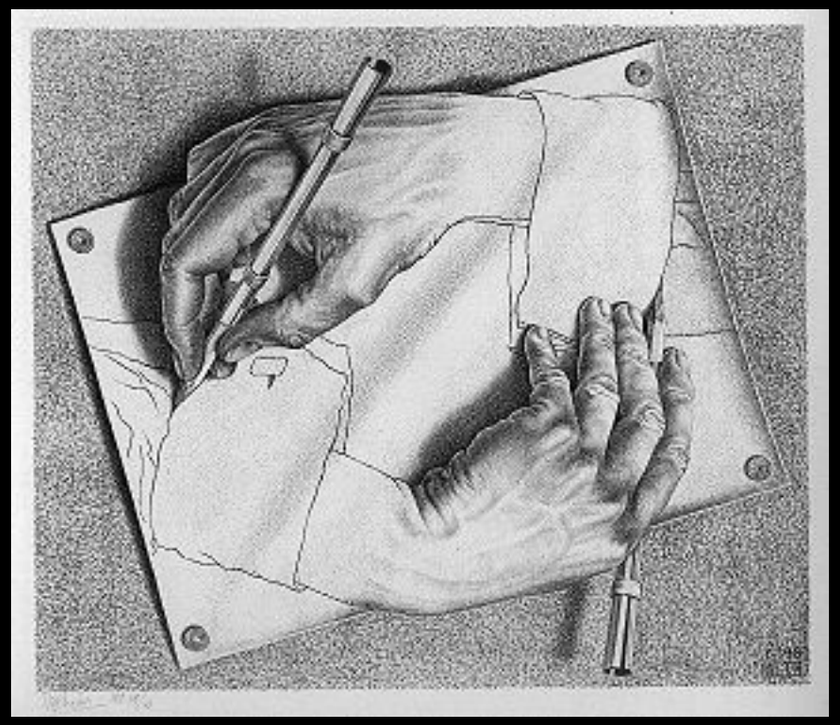
Selecting the k^{th} smallest element

Nishant Mehta

Lecture 13



M. C. Escher (1948): “Drawing Hands”

Selecting Medians and Order Statistics

- Fundamental problem:

  Select the k^{th} smallest element in an unsorted sequence

- **Definition:** An element x is the ***k^{th} order statistic*** of a sequence A if x is the k^{th} smallest element of A

- Selection Problem:
  - Given an array A of n elements and $k \in \{1, 2, \ldots, n\}$,
  - Return the k^{th} order statistic of A
- Example: If n is odd and k = (n+1)/2, we get the median

Selecting Medians and Order Statistics

- Fundamental problem:

  Select the k^{th} smallest element in an unsorted sequence

- **Definition:** An element x is the ***k^{th} order statistic*** of a sequence A if x is the k^{th} smallest element of A

- Selection Problem:
  - Given an array A of n elements and $k \in \{1, 2, \ldots, n\}$,
  - Return the k^{th} order statistic of A

- Example: If n is odd and k = (n+1)/2, we get the median

For simplicity we'll assume all n elements are distinct
(no big ideas are needed for the general case)

A naïve solution

A sorting-based approach:

1. Sort A in increasing order

2. Output the k^{th} element of the sorted sequence

How long does this take?

Is this the best possible?

A naïve solution

A sorting-based approach:

1. Sort A in increasing order

2. Output the k^{th} element of the sorted sequence

How long does this take? $O(n \log n)$

Is this the best possible?

A naïve solution

A sorting-based approach:

1. Sort A in increasing order

2. Output the k^{th} element of the sorted sequence

How long does this take? $O(n \log n)$

Is this the best possible? **No!**

JOURNAL OF COMPUTER AND SYSTEM SCIENCES 7, 448–461 (1973)

Time Bounds for Selection*

MANUEL BLUM, ROBERT W. FLOYD, VAUGHAN PRATT,
RONALD L. RIVEST, AND ROBERT E. TARJAN

Department of Computer Science, Stanford University, Stanford, California 94305

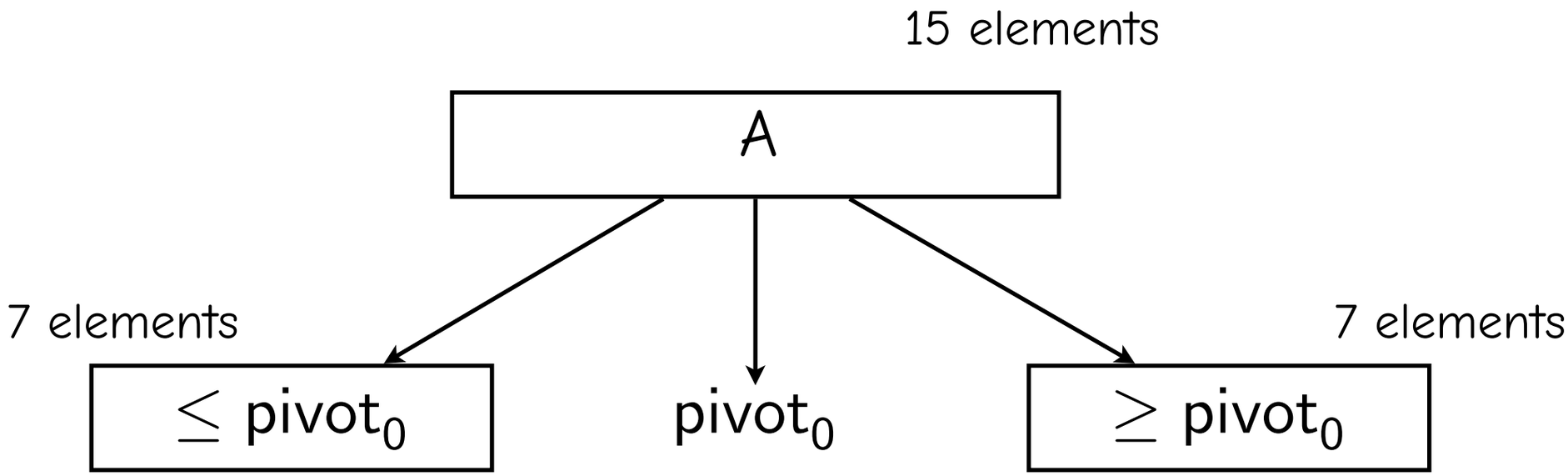
Received November 14, 1972

The number of comparisons required to select the i-th smallest of n numbers is shown to be at most a linear function of n by analysis of a new selection algorithm—PICK. Specifically, no more than $5.4305\dot{}\, n$ comparisons are ever required. This bound is improved for extreme values of i, and a new lower bound on the requisite number of comparisons is also proved.

O(n) is possible!!

