Order Statistics: MEDIAN

Note Title 10/8/2007

if list size is odd,
median is $\lceil n/2 \rceil$ th element
in a sorted version of the list.

1 2 3 (4) 5 6 7
2 2 7 1 4 4 (3)

if list size is even,
then there are two medians:

lower median: $\lceil n/2 \rceil$th element
in a sorted version
of list.

upper median: $\left\lceil \frac{n+1}{2} \right\rceil$th element
in a sorted version
of list.

1 2 (3) (4) 5 6

2 7 (4) 1 6 (5)

When we talk about median, we will be talking about the 'lower median'

or $\lceil n/2 \rceil$th element in a sorted version of the list.

(list size even or odd)

How to find the median?

1. Sort list with Merge Sort.
2. Return the $\lceil n/2 \rceil$th element.

Runtime in terms of comparisons: $n \log n + c$

Can we do better?

Inspiration from Min? Max?

- If we looked for minimum value, discarded it, looked for next minimum value, etc. $n/2$ times we would get median.

- How much time would this take?

  $O(n^2)$

↓

- This is as bad as sorting!

- Using divide + conquer did not lead to any improvements in Min or Max, but did lead to some improvement for MinMax.

What if we try divide and conquer?

Is median finding easy when the list size is small?

When list size = 3
($n = 3$)

median can be found with this algorithm:

1. Find min. Discard.
2. Find max. Discard.
3. Remaining element is median!

This is $2(n-1) = 4$ comparisons.

Can be done in $\frac{3n}{2} - 2 \approx 3$ comparisons. (minMax)

Note: $a R b$, $b R c$, $c R a$ } We need only know these three relations (comparisons).

Median Finding Divide and Conquer*

1. Split list into sublists all of size 3.

2. Find median of each sublist.

3. Take list of medians and find the median of that list.

Is this correct??

*Thanks Evan!

Let's try this algorithm out:

1 3 (5) 2 4 6 7 8 9

actual median: 5   Median-3

median computed by algorithm: 4

split into sublists

135   246   789

Median: 3   Median: 4   median: 8

resulting list of medians: 3 4 8

↑

median computed by algorithm: 4

"Median of medians" approach does not work!

But can we use the median of medians for something?

Related question: Can we verify that we found a median?

example:

2 6 5 12 11 10.

actual median: 6

"median of medians": 5
(when breaking list into 3-element sublists). Let's call this MoM_3.

IDEA: use MoM_3 as a pivot!

- Partition list based on pivot

| 2 | 5 | 6 12 11 10 |

$< MoM_3$ $\quad$ MoM_3 $\quad$ $> MoM_3$

- where is median? It is in the BIGGER sublist.

Let's use this fact to define a recursive median-finding algorithm.

Let's define

SELECT(list, i)
// returns ith value in what would be the sorted version of the list. (The list is not actually sorted!)

So to find median, we call SELECT(list, $n/2$)

(assume n even.)

SELECT(list, 1) finds min.
SELECT(list, n) finds max.

So, what happens when we call
SELECT({2, 6, 5, 12, 11, 10}, $\frac{6}{2}$)? → 6

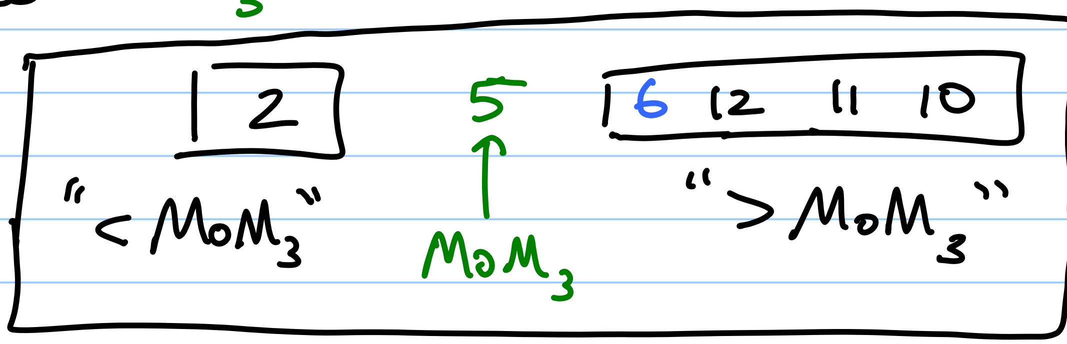
Example input: 2 6 5 12 1 1 10

SELECT (list, i) {

// i = position of desired value.

① Compute MoM_3.

example:



② Partition around MoM_3 and let pos_{MoM_3} = position of MoM_3 after partition.

③ If $i > pos_{MoM_3}$,

call SELECT ("> MoM_3" list, $i - pos_{MoM_3}$)

If $i < pos_{MoM_3}$,

call SELECT ("< MoM_3" list, i)

If $i == pos_{MoM_3}$,

return MoM_3.

