Lower bound for selection (median)

<u>Selection problem</u>

Input $S = \{x_1, x_2, \ldots, x_n\}$ a set of n distinct numbers
and $k \in \{1, 2, \ldots, n\}$

Output: the k'th element that is $y \in S$ s.t
$$|\{i \in [n] \mid x_i \le y\}| = k$$

Easy if we sort first:

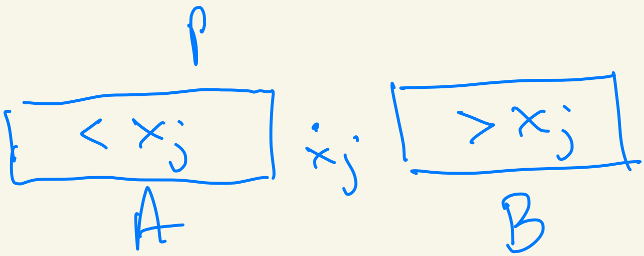
$$S \xrightarrow{\text{sort}} x_{i_1} < x_{i_2} < \cdots < x_{i_k} < \cdots < x_{i_n}$$

answer : y_k

We want to do it in linear time

Revisit idea : Quicksort :

- pick a pivot x_j
- $\underset{A}{\boxed{< x_j}} \quad x_j \quad \underset{B}{\boxed{> x_j}}$



- If $|A| \geq k$ then look for k'th element in A
- if $|A| = k-1$ then x_j is the k'th element
- if $|A| < k-1$ then we look for the k'th element in B when $k' = k - |A| - 1$

If we could always pick the pivot s.t $\min\{|A|, |B|\} \geq cn'$ when n' is the current # numbers in the set we work on Then we obtain a linear algorithm $T(n) \leq T\left(\frac{c-1}{c} n\right) + \Theta(n)$

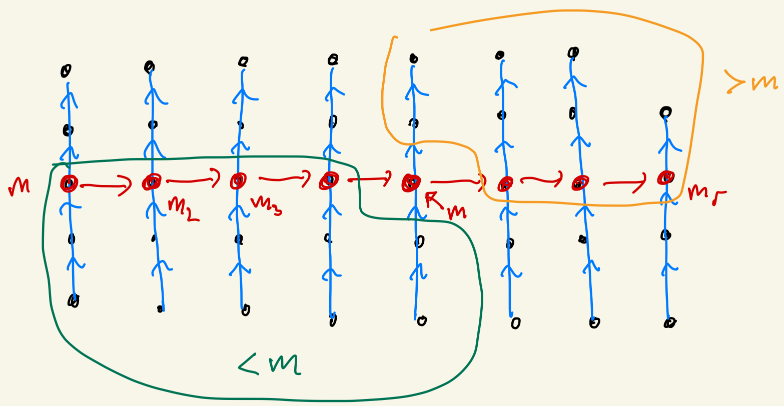
Definition when $|S| = 2k+1$ we call the $k+1$'st element the median of S

when $|S| = 2k$ we call the k'th element the lower median and the $(k+1)$'st element the upper median of S

A deterministic linear time algorithm for selection

Idea: use medians of smaller sets to achieve a good pivot to partition around

1. Partition S into $r = \lceil \frac{n}{5} \rceil$ sets $S_1, S_2 \ldots S_r$ where $|S_i| = 5$ for $i < r$
2. Sort each S_i
3. Identify the (upper) median m_i of S_i $i = 1, 2, \ldots, r$
4. Let $M = \{m_1, m_2, \ldots, m_r\}$
5. Find (upper) median m of M
6. use m as pivot when partitioning S
   let $S_< = \{s \in S \mid s < m\}$, $S_> = \{s \in S \mid s > m\}$
7. If $|S_<| = k-1$ return m
   if $|S_<| > k-1$
      $S := S_<$
      Goto 1.
   if $|S_<| < k-1$
      $k = k - |S_<| - 1$
      $S := S_>$
      Goto 1



The orange set contains at least

$3\left(\left\lceil \frac{1}{2}\left\lfloor \frac{n}{5} \right\rfloor \right\rceil - 2\right) \geq \frac{3n}{10} - 6$ elements of S

[We don't count 3 elements for the last set and the set containing the median of medians]

Similarly, the green set contains at least $\frac{3n}{10} - 6$ elements of S

Hence with $T(n) = $ # comparisons made by the algorithm

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq 140 \\ T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10} + 6\right) + O(n) & n > 140 \end{cases}$$

