Selection
General problem: Given a list $L$ of $n$ keys, and $k \in \{1, \ldots, n\}$, find $k$th smallest element in $L$

Special case: $k = \lfloor n/2 \rfloor$ ... the median

One solution: sort the keys into increasing order, return $k$th entry in the sorted list

This takes time $O(n \log n)$

We can do better: linear time!

- a randomized algorithm with expected running time $O(n)$
- a deterministic algorithm with running time $O(n)$
Quick Select: a randomized selection algorithm

\[QSelect(L, k) :\]

choose \( p \) from \( L \) at random

partition \( L \) into 3 sublists: \( L_{<p}, L_{=p}, L_{>p} \)

if \( k \leq \text{length}(L_{<p}) \) then
  return \( QSelect(L_{<p}, k) \)

else if \( k \leq \text{length}(L_{<p}) + \text{length}(L_{=p}) \) then
  return \( p \)

else // \( k > \text{length}(L_{<p}) + \text{length}(L_{=p}) \)
  return \( QSelect(L_{>p}, k - \text{length}(L_{<p}) - \text{length}(L_{=p})) \)
Intuition:

we split the problem into two problems of “roughly equal” size (in linear time) and then solve one of them

reminds us of the recurrence $T(n) \leq T(n/2) + O(n)$

Master Theorem says: $T(n) = O(n)$

BUT . . . this is not rigorous: the splitting step is probabilistic, and we may get some “bad” splits
Let $W := \text{number of comparisons}$

**Theorem.** $E[W] = O(n)$

For $j = 0, 1, 2, \ldots$ let $N_j := \text{size of the subproblem at level } j$
(or zero if none)

**Claim.** $E[N_j] \leq (3/4)^j n$ and for each $j = 0, 1, 2, \ldots$

**Using the claim:**

$$W \leq N_0 + N_1 + \cdots$$

$$E[W] \leq E[N_0] + E[N_1] + \cdots$$

$$\leq n \sum_{j \geq 0} (3/4)^j$$

$$= (1/(1 - 3/4))n = 4n$$
Proof of Claim.

$N_0 = n$

Let’s first prove that $E[N_1] \leq (3/4)n$

Imagine the keys in $L$ are sorted
Let $R$ the index of the pivot $p$ in the sorted list
$R$ is uniformly distributed over $\{1, \ldots, n\}$
$\text{length}(L_{<p}) \leq R - 1$ and $\text{length}(L_{>p}) \leq n - R$
$\therefore N_1 \leq \max\{R - 1, n - R\}$
A calculation . . .

Assume $R$ uniform over $\{1, \ldots, n\}$

Want to show: $E[\max\{R - 1, n - R\}] \leq (3/4)n$

*NOTE:* $E[\max\{X, Y\}] \not\leq \max\{E[X], E[Y]\}$

Proof by picture ($n = 8$):

expectation $\leq 1/n$ times shaded area
To recap: we have proved $E[N_1] \leq (3/4)n$

What about $N_2$? **Use conditional expectation:**

$$E[N_2] = \sum_m E[N_2 | N_1 = m] \Pr[N_1 = m]$$

same analysis as $N_1$

$$\leq \sum_m \left( \frac{3}{4}m \right) \Pr[N_1 = m]$$

$$= \left( \frac{3}{4} \right) E[N_1] \leq \left( \frac{3}{4} \right)^2 n$$

By induction: $E[N_j] \leq (3/4)^j n$ for $j = 0, 1, 2, \ldots$
Analysis of recursion depth

Let $D :=$ the depth of the recursion tree

**Theorem.** $E[D] = O(\log n)$

**Tail sum formula:** $E[D] = \sum_{j \geq 1} \Pr[D \geq j]$

**Observe:** $D \geq j \iff N_j \geq 1$

**Markov says:** $\Pr[N_j \geq 1] \leq E[N_j] \leq (3/4)^j n$
Set $j_0 := \left\lceil \log_{4/3} n \right\rceil$  // = least $j$ such that $(3/4)^jn \leq 1$

We have:

$$E[D] = \sum_{j \geq 1} \Pr[D \geq j]$$

$$= \sum_{j=1}^{j_0-1} \Pr[D \geq j] + \sum_{j=j_0}^{\infty} \Pr[D \geq j]$$

$$\leq (j_0 - 1) + \sum_{j=j_0}^{\infty} (3/4)^jn$$

$$= (j_0 - 1) + ((3/4)^{j_0}n) \sum_{j=j_0}^{\infty} (3/4)^{j-j_0}$$

$$\leq \log_{4/3} n + 4$$
Practical aspects: a fast, in-place partitioning algorithm

An idea from Bentley & McIlroy (1993)

Two inner loops:

• moving $b$: scan over $<$, swap $=$, halt on $>$
• moving $c$: scan over $>$, swap $=$, halt on $<$

Swap elements $b$ and $c$, $b++$, $c--$
Repeat until $b$ crosses $c$
When finished, the $=$’s are swapped to the middle
Deterministic linear-time selection

Idea:

- divide $L$ into $\approx n/5$ blocks of size 5
- sort each block, and compute median of each block
- let $M :=$ the list of medians (so $\text{length}(M) \approx n/5$)
- recursively find the median $p$ of $M$
- use $p$ as the pivot, and proceed as in Quick Select
Consider a single recursive invocation

Local cost is $O(n)$

Both $\text{length}(L_{<p})$ and $\text{length}(L_{>p})$ are $\leq (7/10)n + O(1)$

Two recursive calls:

- one of size at most $n/5 + O(1)$
- one of size at most $(7/10)n + O(1)$
Sum of subproblem sizes is $\leq 0.9n + c$, for some constant $c$

Choose $n_0$ such that $0.9n + c \leq 0.91n$ for all $n \geq n_0$

Implementation: halt recursion when $n < n_0$

Let $s_j := $ sum of problem sizes at level $j$, for $j = 0, 1, 2, \ldots$

We have $s_j \leq (0.91)^j n$ for $j = 0, 1, 2, \ldots$

Total cost is $O(w)$, where

$$w := \sum_{j \geq 0} s_j \leq \sum_{j \geq 0} (0.91)^j n \leq \left(\frac{100}{9}\right)n$$