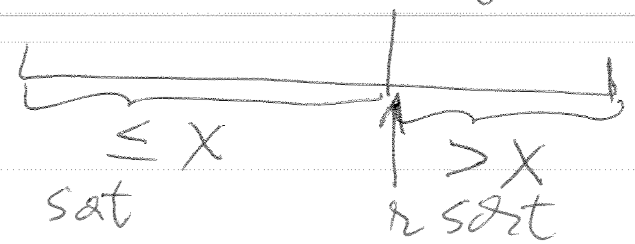
COMP 250 - Lecture 17

QuickSort

MergeSort → divide & conquer, recursive
fast $O(n \log n)$, not in-place → $O(n)$

QuickSort: in-place ($O(1)$ extra mem)
"roughly" same running time

Array a



Algorithm qsort (a, p, q)
Input: array a, indices p, q
Output: a is sorted between p and q
//Call with qsort (a, 0, n-1)

if $p < q$
  int r ← partition(a, p, q)
  qsort (a, p, r)
  qsort (a, r+1, q)

Correctness: induction (for qsort) + loop invariant for partition.

Algorithm partition (a, p, q)
Input: array a, indices p, q
Output: index r between p and q s.t.
$\forall i \le r$, $a[i] \le x$
$\forall j > r$ $a[j] > x$ for some x

$x \leftarrow a[p]$

p 5 3 2 8 1 2 7 6 q

$\underbrace{\qquad\qquad}_{\le 5}$ $\underbrace{\qquad}_{> 5}$

// Indices
$i \leftarrow p - 1$
$j \leftarrow q + 1$

while true do

5 3 2 8 1 2 7 6

i   j

$k < i$

repeat
  $i \leftarrow i + 1$
until $(\underbrace{i > q}_{\text{out of bounds}}$ or $\underbrace{a[i] \ge x}_{\text{need to swap } a[i] \text{ into 2nd half}})$

repeat
  $j \leftarrow j - 1$
until $(\underbrace{j \le p}_{\text{out of bounds}}$ or $\underbrace{a[j] < x}_{\text{need to swap } a[j] \text{ into 1st half}})$

$k > j$

if $i < j$ then swap $(a[i], a[j])$
else return j

after swap $a[i] \le x \le a[j]$

$i \nearrow$ $j \searrow$ at some point they cross $\rightarrow$ return p q

5 3 2 8 1 2 7 6

i i j j

$x \leftarrow a[p] = 5$

$i \leftarrow i+1$ (moving right) $a[i] \geq x$

$j \leftarrow j-1$ $a[j] < x$

2 3 2 8 1 5 7 6

i → i j j

2 3 2 1 8 5 7 6

i j i move to 8 } "crossed"

j 1 } return

Return when i and j are crossed

0	1	2	3	4	5	6	7
2	3	2	1	8	5	7	6

i

j = 3 returned

Recursive call: 1 2 2 3 5 6 7 8

Done!

Suppose partition works correctly.

If it returns index r,

$\forall\, i \le r, \quad a[i] \le X$

$\forall\, j > r, \quad a[j] \ge X$

Proof by induction:

Base case: array of 1 element

$qsort(a,0,0) \rightarrow$ nothing Yay!

Induction step: Assume recursive call works correctly.

$\forall\, p \le i < j \le r \quad a[i] \le a[j]$ (ind hyp)

$\forall\, r+1 \le i < j \le q \quad a[i] \le a[j]$

$\forall\, p \le i \le r,\ r+1 \le j \le q \quad a[i] \overset{?}{\le} a[j]$

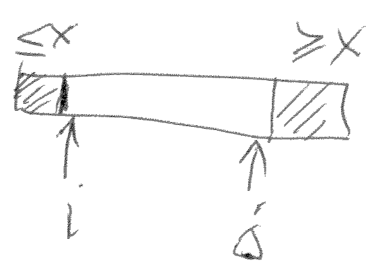
If partition is correct $\quad a[i] \le X \le a[j]$

Done!

Loop invariant:

$\forall\, K \ge p,\ K < i, \quad a[K] \le X$

$\forall\, K \le q,\ K > j \quad a[K] \ge X$



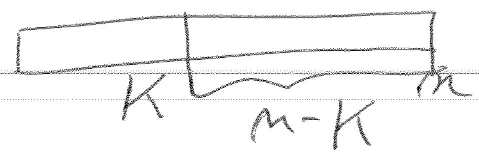
Running time analysis

partition $O(n)$ ($i \nearrow$ touching each el.

$j \searrow$ touching each el

stop when we are out of bound & cross

$$T_{partition}(n) \le 2n$$



$$T(n) = \underbrace{T_{partition}(n)} + T(K) + T(n-K)$$

$$\text{or} \begin{cases} = c \cdot n + T(1) + T(n-1) & \text{worst case} \\ = c \cdot n + T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) & \text{"best" case} \end{cases}$$

Worst-case recurrence:

$$T(n) = c \cdot n + T(1) + T(n-1) =$$

$$= c \cdot \underline{n} + T(1) + T(1) + T(n-2) + c(\underline{n-1})$$

....

$\in O(n^2)$ (see Selection Sort)

Happens if array is already sorted (degenerate)

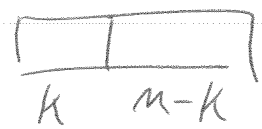
$$T(n) = cn + T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right)$$

$$= cn + 2T\left(\frac{n}{2}\right) \rightarrow \text{Same as MergeSort!}$$

$O(n \log n)$!

<u>Average?</u> Make an assumption on size of cuts

e.g. K "uniformly"



$$T(n) = cn + \underbrace{\frac{1}{n} \sum_{k=1}^{n} \left[T(k) + T(n-k) \right]}_{E[T] \text{ assuming } K \sim \text{Unif to } n}$$

$\cong 2n \ln n$ (COMP-251)

$\hookrightarrow \cong 1.4\, n \log_2 n$