# Linear Time Selection and Applications

Ira A. Fulton Schools of Engineering
Arizona State University

# Selection: Part 1

- Finding the Smallest
- Finding the i-th Smallest: The problem
- A naïve approach
- A simple by good approach
- Linear time algorithm: High level illustration
- Linear time algorithm: Algorithm in details
- Analysis and Applications

# Finding the smallest

```
Smallest(A, n){
      xv = A[1]; xi = 1;
      for (i=2; i<=n; i++){
        if (A[i] < xv) then{
          xv = A[i]; xi = i;
        }
      }
      return xi;
    }
```

Time complexity: $O(n)$

# Finding the $i$-th smallest…

**Problem:** Given an unsorted array A of $n$ elements, and an integer $i$, $1 \leq i \leq n$, find the $i$-th smallest element in A.

# Finding the $i$-th smallest…

**Problem:** Given an unsorted array A of $n$ elements, and an integer $i$, $1 \leq i \leq n$, find the $i$-th smallest element in A.

- **1st approach**: Find (and delete) the smallest $i$ times: $O(in)$ time

# Illustration of approach 1: Finding the 4$^{th}$ smallest

| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 12 | 14 | 15 | 11 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

# Illustration of approach 1: Finding the 4$^{th}$ smallest

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 12 | 14 | 15 | 11 | 16 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

The 1$^{st}$ smallest is A[11]=11

# Illustration of approach 1: Finding the 4th smallest

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 12 | 14 | 15 | 16 | 11 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- The 1st smallest is A[11]=11
- Swap with A[12], and ignore it.

# Illustration of approach 1: Finding the 4th smallest

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 12 | 14 | 15 | 16 | 11 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

The 2nd smallest is A[8]=12

# Illustration of approach 1: Finding the 4th smallest

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 16 | 14 | 15 | 12 | 11 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- The 2nd smallest is A[8]=12
- Swap with A[11], and ignore it.

# Illustration of approach 1: Finding the 4$^{th}$ smallest

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 16 | 14 | 15 | 12 | 11 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- The 3$^{rd}$ smallest is A[7]=13

# Illustration of approach 1: Finding the 4th smallest

| 22 | 21 | 20 | 18 | 17 | 19 | 15 | 16 | 14 | 13 | 12 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- The 3rd smallest is A[7]=13
- Swap with A[10], and ignore it.

# Illustration of approach 1: Finding the 4th smallest

| 22 | 21 | 20 | 18 | 17 | 19 | 15 | 16 | 14 | 13 | 12 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

**The 4th smallest is A[9]=14. This is the answer.**

# Finding the $i$-th smallest…

**Problem: Given an unsorted array A of $n$ elements, and an integer $\boldsymbol{i}$, $1 \leq \boldsymbol{i} \leq \boldsymbol{n}$, find the $\boldsymbol{i}$-th smallest element in A.**

- **1st approach: Find (and delete) the smallest $\boldsymbol{i}$ times: $O(\boldsymbol{i}n)$ time**
- **2nd approach: Sort A, then find the $i$-th element: $O(n \log n)$ time**

# Illustration of approach 2: Finding the 4th smallest

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 12 | 14 | 15 | 11 | 16 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

# Illustration of approach 2: Finding the 4th smallest

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

Sort the given array.

# Illustration of approach 2: Finding the 4th smallest

| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- **Sort the given array.**
- **Return the 4th element A[4] (after sorting)**

Ira A. Fulton Schools of Engineering

Arizona State University

# Selection: Part 2

- Finding the Smallest
- Finding the i-th Smallest: The problem
- A naïve approach
- A simple by good approach
- Linear time algorithm: High level illustration
- Linear time algorithm: Algorithm in details
- Analysis and Applications

# Finding the $i$-th smallest…

**Problem:** Given an unsorted array A of n elements, and an integer $i \leq n$, find the $i$-th smallest element in A.

- **Best Algorithm**: int select(A, p, r, $i$): $O(n)$ time
  - A is the array
  - p and r are the start/end indices of the portion of A
  - select(A, p, r, $i$) finds the $i$-th smallest element in A[p, r]

# High-level Illustration: Case 1

| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 12 | 14 | 15 | 11 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- Given unsorted array A of 12 elements (p=1,r=12).
- Want to find the 6th smallest element ($i$=6).
- Take an element (A[12]=16) to perform partition.