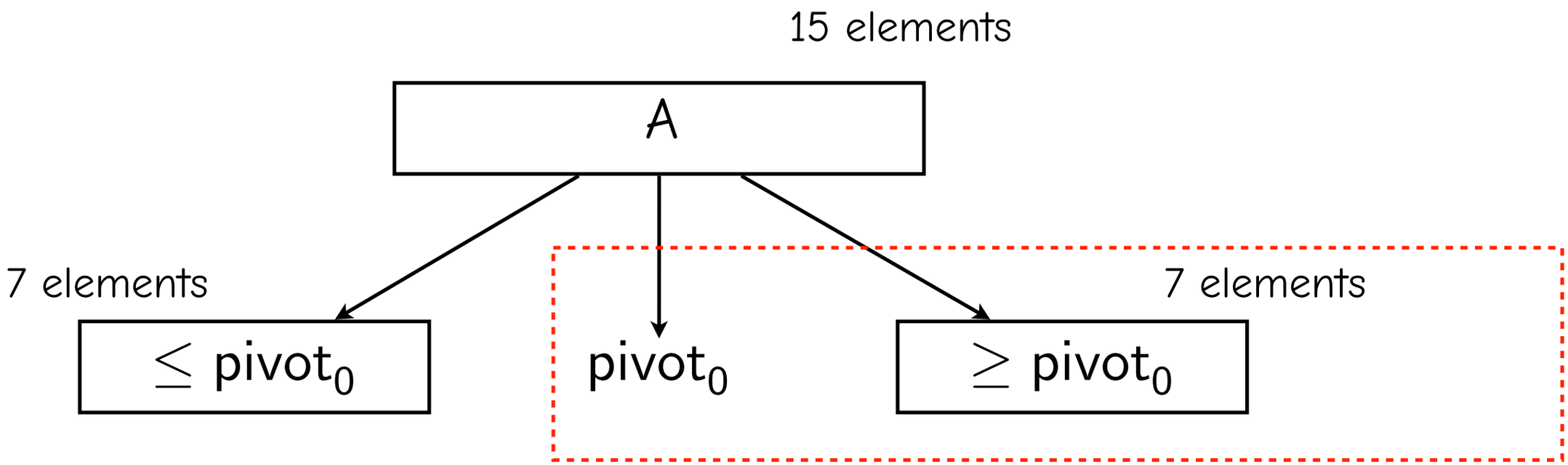
Quickselect: Quicksort with pruning

Goal: Select the 6th smallest element



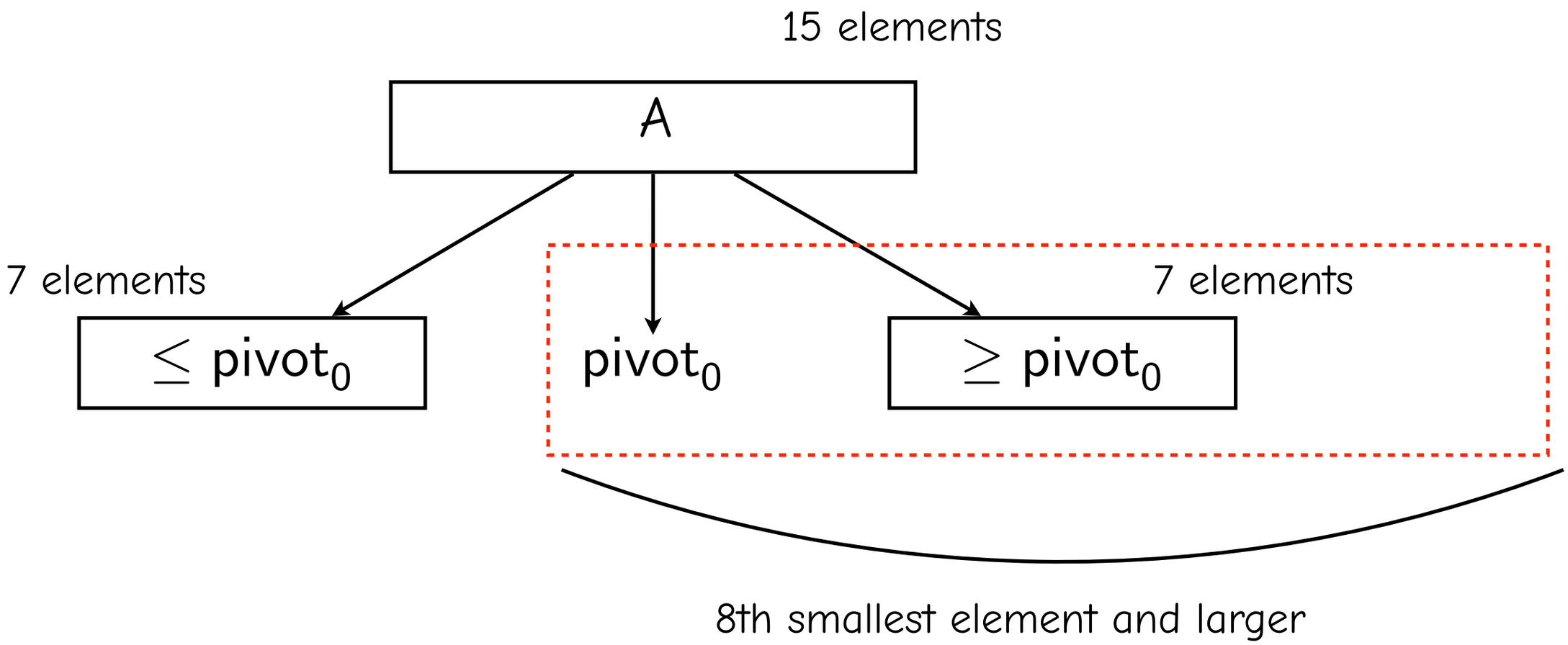
Quickselect: Quicksort with pruning

Goal: Select the 6th smallest element



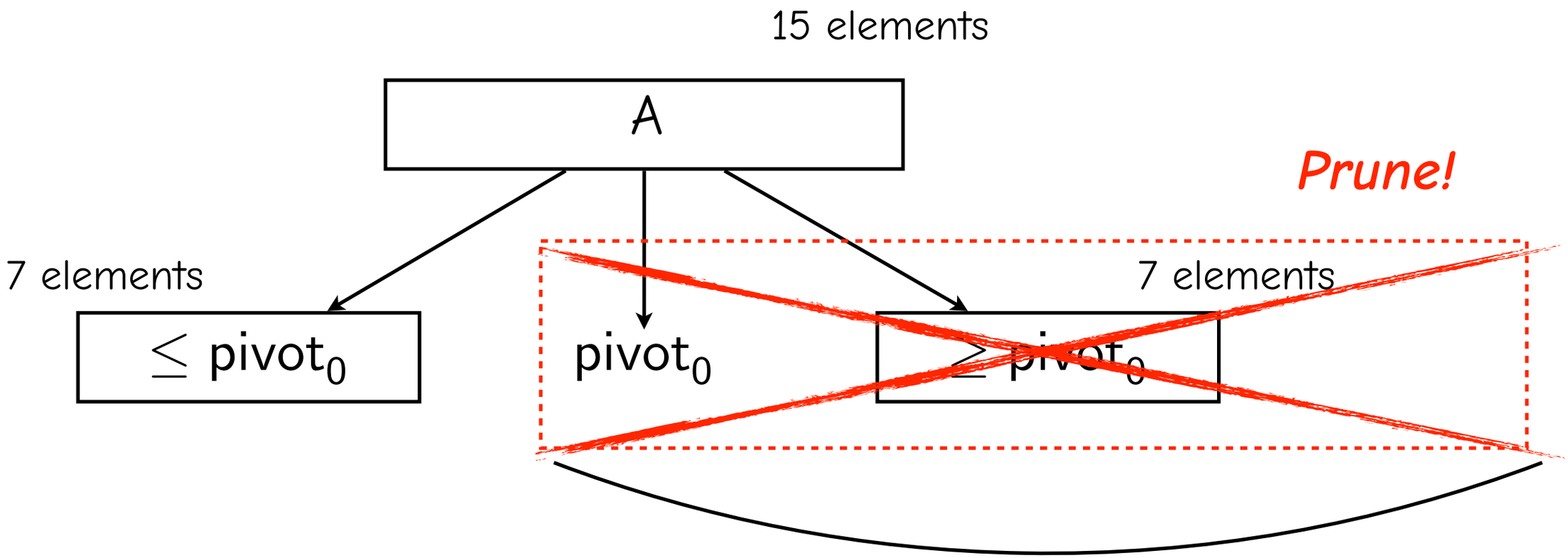
Quickselect: Quicksort with pruning

Goal: Select the 6th smallest element



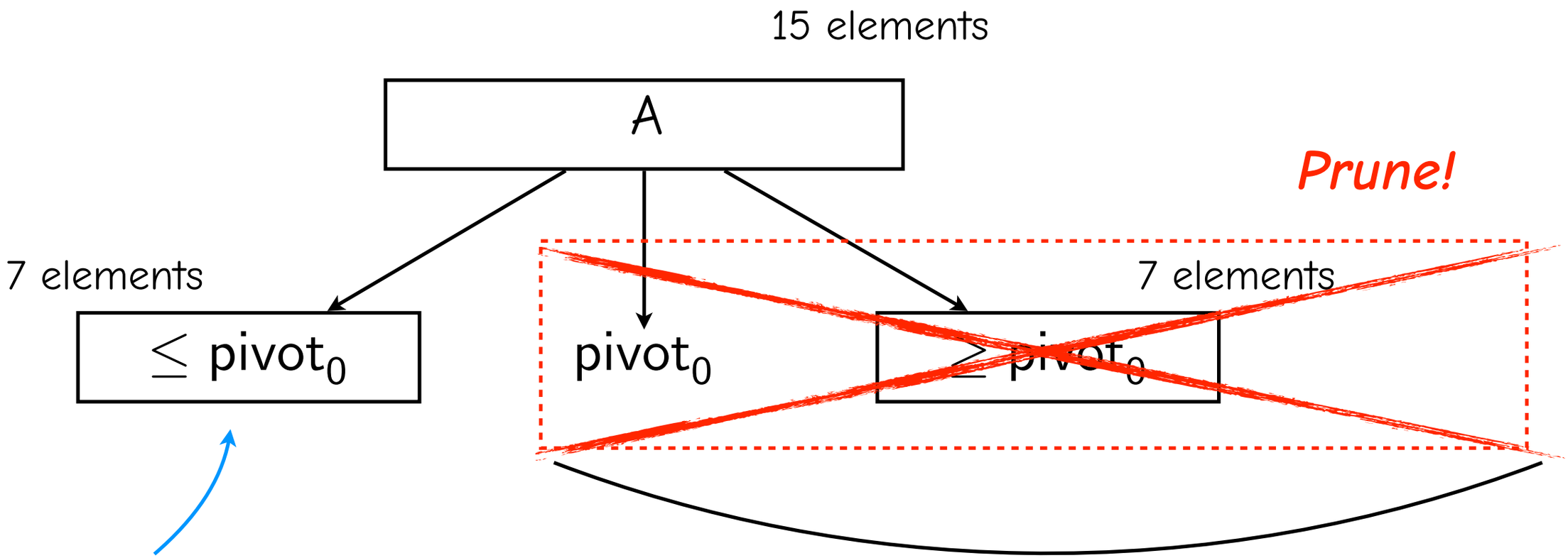
Quickselect: Quicksort with pruning

Goal: Select the 6th smallest element



Quickselect: Quicksort with pruning

Goal: Select the 6th smallest element

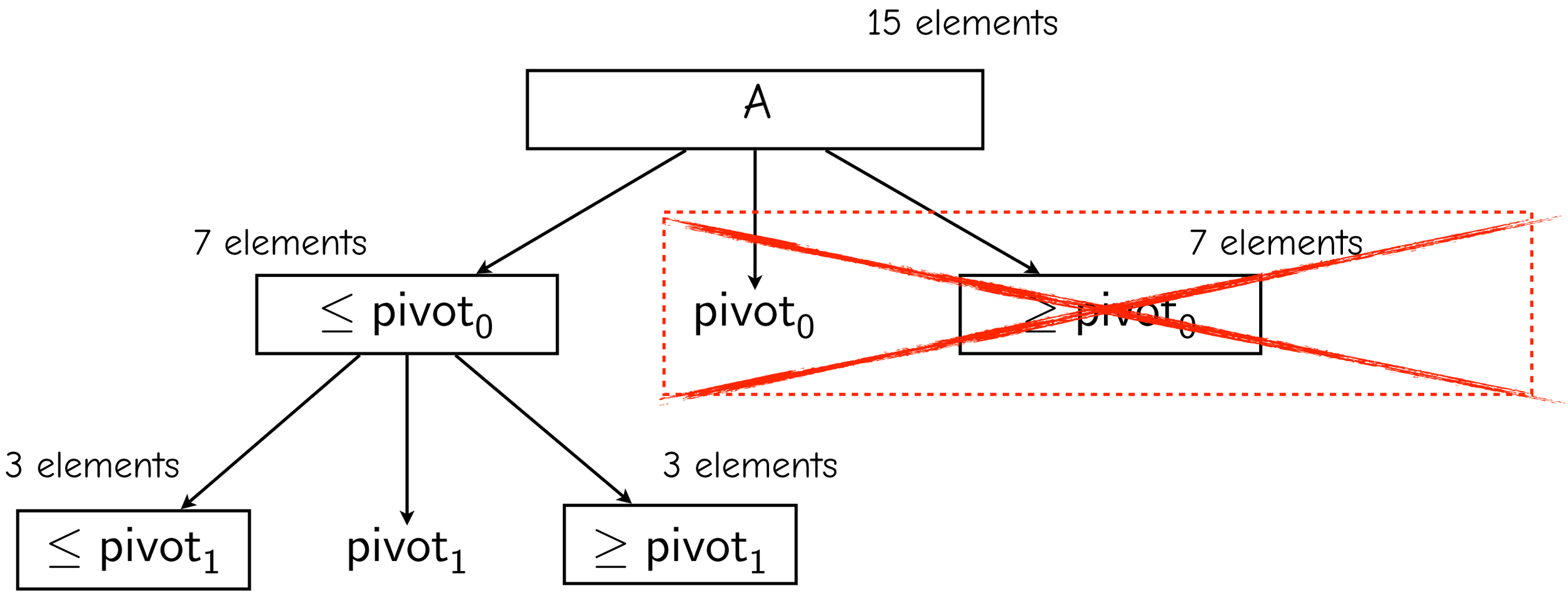


Quickselect with k=6

8th smallest element and larger

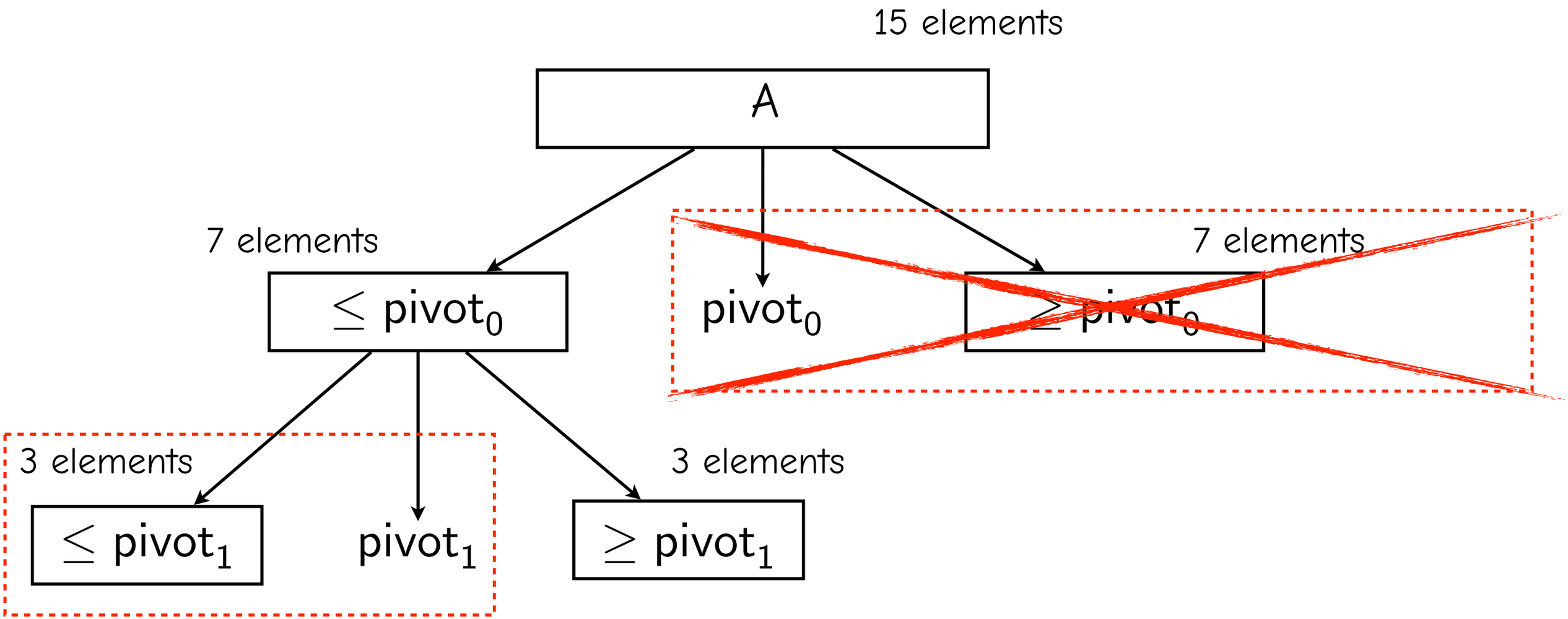
Quickselect: Quicksort with pruning

Goal: Select the 6th smallest element



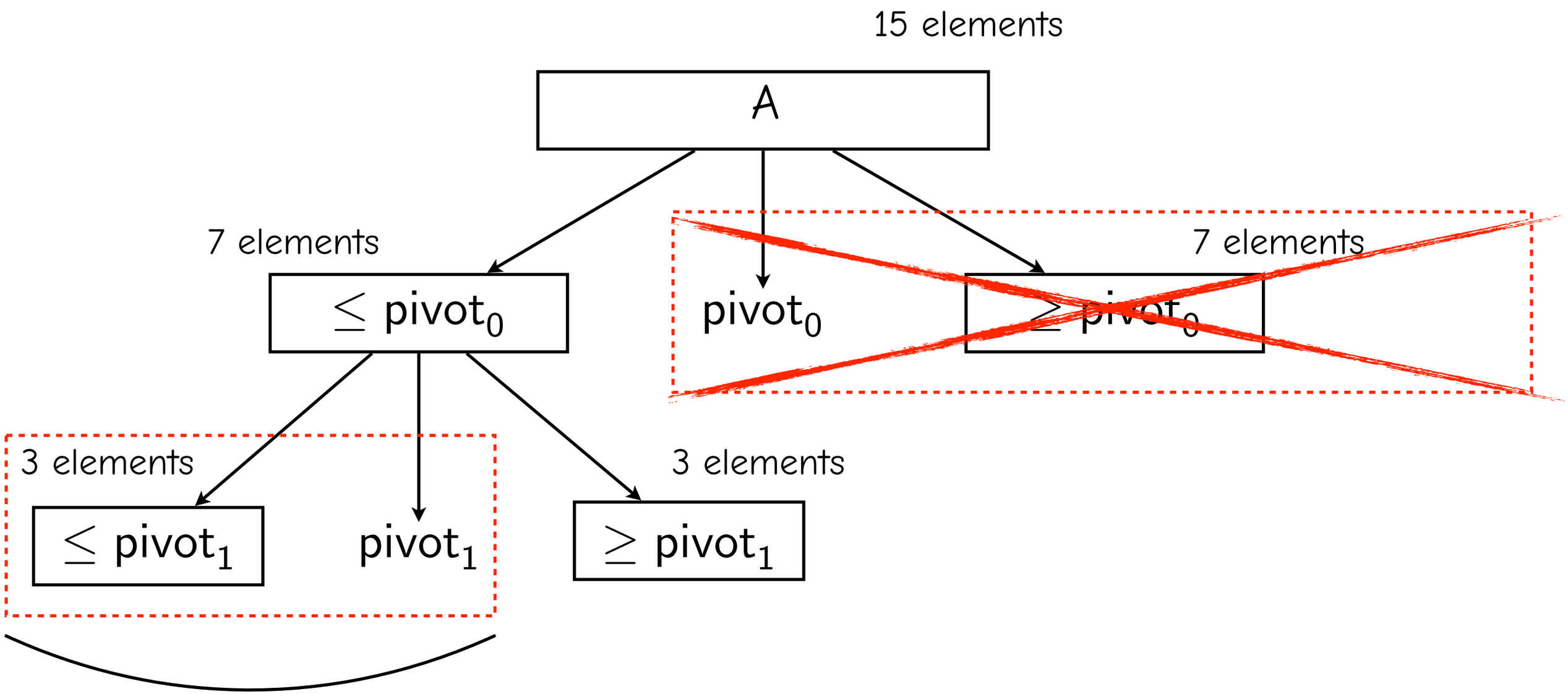
Quickselect: Quicksort with pruning

Goal: Select the 6th smallest element



Quickselect: Quicksort with pruning

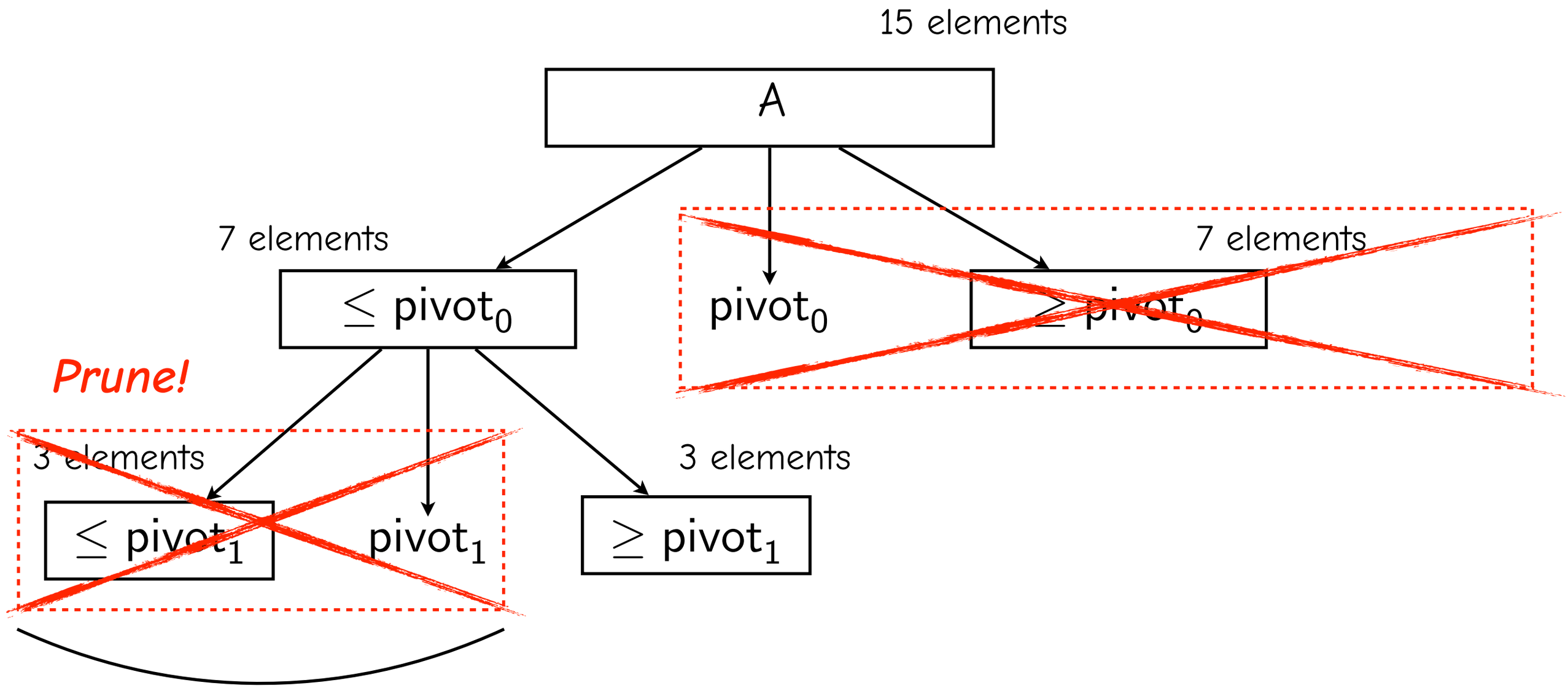
Goal: Select the 6th smallest element



4th smallest element and smaller

