CSE 100: Algorithm Design and Analysis
Chapter 09: Median and Order Statistics

Sungjin Im

University of California, Merced

Last Update: 03-03-2023

It is the time you have wasted for your rose that makes your rose so important.

Antoine de Saint-Exupéry, The Little Prince

Medians and Order Statistics

- ith order statistic is the ith smallest element of a set of n elements.
- The minimum is the 1st order statistic.
- The maximum is the nth order statistic.
- A median is the 'halfway point' of the set.
  - If n is odd, the median is unique, and is $(n+1)/2$ order statistic.
  - If n is even, then there are two medians.
    - The lower median: $i = n/2$.
    - The upper median: $i = n/2 + 1$.

The Selection Problem

Input: A set A of n (distinct) numbers and an integer i, with $1 \leq i \leq n$.
Output: The ith smallest element of A.

The Selection Problem

Input: A set A of n (distinct) numbers and an integer i, with $1 \leq i \leq n$.
Output: The ith smallest element of A.

- If we use sorting, we can solve this problem in $O(n \log n)$ time.
- We will learn $O(n)$ time algorithms.
  - practical. A randomized algorithm with $O(n)$ expected running time
  - A deterministic algorithm with $O(n)$ running time.

$\Rightarrow$

im "

The Selection Problem

Warm-up: Minimum

```
MINIMUM(A)
min = A[1]
for i = 2 to A.length
    if min > A[i]
        min = A[i]
return min
```

Time:

The Selection Problem

Warm-up: Minimum

```
MINIMUM(A)
min = A[1]
for i = 2 to A.length
    if min > A[i]
        min = A[i]
return min
```

Time: $\Theta(n)$.

The Selection Problem

Warm-up: Maximum

Time: $\Theta(n)$.

Selection in linear time in expectation

Based on (randomized) partitioning used in Quicksort.

R-Select(A, p, r, i): Return ith smallest element in $A[p...r]$.

Ex. $A[1...9] = \langle 1, 8, 4, 9, 7, 2, 3, 6, 5 \rangle$.

R-Quicksort$(A, 1, 9)$ $>$ R-Select$(A, 1, 9, 2)$

1 8 4 9 7 2 3 6 5

1 4 2 3 5 8 9 7 6

#=4 < #=4

R-Select(1, 4, 2)

1 2 3 4

R-Select(1, 2, 2).

1 2

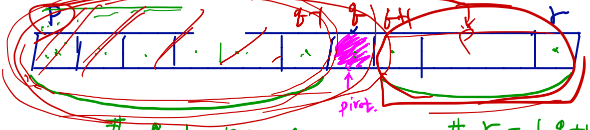
Selection in linear time in expectation

Based on (randomized) partitioning used in Quicksort

```
RANDOMIZED-PARTITION(A, p, r)
i = RANDOM(p, r)
exchange A[r] with A[i]
return PARTITION(A, p, r)
```

Selection in linear time in expectation

1 2 3 4 . . . . . . k . . . . . . r-p+1



\# q-1-p+1 = q-p   \# r-(q+1)+1 = r-q.

i < k.

k = q-p+1
pivot is k th smallest elmt in A[p...r].

RANDOMIZED-SELECT(A, p, r, i) in A[p ... r]   ith smallest elmt.

RS(A, 1, n, i).

```
O(1)   if p == r
           return A[p]
O(n)   q = RANDOMIZED-PARTITION(A, p, r)
       k = q − p + 1
       if i == k          // the pivot value is the answer
           return A[q]
       elseif i < k
           return RANDOMIZED-SELECT(A, p, q − 1, i)
       else return RANDOMIZED-SELECT(A, q + 1, r, i − k)
```

⟹ ith smallest elmt in A[p ... r]

= (i-k)th smallest in A[q+1 ... r]

Selection in linear time in expectation

$$T(n) = T(\max\{\text{left subarray size}, \text{right subarray size}\}) + \Theta(n)$$

Selection in linear time in expectation

$$T(n) = T(\max\{\text{left subarray size}, \text{right subarray size}\}) + \Theta(n)$$

- Best Partitioning: $\underline{T(n) = T(n/2) + \Theta(n)}$

Selection in linear time in expectation

$$T(n) = T(\max\{\text{left subarray size}, \text{right subarray size}\}) + \Theta(n)$$