# High-level Illustration: Case 1

| 14 | 13 | 12 | 11 | 15 | 16 | 22 | 18 | 17 | 19 | 20 | 21 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- Pivot has index $q = 6$, $\mathrm{k} = \mathrm{q} - \mathrm{p} + 1 = 6$.
- Since $k = i$, A[$i$] (after partition) is the $i$-th smallest element in A.
- Solution found.

# High-level Illustration: Case 2

| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 12 | 14 | 15 | 11 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- **Given unsorted array A of 12 elements (p=1, r=12).**
- **Want to find the $4^{th}$ smallest element (*i*=4).**
- **Take an element (16) to perform partition**

# High-level Illustration: Case 2

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | 13 | 12 | 11 | 15 | 16 | 22 | 18 | 17 | 19 | 20 | 21 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- Pivot has index $q = 6, \mathrm{k} = \mathrm{q} - \mathrm{p} + 1 = 6$.
- Since $i < k$, we call select(A, $p$, $q - 1$, $i$) to find the $i^{\text{th}}$ smallest in A[$p$, $q$−$1$], which is the $i^{\text{th}}$ smallest element in A[p, r].
- Recursive call to a smaller instance of the same problem.

# High-level Illustration: Case 3

| 22 | 21 | 20 | 18 | 17 | 19 | 13 | 12 | 14 | 15 | 11 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- Given unsorted array A of 12 elements (p=1, r=12).
- Want to find the 8th smallest element ($i$=8).
- Take an element (16) to perform partition

# High-level Illustration: Case 3

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | 13 | 12 | 11 | 15 | 16 | 22 | 18 | 17 | 19 | 20 | 21 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

- **Pivot has index $q = 6, k = q - p + 1 = 6$.**

- **Since $i > k$, we call select(A, $q + 1, r, i - k$) to find the ($i - k$)th smallest element in A[$q + 1$, 12], which is the $i$th smallest element in A[p, r].**

- **Recursive call to a smaller instance of the same problem.**

Ira A. Fulton Schools of Engineering

Arizona State University

# Selection: Part 3

- Finding the Smallest
- Finding the i-th Smallest: The problem
- A naïve approach
- A simple by good approach
- Linear time algorithm: High level illustration
- Linear time algorithm: Algorithm in details
- Analysis and Applications

# The Main Question is:

- How to find a good pivot element???

# The Key Idea...Median of Medians

Blum, M.; Floyd, R. W.; Pratt, V. R.; Rivest, R. L.; Tarjan, R. E.,
“Time bounds for selection”;
Journal of Computer and System Sciences;
Vol. 7 (1973): 448–461.

- Manuel Blum, Turing Award 1995.
- Robert Floyd, Turing Award 1978.
- Vaughan Pratt, Known for the KMP algorithm.
- Ron Rivest, Turing Award 2002.
- Robert Tarjan, Turing Award 1986.

SELECT (A, p, r, $i$) for finding the $i^{th}$ smallest element in A[p..r], which is the (i-p+1)$^{th}$ smallest element of an input array A[1..N]. Denote n=p-r+1.

Step 1: Divide the n elements of A[p..r] into $\lfloor n/5 \rfloor$ groups of <u>5</u> elements each and at most one group made up of the remaining (n mod 5) elements.

Step 2: Find the median of each of the $\lceil n/5 \rceil$ groups by insertion sorting the (at most 5) elements of each group and taking its middle element (if the group has an even # of elements, take the lower of the two medians).