Quickselect: Quicksort with pruning

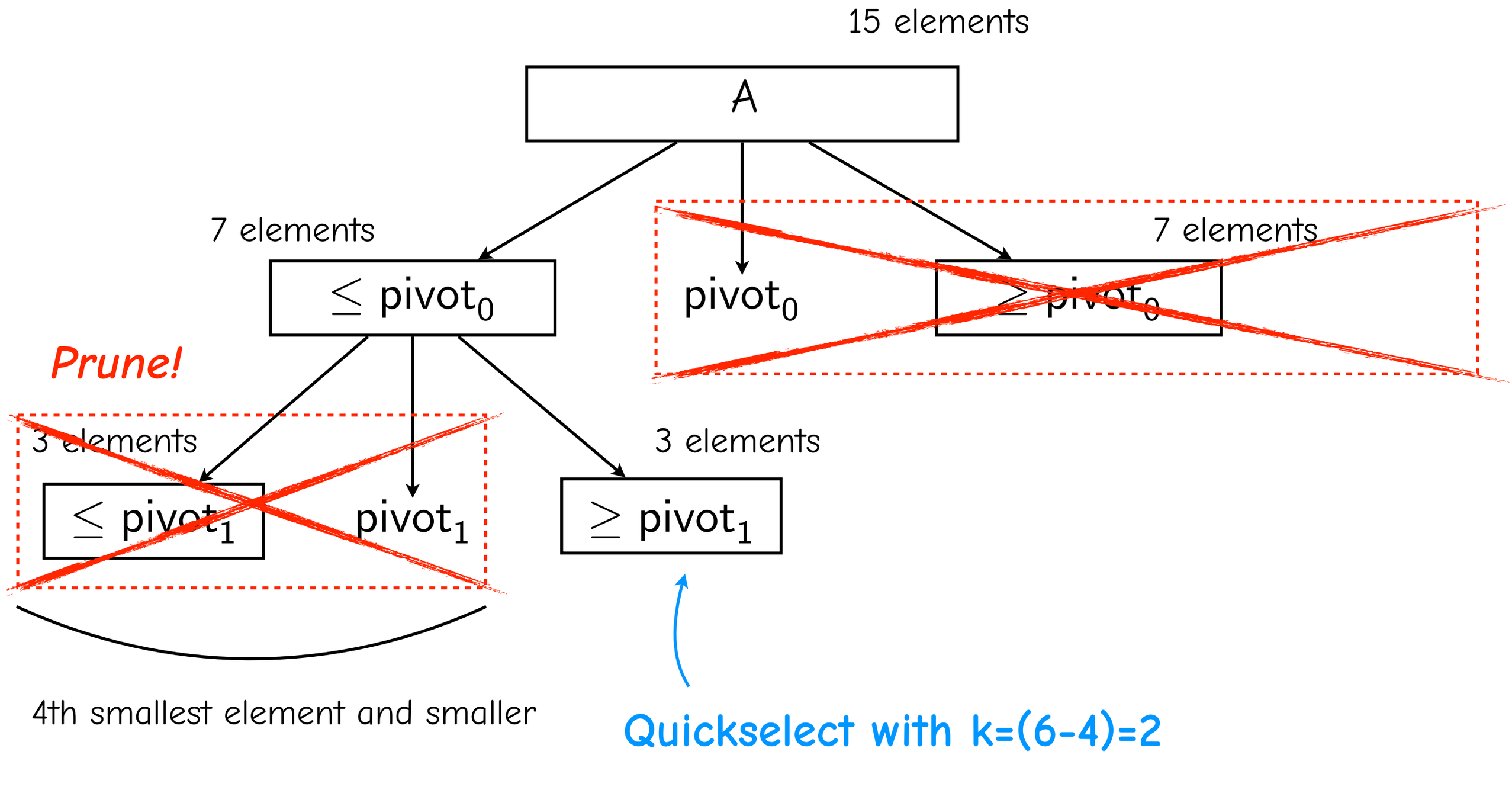
Goal: Select the 6th smallest element



4th smallest element and smaller

Quickselect: Quicksort with pruning

Goal: Select the 6th smallest element



Quickselect

```
Quickselect(A, k):

    If A.length() == 1

        Return A[0]

    p = PickPivot(A)     // how to pick pivot? To be explained later!

    [L, G] = Partition(A, p) // 'L' for "less than", 'G' for "greater than"

    If k ≤ length(L)

        Return Quickselect(L, k)

    ElseIf k == (length(L) + 1)

        Return p

    Else  // k > (length(L) + 1)

        Return Quickselect(G, k - length(L) - 1)
```

Quickselect with Arbitrary Pivot - Optimistic analysis (Warm-up)

- Suppose we always take the pivot to be the first element in the sequence and are so lucky that it always is the median
- Then PickPivot(A) just returns A[0] and so costs 1
- Quickselect on sequence of length n either:

  (a) calls Quickselect on sequence of length at most $\lfloor n/2 \rfloor$

OR

  (b) returns the k^{th} order statistic itself