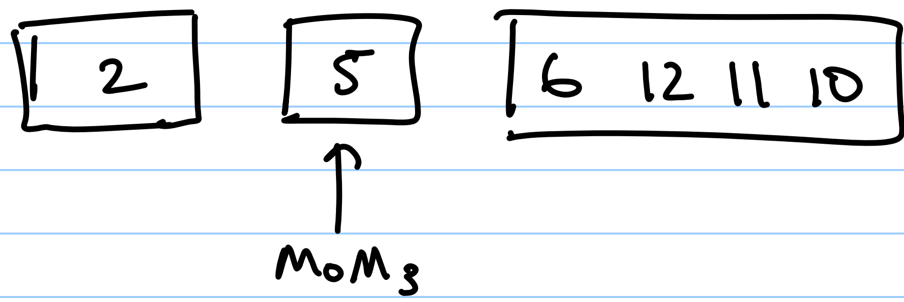
Note: We are not necessarily looking for the median in the recursive call!

Concrete example:

① SELECT({2, 6, 5, 12, 11, 10}, $\frac{6}{2}$)

i = position of desired value

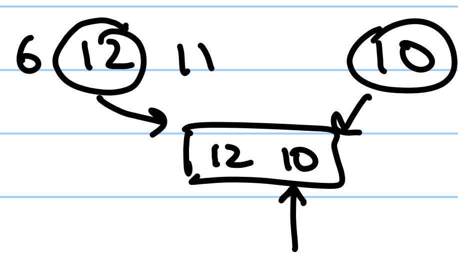
$= \frac{6}{2} = 3$ initially



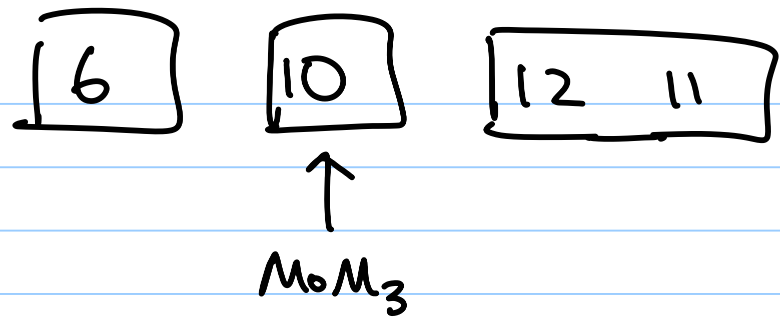
pos_{MoM_3} = 2

$(i > pos_{MoM_3})$, so

② SELECT ({6, 12, 11, 10}, 3 − 2)



10 is MoM_3.



$i = 1$

$pos_{MoM_3} = $ 2

$i < pos_{MoM_3}$,

so ③ SELECT({6}, 1)

$MoM_3 = 6$

$pos_{MoM_3} = 1$

$i == pos_{MoM_3}$, so return 6!

SELECT (list, i) {

// i = position of desired value. |x x x|x x x|

① Compute MoM_3.

↳ by calling SELECT(list of medians, $\frac{n}{6}$)

(Found w/ Median-3) ($\frac{n}{3} \cdot \frac{1}{2}$)

example: MoM_3 of {2, 6, 5, 12, 11, 10}
is SELECT({5, 11}, 1) = 5

② Partition around MoM_3 and
Let pos_{MoM_3} = position of MoM_3 after partition.

③ If $i > pos_{MoM_3}$,
call SELECT("> MoM_3" list, $i - pos_{MoM_3}$)

If $i < pos_{MoM_3}$,
call SELECT("< MoM_3" list, i)

If $i == pos_{MoM_3}$,
return MoM_3.

What is the RT of ① and ②?

① Compute Mom_3

- Time to find medians of each sublist = 3 comparisons $\frac{n}{3}$ times $= O(n)$

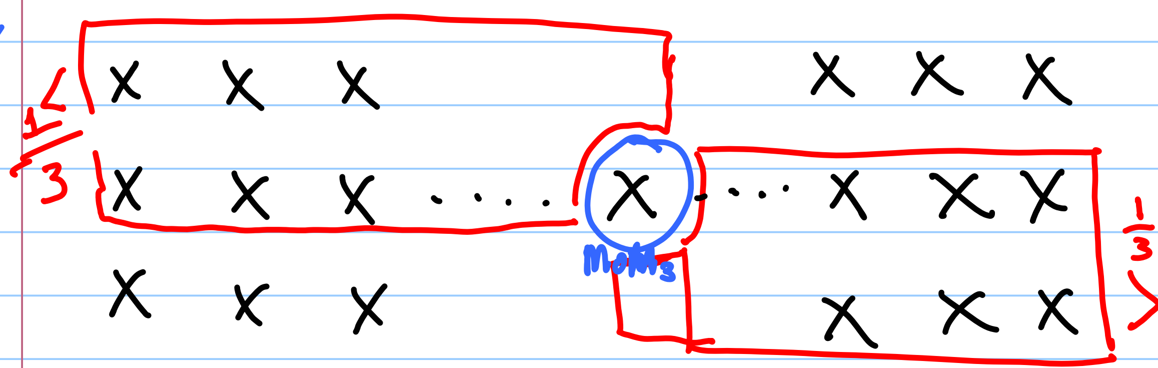
- Time to compute Mom_3 from list of medians of size $\frac{n}{3}$ done recursively: $T(\frac{n}{3})$

② Time to partition: $O(n)$

But how hard is it to compute step ③ — the main recursive call?