Step 3: Use SELECT recursively to find the median x of the $\lceil n/5 \rceil$ medians found in Step2;

Step 4: Partition A[p..r] around the median-of-the-medians x using a modified version of quicksort's PARTITION that takes x as a parameter and uses x as the pivot element. Let q be the index of the median x, and k=q-(p-1).

Step 5: If $i$==k, return x. O/w, use SELECT(A, p, q-1, $i$) to find the $i^{th}$ smallest element on the <u>low side</u>, if $i$<k, or use SELECT(A, q+1, r, $i$-k) to find the ($i$-k)$^{th}$ smallest element on the <u>high side</u>, if $i$>k.

# Illustration of the Algorithm

## We illustrate the select algorithm with an example.

- A contains 37 integers (column major on next page)
- $i$ is set to 8

## Call select(A, 1, 37, 8)

- p=1, r=37, $i$=8

# Illustration of the Algorithm: select(A, 1, 37, 8)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 30 | 28 | 26 | 24 | 22 | 20 | 18 | 16 |
| 73 | 79 | 4 | 6 | 8 | 10 | 12 | 14 |
| 71 | 67 | 61 | 59 | 53 | 47 | 43 | |
| 41 | 37 | 31 | 29 | 23 | 19 | 17 | |
| 2 | 3 | 5 | 7 | 9 | 11 | 13 | |

# Groups of 5 (possible exception of the last group)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 30 | 28 | 26 | 24 | 22 | 20 | 18 | 16 |
| 73 | 79 | 4 | 6 | 8 | 10 | 12 | 14 |
| 71 | 67 | 61 | 59 | 53 | 47 | 43 | |
| 41 | 37 | 31 | 29 | 23 | 19 | 17 | |
| 2 | 3 | 5 | 7 | 9 | 11 | 13 | |

# Find the Median in Each Group

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 30 | 28 | 26 | 24 | 22 | 20 | 18 | 16 |
| 73 | 79 | 4 | 6 | 8 | 10 | 12 | 14 |
| 71 | 67 | 61 | 59 | 53 | 47 | 43 | |
| 41 | 37 | 31 | 29 | 23 | 19 | 17 | |
| 2 | 3 | 5 | 7 | 9 | 11 | 13 | |

# Form New Array B of Medians: B[1..8]

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 41 | 37 | 26 | 24 | 22 | 19 | 17 | 14 |

# Find the Median of Medians: select(B, 1, 8, 4)

Call select(B, 1, 8, 4)

| | |
|---|---|
| 41 | 19 |
| 37 | 17 |
| 26 | 14 |
| 24 | |
| 22 | |

# Find the Median of Medians: select(B, 1, 8, 4)

| | |
|---|---|
| 41 | 19 |
| 37 | <u>17</u> |
| <u>26</u> | 14 |
| 24 | |
| 22 | |

# Form New Array BB of Medians: BB[1..2]

| | |
|---|---|
| 26 | 17 |

# Find the Median of Medians: select(BB, 1, 2, 1)

Call select(BB, 1, 2, 1)
Pivot position: q=1 (for array BB)
Return 17

26

<u>17</u>

# Return to select(B, 1, 8, 4), with pivot=17 found in select(BB, 1, 2, 1)

Call select(BB, 1, 2, 1)
Pivot position: q=1 (for array BB)
Return 17

17

26

# Use MM (17) to Partition Array B[1..8]: within select(B, 1, 8, 4)

Pivot position: q=2 (in array B)
Need to call Select(B, 3, 8, 2) to find the 2nd smallest element in B[3..8], which is th 4th smallest element in B[1..8].