$\Rightarrow$ $T(n) \leq cn$ for c suitably large

(see Cormen page 222)

Lower bound for finding median

assume $n = 2k+1$ so median m is the $(k+1)$'st element

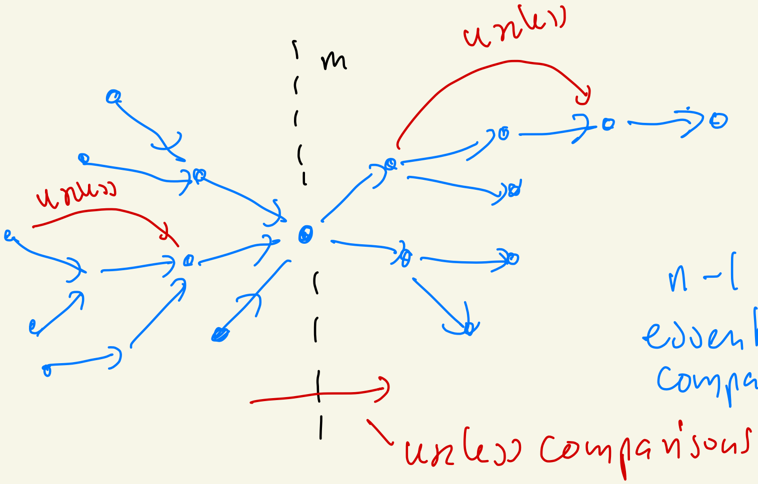
<u>Observation</u>: In order for any algorithm $\mathcal{A}$ to correctly determine (find) m it must find
k elements which are all smaller than m and
k elements which are all larger than m

In terms of acyclic orientations:

if x is smaller than m then $x \to \to \to \cdots \to m$

if y is larger than m then $m \to \to \to \cdots \to y$

So $\mathcal{A}$ must have made a set of comparisons corresponding to the following structure



$n = 17$

$n-1$ essential comparisons

Adversary strategy

Goal: force A to make many useless comparisons

Maintain the following sets U, L, S and $\{m\}$

where U contains elements never in a comparison yet

L contains elements that adversary will make larger than the median

S contains element that adversary will make smaller than the median

m will become the median

Initially S, L are empty and $U = V$ (all elements)

upon query compare(x, y):

1. $x, y \in U \Rightarrow$ reply $x < y$
   $U := U - \{x, y\}$, $S := S + x$, $L := L + y$

2. $x \in U, y \in S \Rightarrow$ reply $x > y$
   $U := U - x$, $L := L + x$

3. $x \in U, y \in L \Rightarrow$ reply $x < y$
   $U := U - x$, $S := S + x$

4. $x \in S$, $y \in L$: reply $x < y$

5. $x, y \in S$ or $x, y \in L$: if $x \rightsquigarrow y$ reply $x < y$
else reply $y < x$

When $|U| = 1$ adversary lets m be the last element in U. Now m is fixed

<u>Claim</u> adversary can for at least $\frac{n-1}{2}$ useless comparisons

P: all comparisons of type 1.–4. are 'useless' (nonessential)

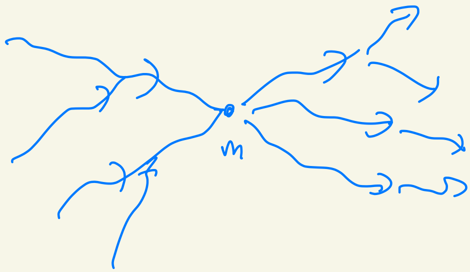
The best case for A (worst for adversary) is when A makes comparisons of type 1 as it reduces $|U|$ by 2.

There are at least $\frac{n-1}{2} = k$ comparisons involving elements.

Thus A must make at least

$$n-1 + \frac{n-1}{2} = \frac{3}{2}n - \frac{3}{2} \text{ comparisons}$$

at the end the essential edges form a tree structure like this



all other comparisons made are answered consistently with blue and red arcs

The adversary finds an acyclic ordering of the acyclic digraph D consisting of red and blue arcs and constructs a bad input consisting of some permutation of $\{1, 2, \ldots n\}$ which forces the deterministic algorithm $\mathcal{A}$ to make at least

$$\frac{3n}{2} - \frac{3}{2}$$ comparisons before it can output the median m.