Imagine you could line up "towers of 3", sorted by medians. (Not actually sorted, just described conceptually as sorted!!)

What is the run time of (3)?



sorted towers of 3.

What do we know?

Dead center is the median of medians.

Which x's are definitely less than MoM_3?

Which are definitely greater?

What about others?

$$T\left(\frac{2n}{3}\right)$$

$T(n) =$ time to compute MoM_3 + time to partition + $T\left(\frac{2n}{3}\right)$

time to compute MoM_3 = time to find medians of each sublist + time to compute MoM_3 from list of medians

time to find medians of each sublist → $O(n)$

time to compute MoM_3 from list of medians: list size: $\frac{n}{3}$, done recursively, $T\left(\frac{n}{3}\right)$

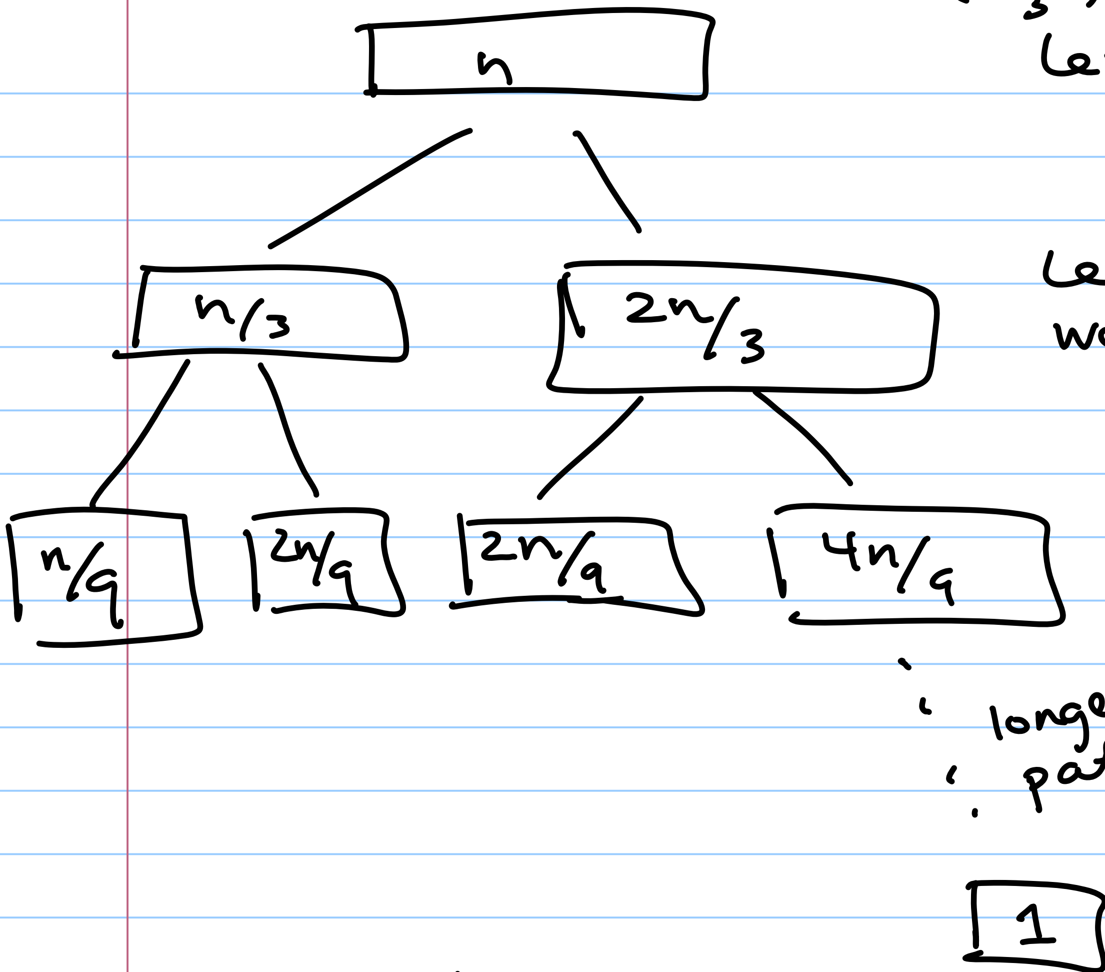
time to partition → $O(n)$

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

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

Let's solve this...

$$T(n) = T(n/3) + T\left(\frac{2n}{3}\right) + an$$



Level ∅

Level 1
work $= \frac{an}{3} + \frac{2an}{3}$

Level 2
work $= an$

Level m.
$m = \log_{3/2} n$

$$T(n) \le \sum_{i=0}^{\log_{3/2} n} an$$

$$= an(\log_{3/2} n + 1)$$

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

Exam

Do not make any change to the exam itself.

Write regrade request on separate sheet of paper explaining why something you lost points for was correct.

Partial credit is not a negotiating point.

We reserve the right to review other questions to look for grading deductions that were missed.

Show of hands:

Friday 3PM Review to go over exam?

Median = 81