Quickselect with Arbitrary Pivot - Optimistic analysis (Warm-up)

- Suppose we always take the pivot to be the first element in the sequence and are so lucky that it always is the median
- Then PickPivot(A) just returns A[0] and so costs 1
- Quickselect on sequence of length n either:

  (a) calls Quickselect on sequence of length at most $\lfloor n/2 \rfloor$

OR

  (b) returns the k^{th} order statistic itself

So: $T(n) \leq T(n/2) + cn$

Quickselect with Arbitrary Pivot - Optimistic analysis (Warm-up)

$$
\begin{aligned}
T(n) &\leq T(n/2) + cn \\
&\leq T(n/4) + cn/2 + cn \\
&\leq T(n/8) + cn/4 + cn/2 + cn \\
&\vdots \\
&\leq T(1) + (cn)\sum_{j=0}^{\infty} 2^{-j} \\
&= T(1) + 2cn \\
&= O(n)
\end{aligned}
$$

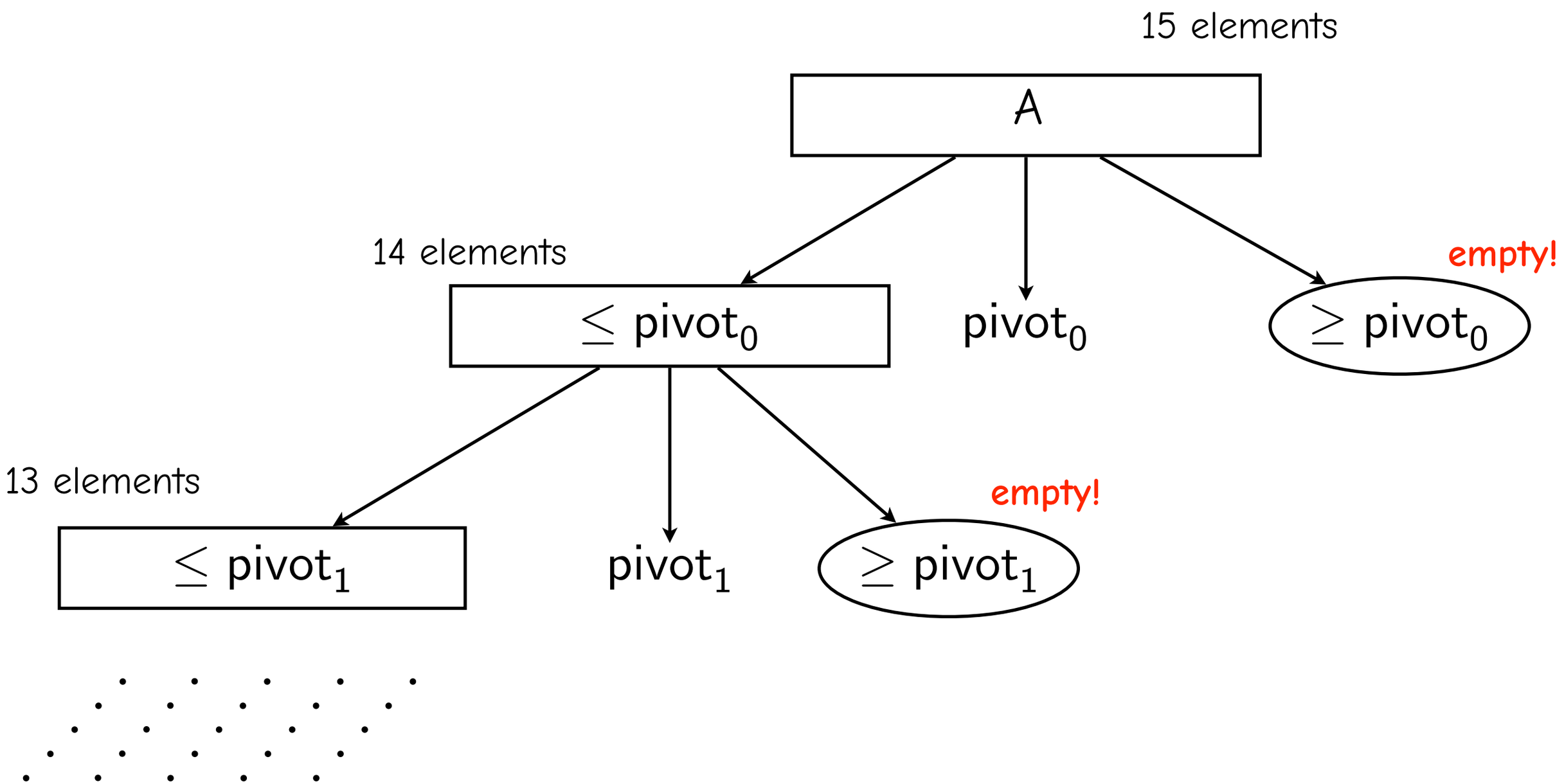
Quickselect with Arbitrary Pivot - Optimistic analysis (Warm-up)

$$
\begin{aligned}
T(n) &\leq T(n/2) + cn \\
&\leq T(n/4) + cn/2 + cn \\
&\leq T(n/8) + cn/4 + cn/2 + cn \\
&\vdots \\
&\leq T(1) + (cn)\sum_{j=0}^{\infty} 2^{-j} \\
&= T(1) + 2cn \\
&= O(n)
\end{aligned}
$$