- Best Partitioning: $T(n) = T(n/2) + \Theta(n) \to T(n) = \Theta(n)$.
- Worst Partitioning: $T(n) = T(n-1) + \Theta(n)$

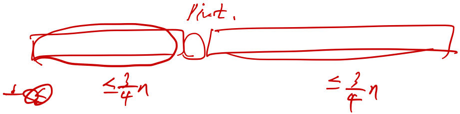
Selection in linear time in expectation

$T(n) = T(\max\{\text{left subarray size}, \text{right subarray size}\}) + \Theta(n)$

- Best Partitioning: $T(n) = T(n/2) + \Theta(n) \rightarrow T(n) = \Theta(n)$.
- Worst Partitioning: $T(n) = T(n-1) + \Theta(n)$
  $\rightarrow T(n) = \Theta(n^2)$.
- 'Balanced' Partitioning: $T(n) = T(\frac{3}{4}n) + \Theta(n)$.

Let's say that the partitioning is 'balanced' if

$$\frac{1}{4}n \leq \text{ pivot's order } \leq \frac{3}{4}n$$



Selection in linear time in expectation

- 'Balanced' Partitioning: $T(n) = T(\frac{3}{4}n) + \Theta(n)$
  $\rightarrow T(n) = \Theta(n)$.

Let's say that the partitioning is 'balanced' if

$$\frac{1}{4}n \leq \text{ pivot's order } \leq \frac{3}{4}n$$

50%.

We can show that RT of Randomized Selection is $\Theta(n)$ in expectation (for any input).

* We can even show that RT $= \Theta(n)$ with probability that tends to 1 as n grows. (Beyond the scope of this course)

Selection in linear time in expectation

Formal proof

$$E(n) \le \frac{1}{2}E(\tfrac{3}{4}n) + \frac{1}{2}E(n-1) + O(n)$$

$$\le \frac{1}{2}E(\tfrac{3}{4}n) + \frac{1}{2}E(n) + O(n)$$

- Let $E(n) := \max_{I:\text{input of size} n} E[RT(I)]$.
- $E(n)$ is non-decreasing in n.
- The partitioning is 'balanced' with probability $1/2$.
- If 'balanced', the bigger subproblem size $\le \frac{3}{4}n$; else $\le n-1$.
- $E(n) \le \frac{1}{2}E(\frac{3}{4}n) + \frac{1}{2}E(n-1) + O(n) \le \frac{1}{2}E(\frac{3}{4}n) + \frac{1}{2}E(n) + O(n)$.
- $E(n) \le E(\frac{3}{4}n) + O(n) \to E(n) = O(n)$.

$\Rightarrow \frac{1}{2}E(n) \le \frac{1}{2}E(\frac{3}{4}n) + O(n)$

$\Rightarrow E(n) \le E(\frac{3}{4}n) + O(n)$

* Let's say that the partitioning is 'balanced' if $\frac{1}{4}n \le$ pivot's order $\le \frac{3}{4}n$

Selection in linear time

(Deterministic) Select

- If we can find a good pivot leading to a balanced partition in linear time, we have a linear time deterministic algorithm.
- But finding a good pivot is a kind of select problem.

Selection in linear time

(Deterministic) Select

- If we can find a good pivot leading to a balanced partition in linear time, we have a linear time deterministic algorithm.
- But finding a good pivot is a kind of select problem. So, recursion will be used to find such a pivot.

Selection in linear time

(Deterministic) Select

- If we can find a good pivot leading to a balanced partition in linear time, we have a linear time deterministic algorithm.
- But finding a good pivot is a kind of select problem. So, recursion will be used to find such a pivot.

Key idea: Find a good pivot from a smaller problem.

Selection in linear time

(Deterministic) Select

1. Divide the n elements into groups of size 5. So $n/5$ groups.
2. Find the median of each group.
3. Find the median x of the $n/5$ medians by a recursive call to Select.
4. Call Partition with x as the pivot.
5. Make a recursive call to Select either on the smaller elements or larger elements (depending on the situation); if the pivot is the answer we are done.

Key question: Is x a good pivot?

Selection in linear time

(Deterministic) Select

$T(n)$ Claim: $\frac{3}{10}n \le x$'s order $\le \frac{7}{10}n$.

" NOTE: for simplicity, small additive constants are ignored.

$O(n)$ 1. Divide the n elements into groups of size 5. So $n/5$ groups.