| | |
|---|---|
| 14 | 26 |
| 17 | 37 |
| 19 | 41 |
| 22 | |
| 24 | |

# Need to Find the 2nd Smallest in B[3..8]: select(B, 3, 8, 2)

Call Select(B, 3, 8, 2)

| | |
|---|---|
| 14 | 26 |
| 17 | 37 |
| 19 | 41 |
| 22 | |
| 24 | |

# Need to Find the 2nd Smallest in B[3..8]: select(B, 3, 8, 2)

19 41

22

24

26

37

# Need to Find the 2nd Smallest in B[3..8]: select(B, 3, 8, 2)

19 <u>41</u>

22

<u>24</u>

26

37

# Form New Array BBB of Medians: select(BBB, 1, 2, 1)

| | |
|---|---|
| 24 | 41 |

# Find Median of Medians: select(BBB, 1, 2, 1)

Call Select(BBB, 1, 2, 1)
There is only one group and one median 24
MM is 24
Partition BBB[1..2] using 24
Pivot position: q=1 (in array BBB)
Return MM: 24

<ins>**24**</ins>

**41**

# Use the MM (24) to Partition B[3..8]: select(B, 3, 8, 2)

Use MM 24 to partition array B[3..8]
Pivot position: q=5
Need to call Select(B, 3, 4, 2)

19 41

22

24

26

37

# Need to Find the 2nd Smallest in B[3..4]: select(B, 3, 4, 2)

19

22

# Find the Medians: select(B, 3, 4, 2)

There is only one group, one median 19
MM is 19

<u>19</u>

22

# Form the New Array of the Medians (1 median)

19

# Find the Median of Medians

**19**

# Use the MM to Partition the 2 Medians

Use MM (19) to partition B[3..4]
Pivot position: q=3
Need to call Select(B, 4, 4, 1)

19

22

# Find the Smallest Element in B[4..4]: select(B, 4, 4, 1)

The Median for B[1..8] (MM for A[1, 37]) is 22

# Use MM (22) to Partition Original Array (q=19)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 7 | 12 | 18 | 24 | 31 | 53 | 73 |
| 3 | 8 | 13 | 19 | 26 | 37 | 59 | 79 |
| 4 | 9 | 14 | 20 | 28 | 41 | 61 | |
| 5 | 10 | 16 | 22 | 29 | 43 | 67 | |
| 6 | 11 | 17 | 23 | 30 | 47 | 71 | |

# $8 = i$<k=19: Find the 8$^{th}$ Smallest Element in the Blue Array

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 7 | 12 | 18 | 24 | 31 | 53 | 73 |
| 3 | 8 | 13 | 19 | 26 | 37 | 59 | 79 |
| 4 | 9 | 14 | 20 | 28 | 41 | 61 | |
| 5 | 10 | 16 | 22 | 29 | 43 | 67 | |
| 6 | 11 | 17 | 23 | 30 | 47 | 71 | |

# $8 = i < k = 19$: Find the $8^{th}$ Smallest Element in the Blue Array

Call Select(A, 1, 18, 8)
Omitting detailed comments…

| | | | |
|---|---|---|---|
| 2 | 7 | 12 | 18 |
| 3 | 8 | 13 | 19 |
| 4 | 9 | 14 | 20 |
| 5 | 10 | 16 | |
| 6 | 11 | 17 | |

# $p=1$, $r=18$, $i=8$

| | | | |
|---|---|---|---|
| 2 | 7 | 12 | 18 |
| 3 | 8 | 13 | 19 |
| 4 | 9 | 14 | 20 |
| 5 | 10 | 16 | |
| 6 | 11 | 17 | |

# Find the Median of 4 Medians

**4** **9** **14** **19**

# Use the MM to Partition the 18 Medians

| | | | |
|---|---|---|---|
| 2 | 7 | 12 | 18 |
| 3 | 8 | 13 | 19 |
| 4 | <u>9</u> | 14 | 20 |
| 5 | 10 | 16 | |
| 6 | 11 | 17 | |