This is great! But we cheated by assuming that taking the pivot as the first element always gives us the median

Quickselect with Arbitrary Pivot - Worst-case analysis

Goal: Select the 6th smallest element



Quickselect with Arbitrary Pivot - Worst-case analysis

$$
\begin{aligned}
T(n) &= T(n-1) + cn \\
&= T(n-2) + c(n-1) + cn \\
&= T(n-3) + c(n-2) + c(n-1) + cn \\
&\vdots \\
&= T(1) + c\sum_{j=2}^{n} j \\
&= O(1) + c \cdot \left(\frac{n(n+1)}{2} - 1 \right) \\
&= \Omega(n^2)
\end{aligned}
$$

Picking a good pivot

- Median pivots are the best possible choice
- But if we knew how to get the median, we would be done!
- Idea: Try to find an "approximate median" using less work
  - Find the median of a well-chosen subset of the sequence

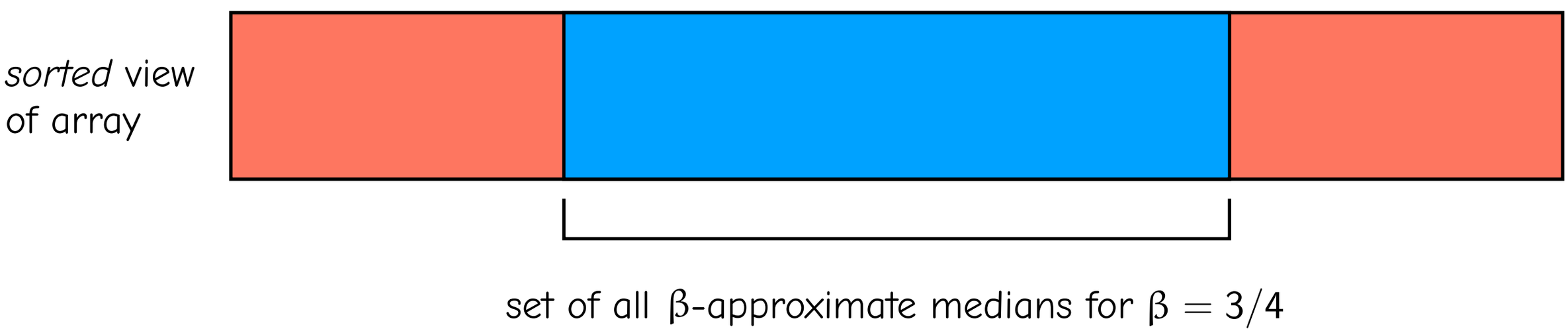
Picking a good pivot

Definition: Let β satisfy $1/2 \leq \beta < 1$. We say that an element *m* of sequence A is a β-approximate median of A if:

At most βn elements of A are less than *m*

and

At most βn elements of A are greater than *m*



Is a β-approximate median a good pivot?

- If the pivot is a β-approximate median, then calling Quickselect on a sequence of n points leads to both L and G that each are of size at most βn

- If Quickselect always uses a β-approximate median, then at level j of Quickselect (i.e. inside the j^{th} recursive call), both L and G each can have size at most $\beta^j n$

Is a β-approximate median a good pivot?

- If the pivot is a β-approximate median, then calling Quickselect on a sequence of n points leads to both L and G that each are of size at most βn

- If Quickselect always uses a β-approximate median, then at level j of Quickselect (i.e. inside the j^{th} recursive call), both L and G each can have size at most $\beta^j n$

$$
\begin{aligned}
T(n) &\leq T(\beta n) + cn \\
&\leq T(\beta^2 n) + c\beta n + cn \\
&\leq T(\beta^3 n) + c\beta^2 n + c\beta n + cn \\
&\vdots \\
&\leq T(1) + cn \sum_{j=0}^{k} \beta^j \qquad \Bigg] = O(1) + \frac{cn}{1-\beta}
\end{aligned}
$$

Quickselect with β-approximate median

So, runtime of Quickselect using a β-approximate median is

$$T(n) = O(1) + \frac{cn}{1-\beta} = O(n)$$

Quickselect with β-approximate median

So, runtime of Quickselect using a β-approximate median is

$$T(n) = O(1) + \frac{cn}{1-\beta} = O(n)$$

This is great, but we are still cheating...

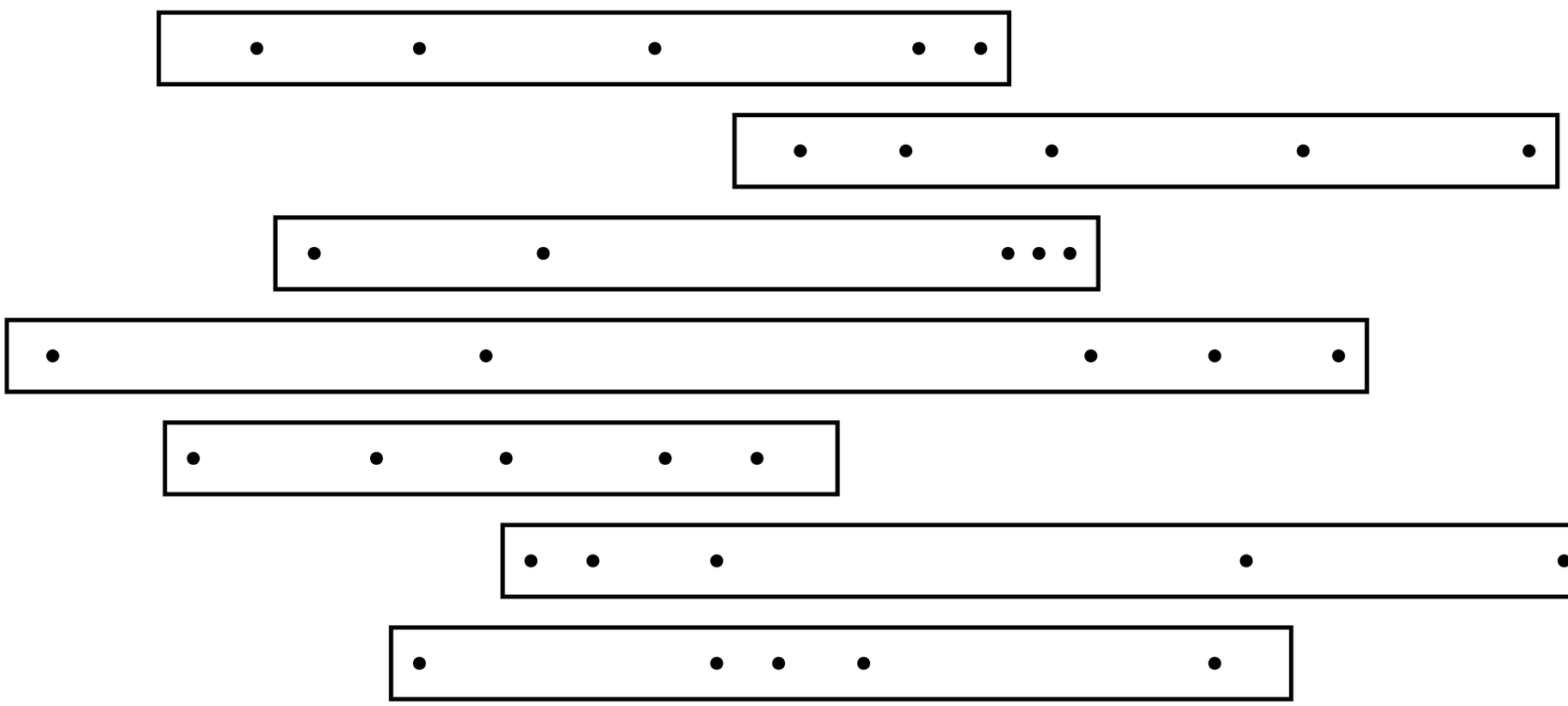
We need a way to find β-approximate median AND must account for the computational cost for doing so