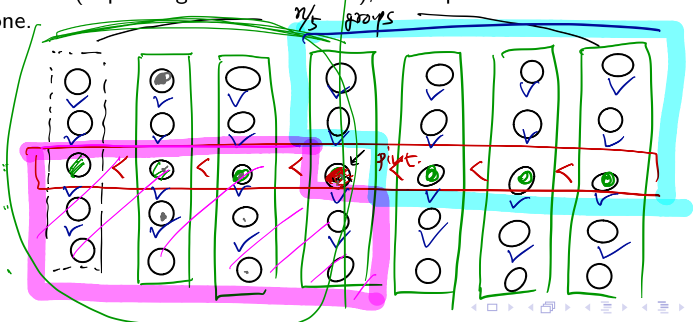
$+\,O(n)$ 2. Find the median of each group.

$+\,T(\frac{n}{5})$ 3. Find the median x of the $n/5$ medians by a recursive call to Select.

$+\,O(n).$ 4. Call Partition with x as the pivot.

$+\,T(\frac{7}{10}n)$ 5. Make a recursive call to Select either on the smaller elements or larger elements (depending on the situation); if the pivot is the answer we are done.

$\frac{1}{2} * \frac{3}{5} = \frac{3}{10}$



bigger than pivot.

smaller than pivot.

Selection in linear time

(Deterministic) Select: RT analysis

Claim: $\frac{3}{10}n \leq x$'s order $\leq \frac{7}{10}n$.

1. Divide the n elements into groups of size 5. So $n/5$ groups.

Selection in linear time

(Deterministic) Select: RT analysis

Claim: $\frac{3}{10}n \leq x$'s order $\leq \frac{7}{10}n$.

1. Divide the n elements into groups of size 5. So $n/5$ groups. $\Theta(n)$.
2. Find the median of each group.

Selection in linear time

(Deterministic) Select: RT analysis

Claim: $\frac{3}{10}n \leq x$'s order $\leq \frac{7}{10}n$.

1. Divide the n elements into groups of size 5. So $n/5$ groups. $\Theta(n)$.
2. Find the median of each group. $\Theta(n)$.
3. Find the median x of the $n/5$ medians by a recursive call to Select.

Selection in linear time

(Deterministic) Select: RT analysis

Claim: $\frac{3}{10}n \leq x$'s order $\leq \frac{7}{10}n$.

1. Divide the n elements into groups of size 5. So $n/5$ groups. $\Theta(n)$.
2. Find the median of each group. $\Theta(n)$.
3. Find the median x of the $n/5$ medians by a recursive call to Select. $T(n/5)$.
4. Call Partition with x as the pivot.

Selection in linear time

(Deterministic) Select: RT analysis

Claim: $\frac{3}{10}n \leq x$'s order $\leq \frac{7}{10}n$.

1. Divide the n elements into groups of size 5. So $n/5$ groups. $\Theta(n)$.
2. Find the median of each group. $\Theta(n)$.
3. Find the median x of the $n/5$ medians by a recursive call to Select. $T(n/5)$.
4. Call Partition with x as the pivot. $\Theta(n)$.
5. Make a recursive call to Select either on the smaller elements or larger elements (depending on the situation); if the pivot is the answer we are done.

Selection in linear time

(Deterministic) Select: RT analysis

Claim: $\frac{3}{10}n \leq x$'s order $\leq \frac{7}{10}n$.

1. Divide the n elements into groups of size 5. So $n/5$ groups. $\Theta(n)$.
2. Find the median of each group. $\Theta(n)$.
3. Find the median x of the $n/5$ medians by a recursive call to Select. $T(n/5)$.
4. Call Partition with x as the pivot. $\Theta(n)$.
5. Make a recursive call to Select either on the smaller elements or larger elements (depending on the situation); if the pivot is the answer we are done. $T(\frac{7}{10}n)$.

Selection in linear time

(Deterministic) Select: RT analysis

$$T(n) = T(n/5) + T(7n/10) + \Theta(n)$$

Selection in linear time

(Deterministic) Select: RT analysis

$$T(n) = T(n/5) + T(7n/10) + \Theta(n)$$
$$\Rightarrow T(n) = \Theta(n).$$

Selection in linear time

(Deterministic) Select: RT analysis

$$T(n) = T(n/5) + T(7n/10) + \Theta(n)$$
$$\Rightarrow T(n) = \Theta(n).$$