# $i$==k: the 8th Smallest Element is 9

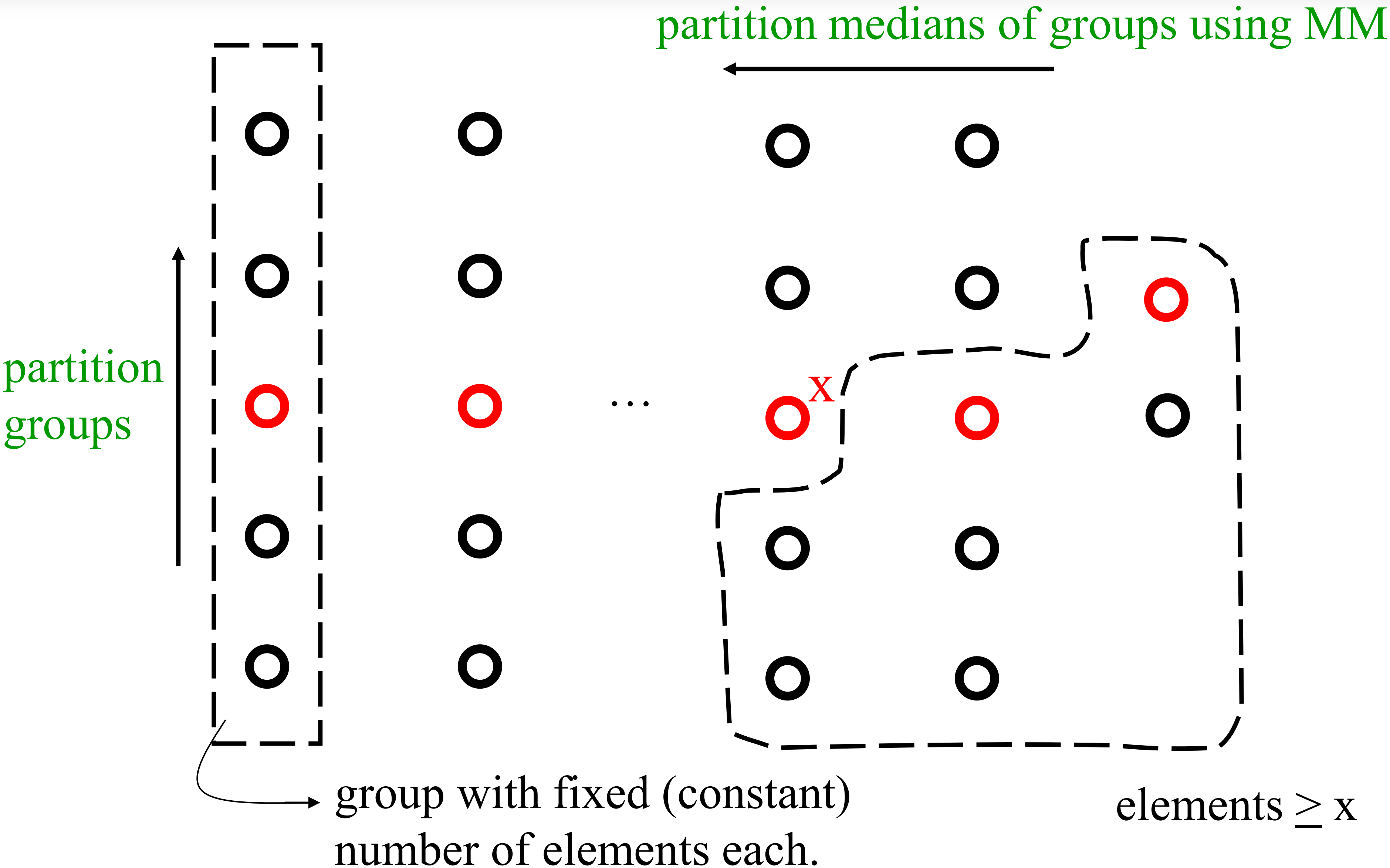
| | | | |
|---|---|---|---|
| 2 | 7 | 12 | 18 |
| 3 | 8 | 13 | 19 |
| 4 | <u>9</u> | 14 | 20 |
| 5 | 10 | 16 | |
| 6 | 11 | 17 | |

ASU®
Ira A. Fulton Schools of
Engineering
Arizona State University

# Selection: Part 4

- Finding the Smallest
- Finding the i-th Smallest: The problem
- A naïve approach
- A simple by good approach
- Linear time algorithm: High level illustration
- Linear time algorithm: Algorithm in details
- Analysis and Applications