Computing a β-approximate median

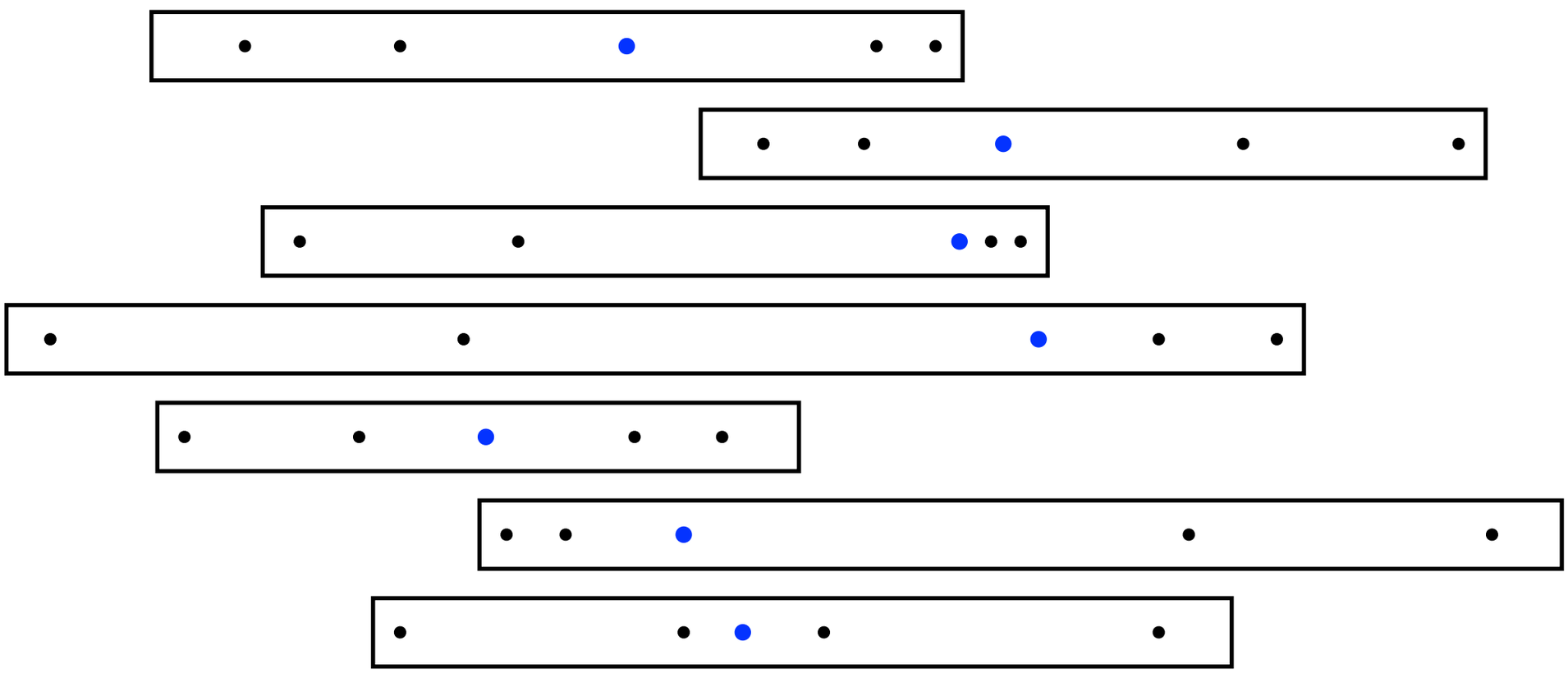
- Partition sequence into n/5 segments, each of size 5
  - For simplicity, we ignore the fact that the last segment might have size less than 5.
- Find the median of each segment.
- Find the median of the n/5 medians (somehow)

Median of medians

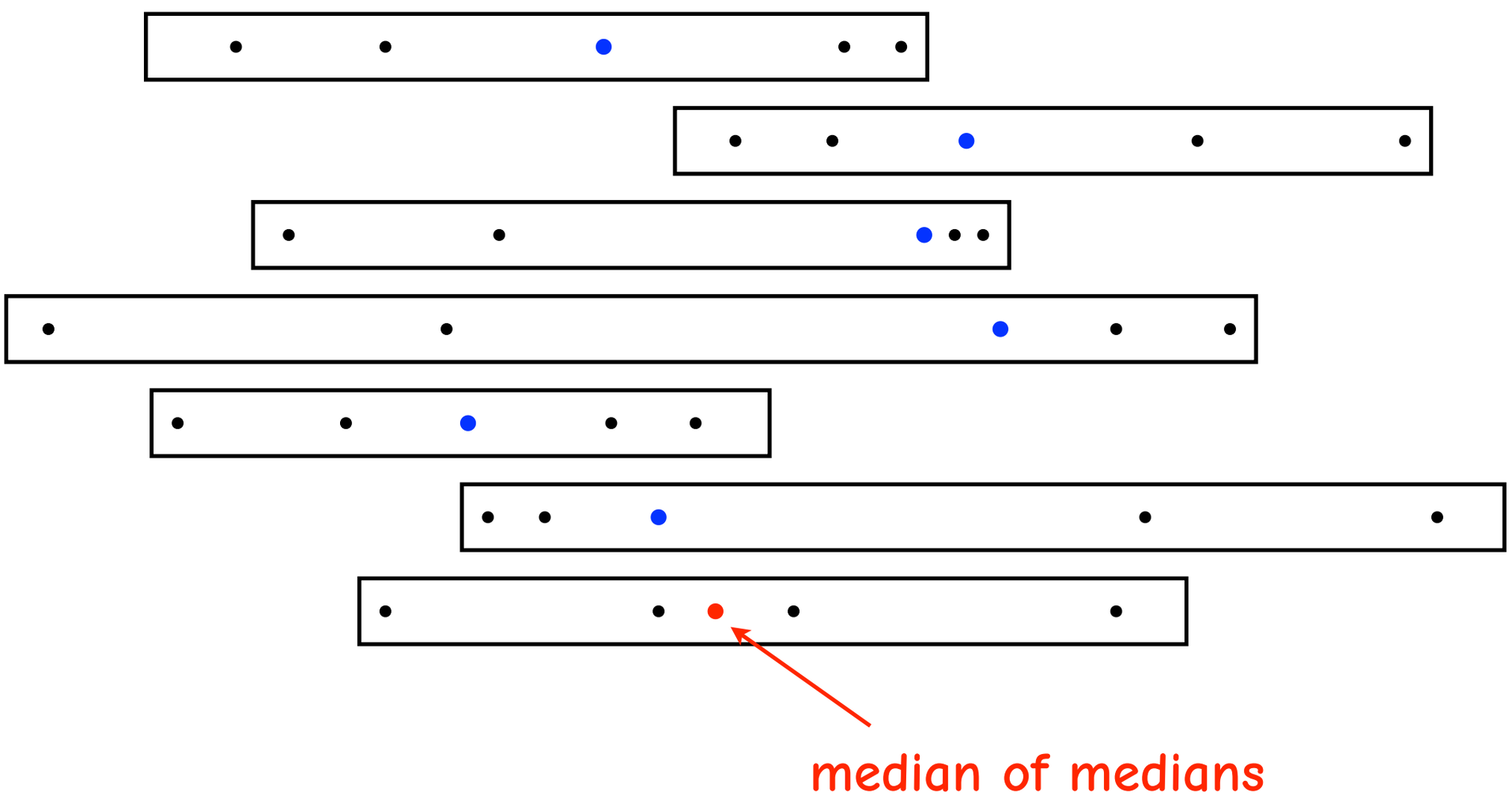
Median of medians



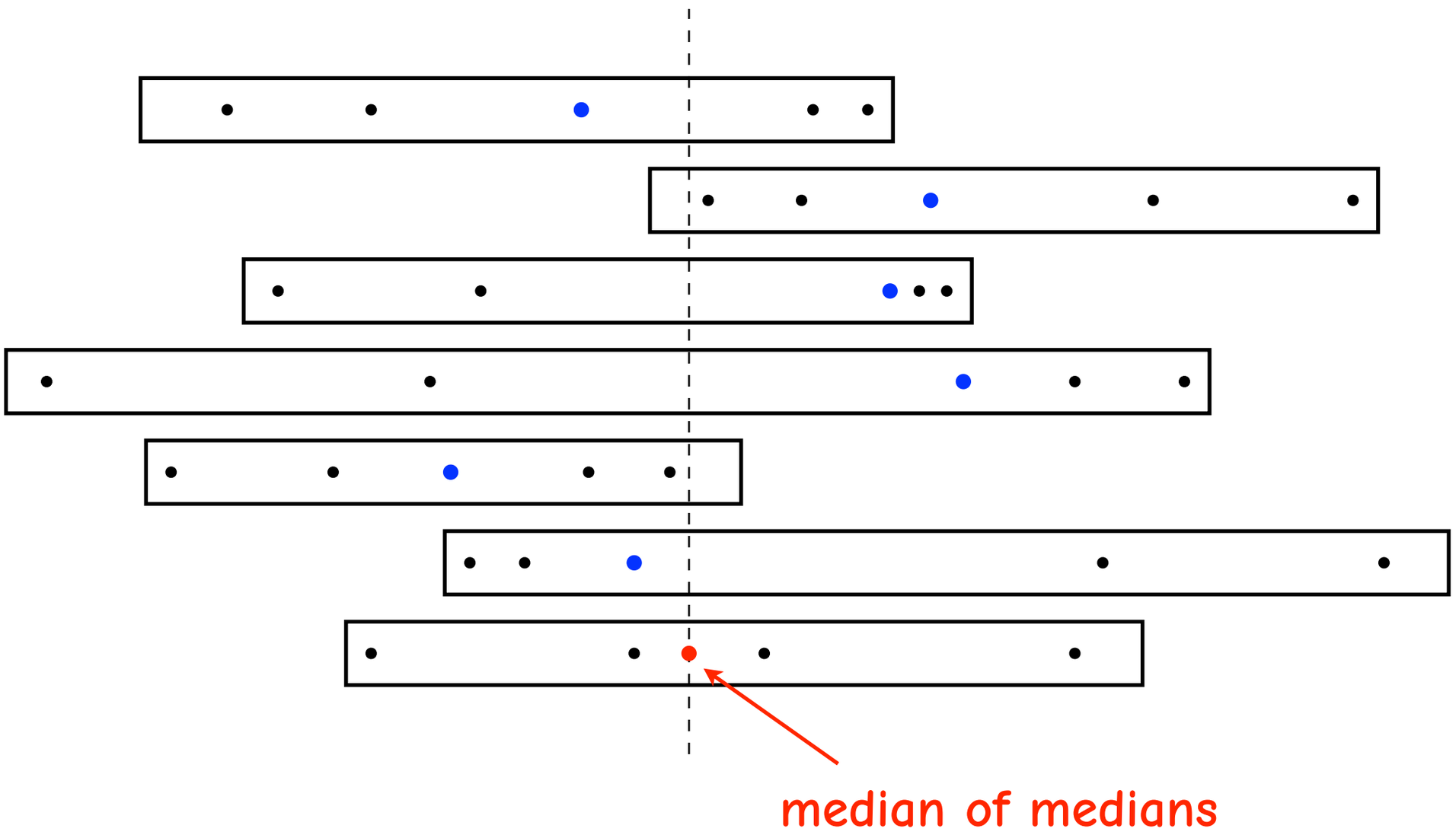
Median of medians



Median of medians



Median of medians



Quickselect with median-of-medians pivot

- What does it cost to compute all the medians?
  - For each segment of length 5:

