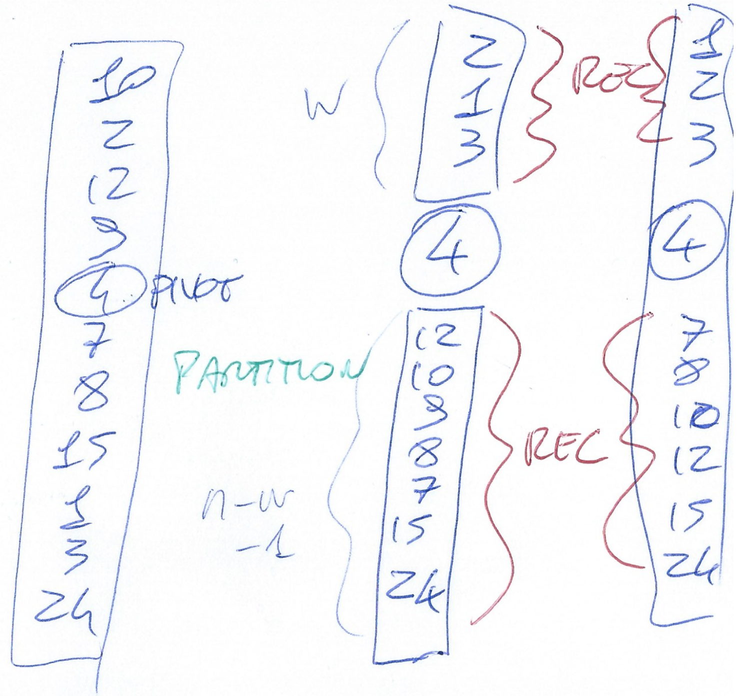
$n :: n^{\log_2 2} = n^1$

SAME FUNCT. (ASYMPT)

$T_{MS}(n) = \Theta(n \cdot \log n)$ OPTIMAL $\rightarrow$ EQUALS THE INTRINSIC COMPL. SORTING

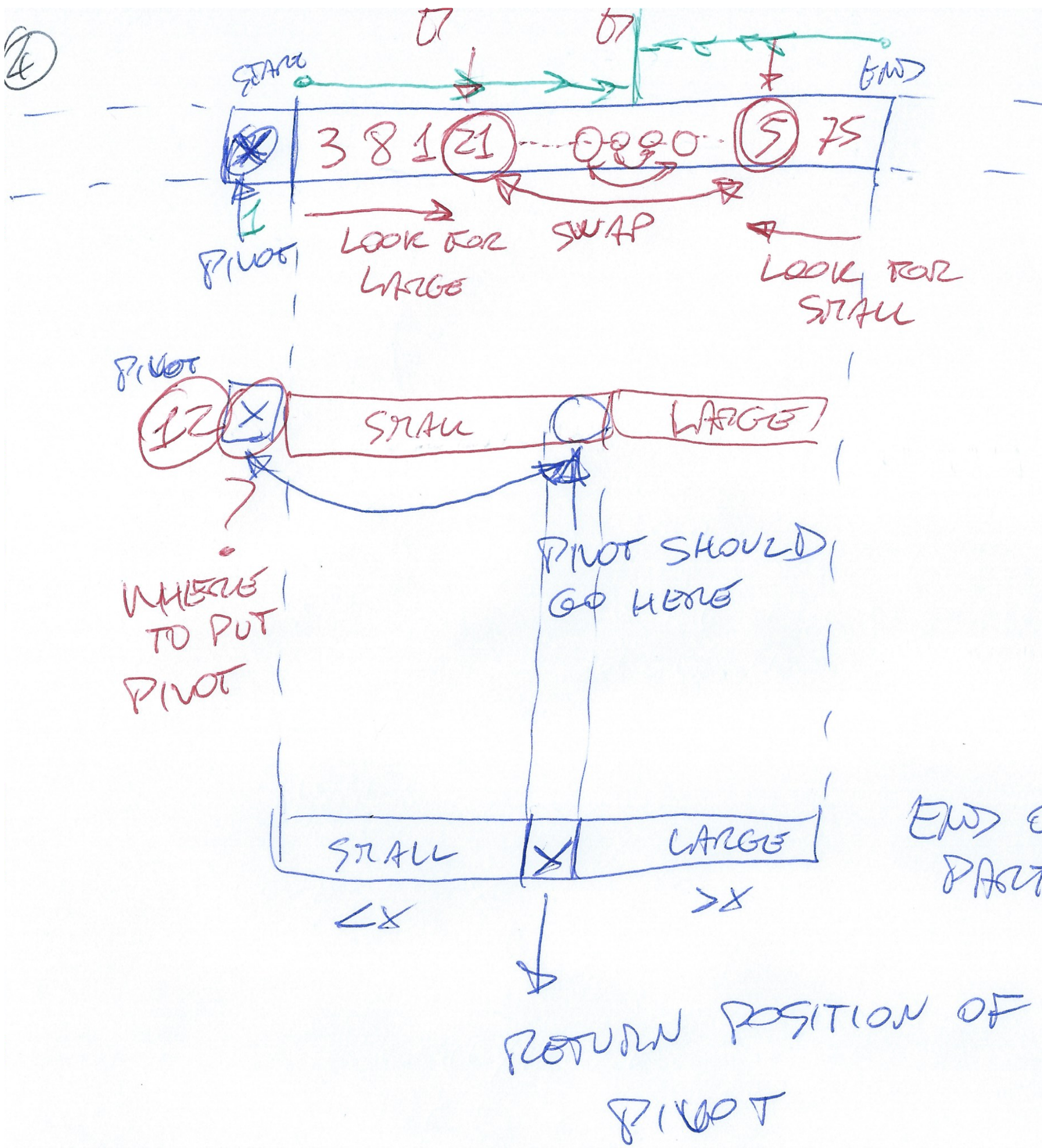
QUICK SORT (DISTINCT ELEMENTS)