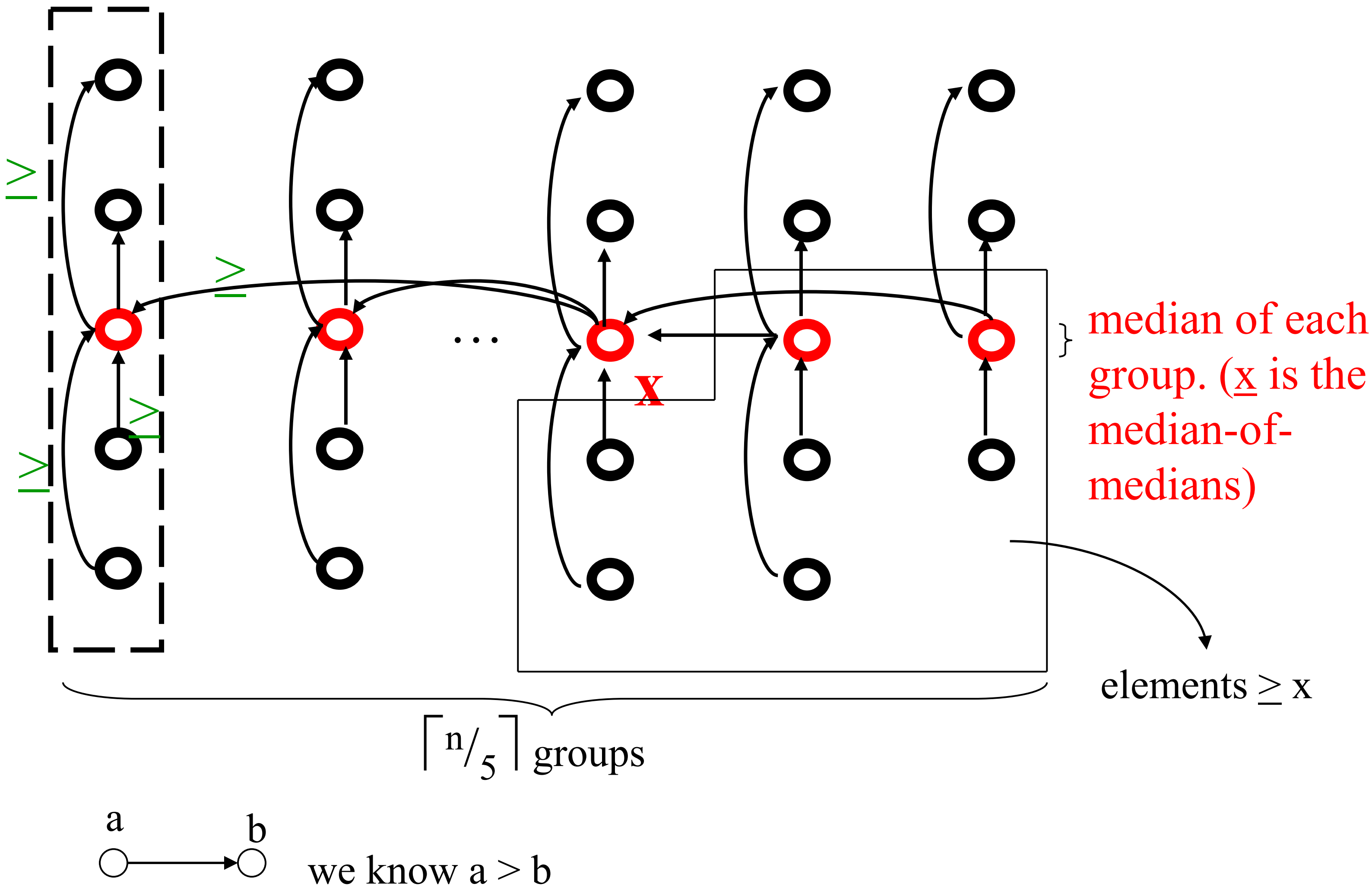
# Analysis (not part of the algorithm)



# Analysis (not part of the algorithm)

**(Step5:)** SELECT(A, p, q-1, *i*), if k > *i*

SELECT(A, q+1, r, *i*-k), if k > *i*.