    Sort to get median. Cost: O(1)

    ⎤ n/5 such segments
    ⎦ Total cost: O(n)

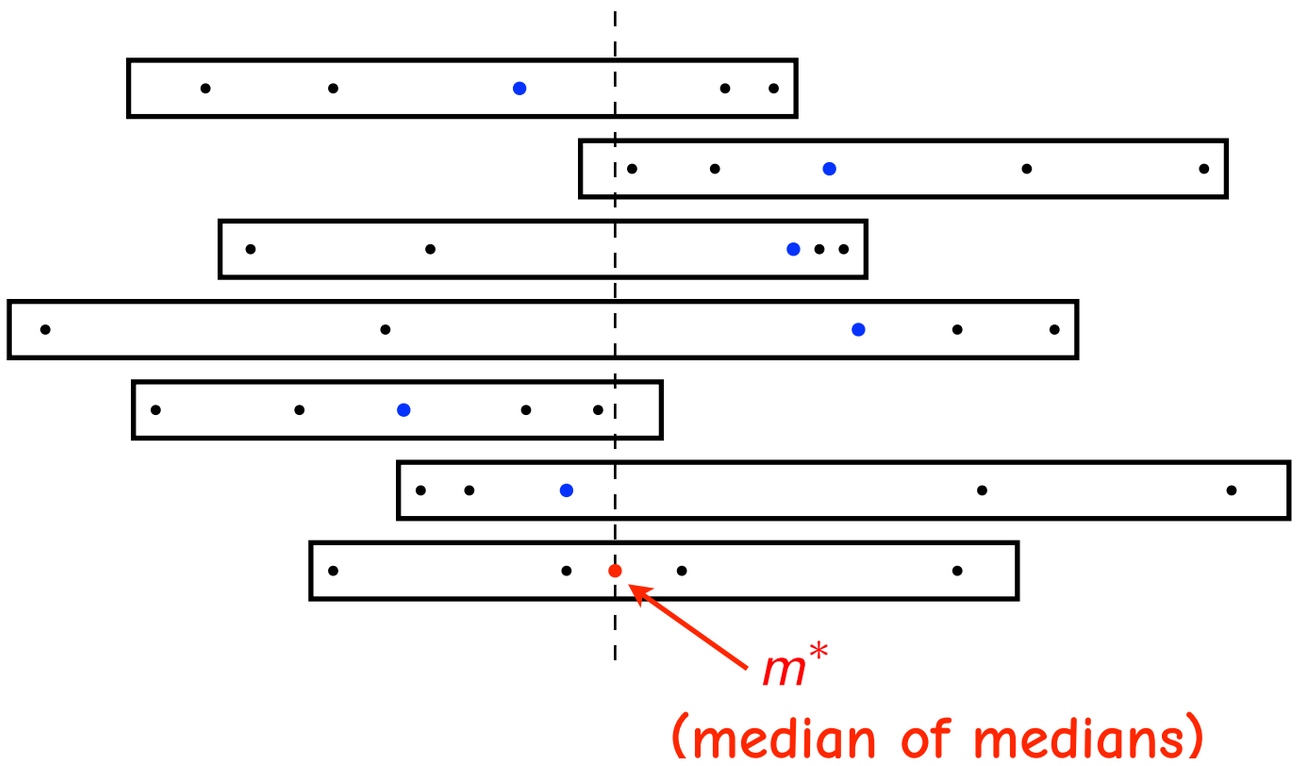
- Two outstanding problems:

  (1) Is median of medians a good pivot,
  i.e. is it a β-approximate median?

  (2) How do you efficiently compute median of medians?

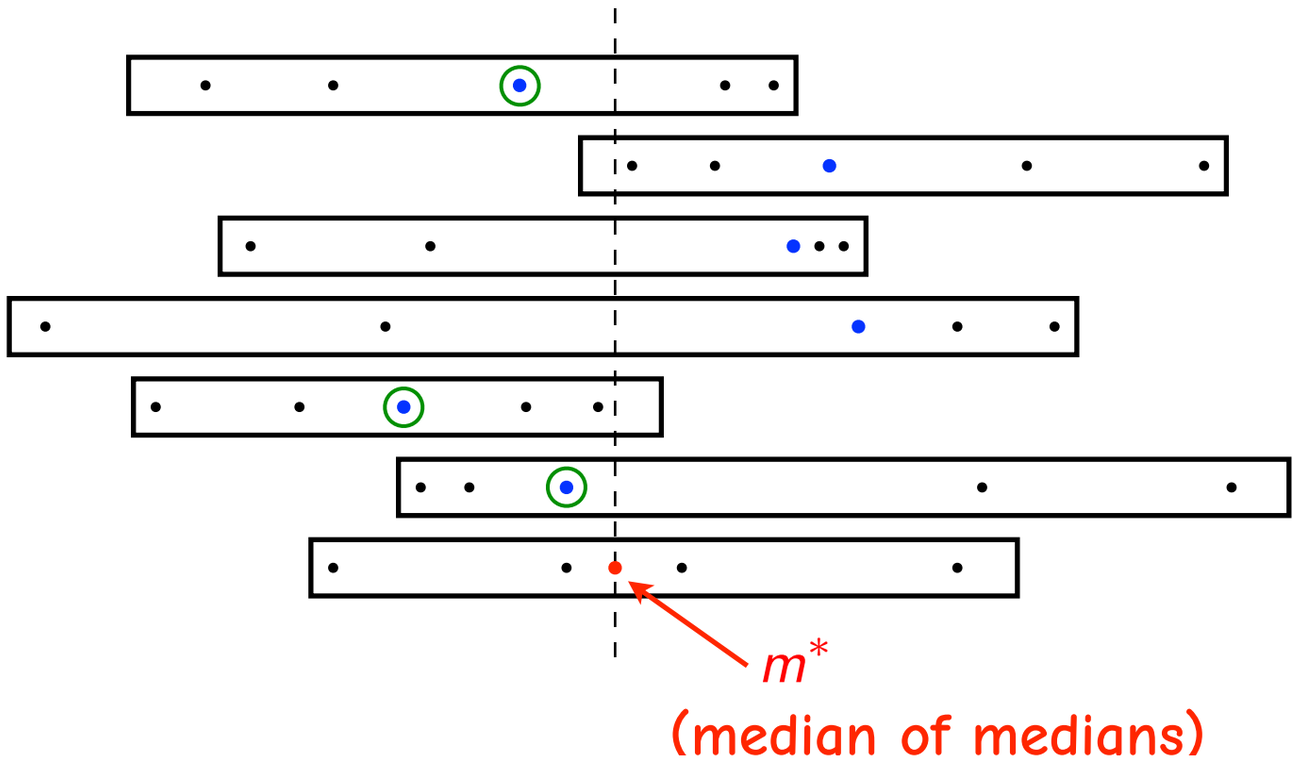
Quickselect with median-of-medians pivot

- Is median of medians a good pivot,
  i.e. is it a β-approximate median?



Quickselect with median-of-medians pivot

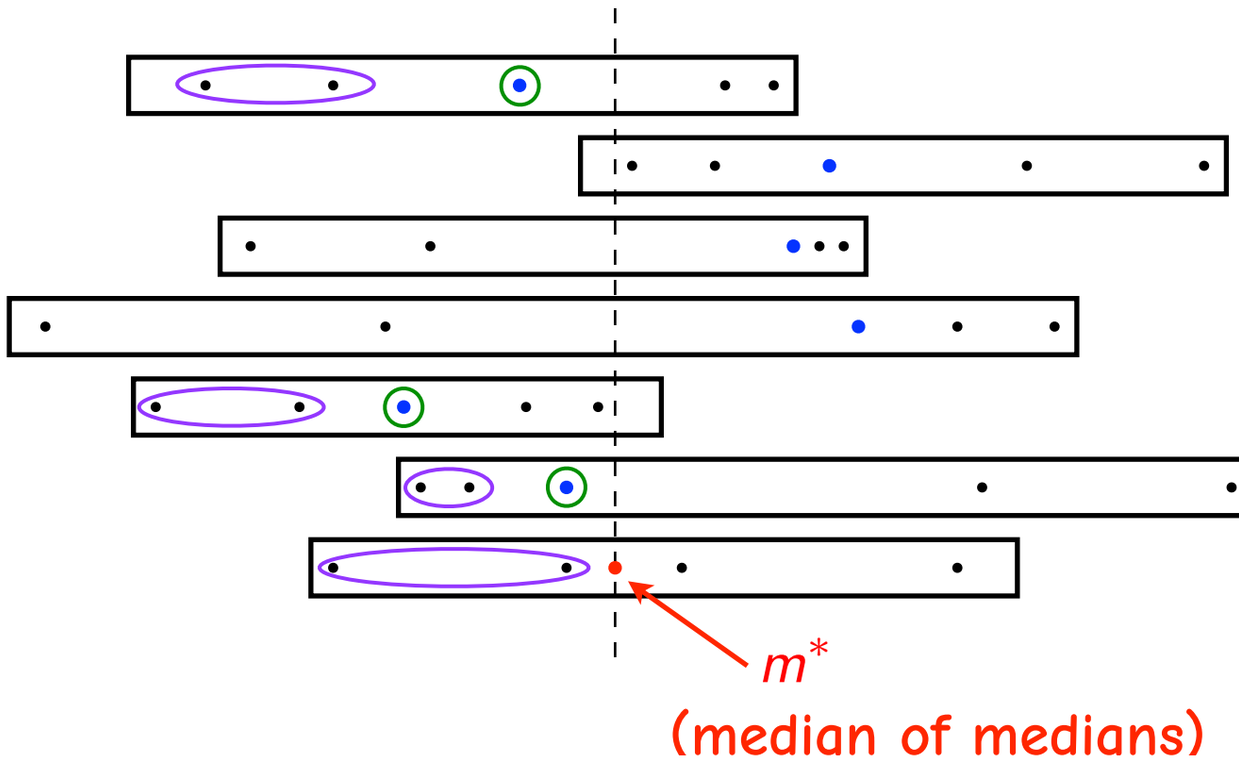
- Is median of medians a good pivot, i.e. is it a β-approximate median?