- CHOOSE AN X PIVOT
- SEPARATE SMALL/LARGE ELEMENTS (REARRANGE)
- PUT PIVOT IN BETWEEN

4



END OF PARTITION

PARTITION PROCESS

COMPLEXITY ?

OF PARTITION ON n ELEMENTS

$\Theta(n)$

$\boxed{\text{NO EXTRA SPACE}}$

5

COMPLEXITY OF QUICK SORT (NO RANDOMIZATION)

PIVOT IS THE FIRST ELEMENT

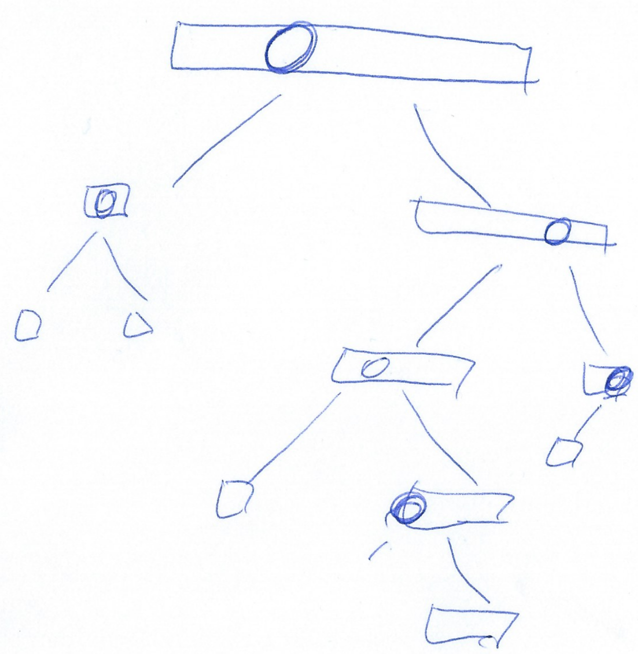
$$T_{QS}(n) = \begin{cases} k & \text{if } n \leq 1 \\ T_{QS}(w) + T_{QS}(n-w-1) + \Theta(n) \end{cases}$$