# Illustration (1)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 30 | 28 | <u>26</u> | <u>24</u> | <u>22</u> | 20 | 18 | 16 |
| 73 | 79 | 4 | 6 | 8 | 10 | 12 | <u>14</u> |
| 71 | 67 | 61 | 59 | 53 | 47 | 43 | |
| <u>41</u> | <u>37</u> | 31 | 29 | 23 | <u>19</u> | <u>17</u> | |
| 2 | 3 | 5 | 7 | 9 | 11 | 13 | |

# Illustration (2)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 30 | 28 | 5 | 7 | 9 | 11 | 13 | |
| 2 | 3 | 4 | 6 | 8 | 10 | 12 | |
| **41** | **37** | **26** | **24** | **22** | **19** | **17** | **14** |
| 73 | 79 | 31 | 29 | 23 | 20 | 18 | 16 |
| 71 | 67 | 61 | 59 | 53 | 47 | 43 | |

# Illustration (3)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 11 | | 13 | 9 | 7 | 28 | 30 | 5 |
| 10 | | 12 | 8 | 6 | 3 | 2 | 4 |
| 19 | 14 | 17 | 22 | 24 | 37 | 41 | 26 |
| 20 | 16 | 18 | 23 | 29 | 79 | 73 | 31 |
| 47 | | 43 | 53 | 59 | 67 | 71 | 61 |

# Illustration (3)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 11 | | 13 | 9 | 7 | 28 | 30 | 5 |
| 10 | | 12 | 8 | 6 | 3 | 2 | 4 |
| 19 | 14 | 17 | 22 | 24 | 37 | 41 | 26 |
| 20 | 16 | 18 | 23 | 29 | 79 | 73 | 31 |
| 47 | | 43 | 53 | 59 | 67 | 71 | 61 |

- Blue portion is **guaranteed** to be smaller than MM
- Green portion is **guaranteed** to be larger than MM

# Analysis of the Algorithm

- For simplicity, assume all n elements in A are distinct.
- At least half of the medians found in Step2 are $\geq x$
- At least half of the $\lceil n/5 \rceil$ groups (except 2 groups, i.e., the group that has fewer than 5 elements and the group that contains x itself) contribute at least 3 elements that are > x.
- Therefore, the number of elements that are greater than x is at least

$$3\left(\frac{\lceil \frac{n}{5} \rceil}{2} - 2\right) \geq \frac{3n}{10} - 6$$

- Similarly, the number of elements that are < x is at least $3n/10 - 6$.
- SELECT is called recursively on at most $7n/10 + 6$ (i.e., n – (3n/10-6)) elements in Step5.

Worst-Case Running Time of SELECT(A, p, r, $i$):

- Steps 1, 2, and 4: O($n$) time.
  (Step2 consists of O($n$) calls of insertion sort on sets with $\leq$ 5 elements each).
- T($n$) = worst-case running time of SELECT on an array with $n$ elements.
- Step3: $T(\lceil n/5 \rceil)$ time
- Step 5: $T(7n/10 + 6)$ time (follows from previous considerations)
- Thus $T(n) \leq T\left(\lceil\frac{n}{5}\rceil\right) + T\left(\frac{7n}{10} + 6\right) + bn$

We can ignore the ceiling function and the constant 6 in our analysis.

$T(n) \leq bn(1 + \left(\frac{9}{10}\right) + +(\frac{9}{10})^2 + (\frac{9}{10})^3 + \ldots = O(n)$

# Applications

- Quicksort in O(n log n) worst-case time.
- Finding the median in linear time.
- Return the *k* smallest in sorted order.
- Many other applications.

ASU® Ira A. Fulton Schools of Engineering
Arizona State University