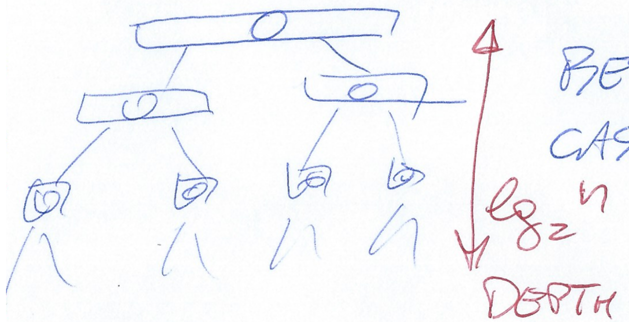
NUMBER OF SMALL EL. REC. CALL

LARGE ELEM REC CALL

PARTITION

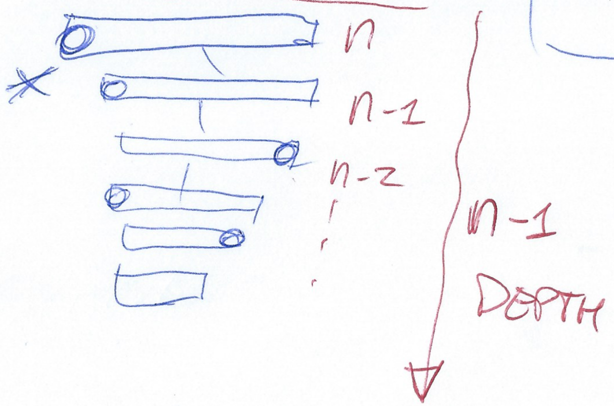
RECURSION TREE





BEST CASE

WORST CASE



⑥ QUICK SORT BEST CASE $\Theta(n \lg n)$

$$T_{QS}(n) = \begin{cases} k & \text{if } n \le 1 \\ \Theta(n) + 2\,T_{QS}\left(\frac{n}{2}\right) \end{cases}$$

THE SAME AS MERGE SORT

$f(n) = \Theta(n)$

$a = 2$

$b = 2$

$T_{QS}(n) = \Theta(n \lg n)$

BEST CASE

QUICK SORT WORST CASE $\Theta(n^2)$

$$T_{QS}(n) \begin{cases} k & \text{---} \\ n\alpha + \underbrace{T_{QS}(n-1)}_{} \end{cases}$$

$(n-1) + \underbrace{T_{QS}(n-2)}_{}$

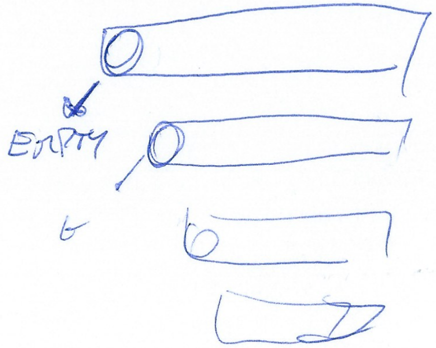
$(n-2) + T_{QS}(n-3)$

$(n-3) + \text{---}$

$$T_{QS}(n) = \sum_{i=n}^{1} \alpha i = \alpha \frac{n(n+1)}{2} = \Theta(n^2)$$

7

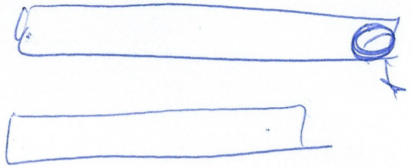
WORST CASE QUICK SORT



SEQUENCE ALREADY SORTED

$\Theta(n^2)$

REVERSED SEQUENCE ≈

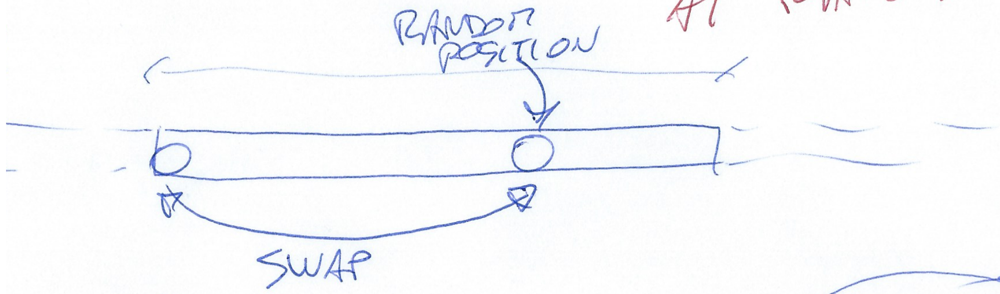


$\Theta(n^2)$

WHAT TO DO?

PARTITION RANDOMIZATION

CHOOSE PIVOT AT RANDOM



EXPECTED COMPLEXITY RANDOMIZED QUICK SORT

$\Theta(n \cdot \log n)$

PSEUDO RANDOM NUMBER GENERATOR

```
import random
⋮
= random.randint(a,b)
```

RANDOM INTEGER IN [a - - b]

INCLUDED