- $n/5$ medians, of which $n/10$ of are less than or equal to m^*

Quickselect with median-of-medians pivot

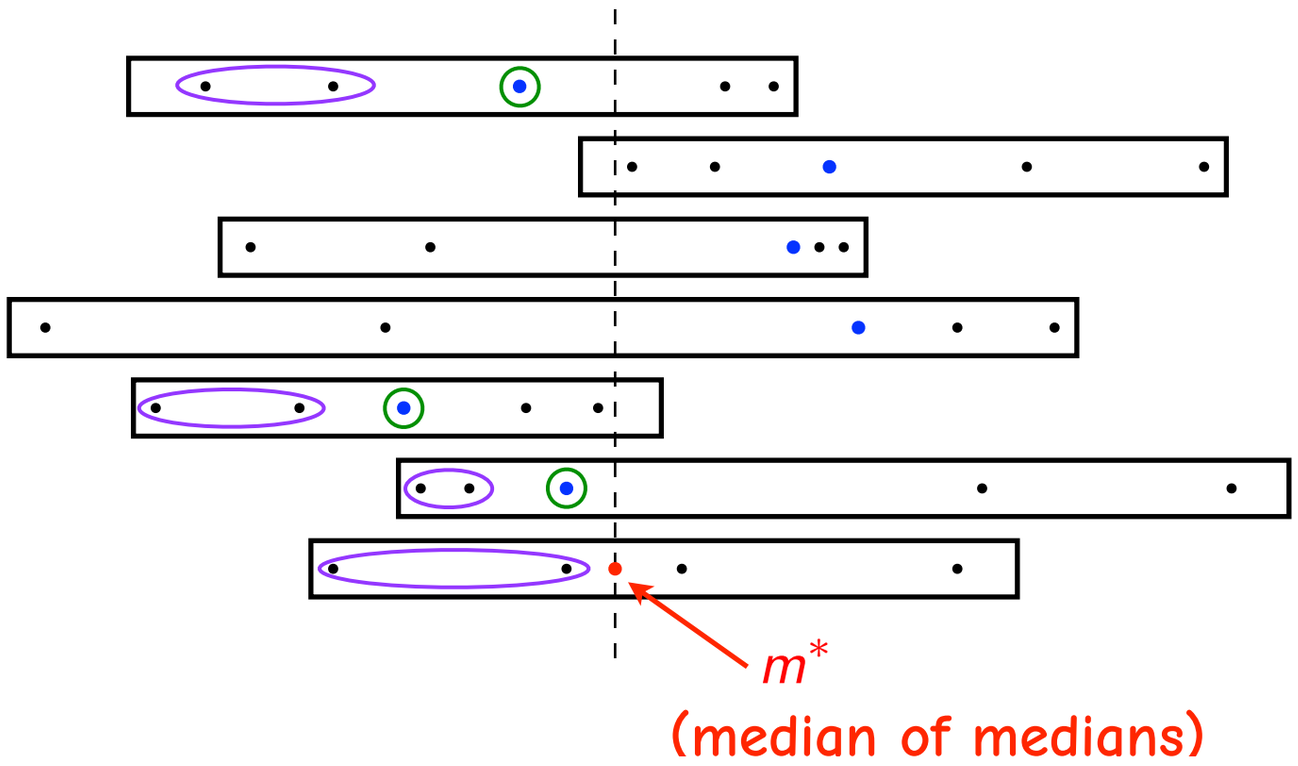
- Is median of medians a good pivot, i.e. is it a β-approximate median?



- $n/5$ medians, of which $n/10$ of are less than or equal to m^*
- For each such median, 2 more elements are less than m^*

Quickselect with median-of-medians pivot

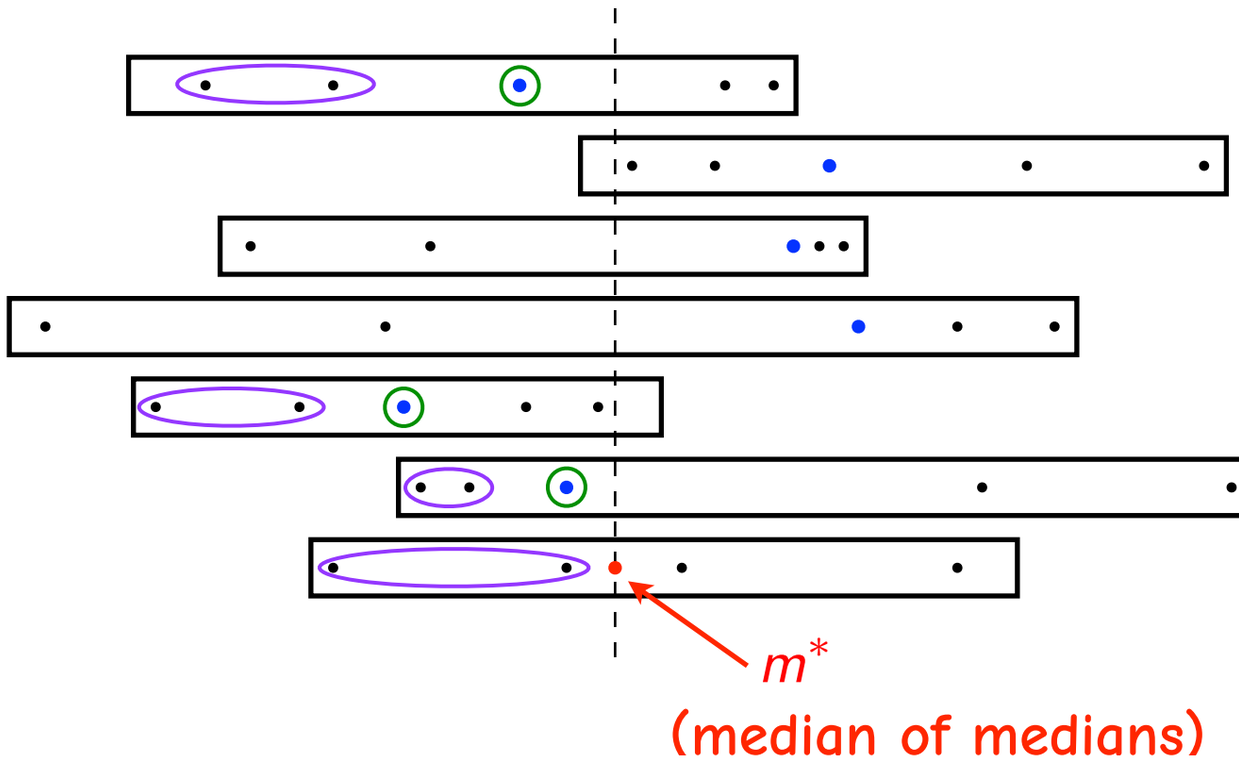
- Is median of medians a good pivot, i.e. is it a β-approximate median?



- $n/5$ medians, of which $n/10$ of are less than or equal to m^*
- For each such median, 2 more elements are less than m^*
- At least $(3n/10) - 1$ elements less than m^*

Quickselect with median-of-medians pivot

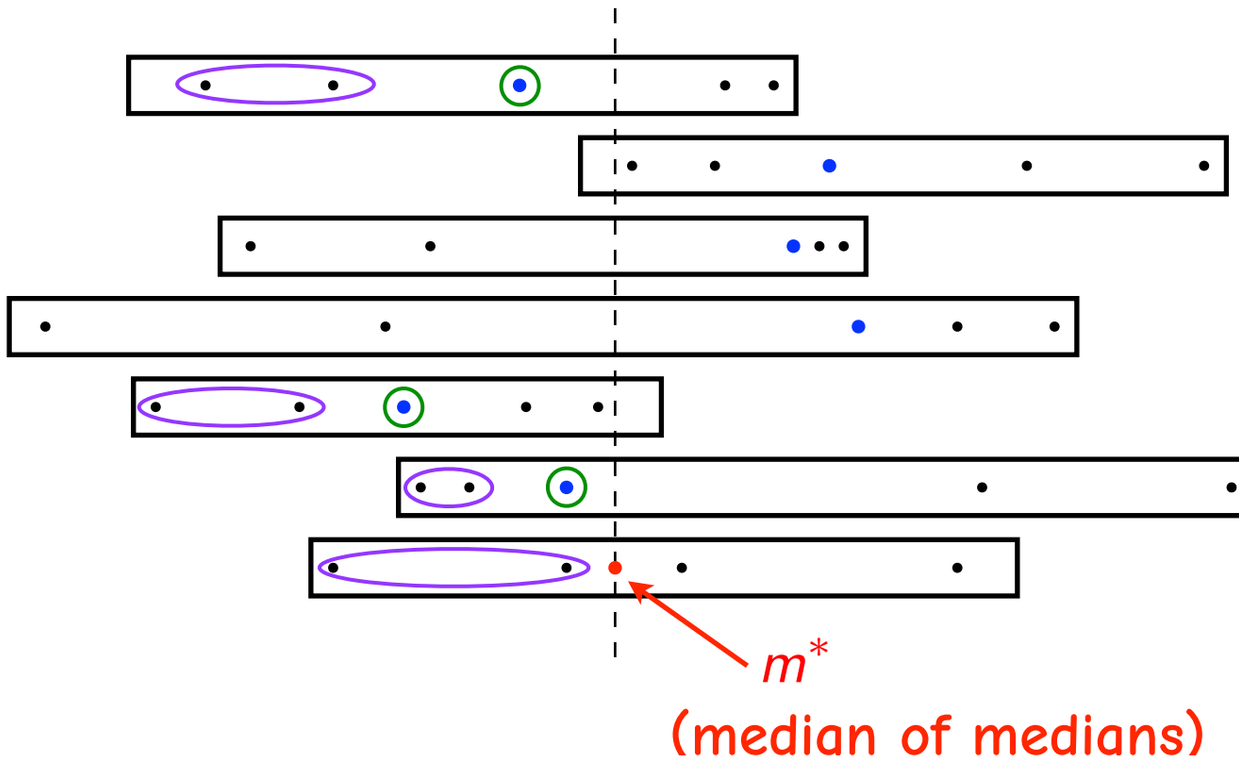
- Is median of medians a good pivot, i.e. is it a β-approximate median?



- $n/5$ medians, of which $n/10$ of are less than or equal to m^*
- For each such median, 2 more elements are less than m^*
- At least $(3n/10) - 1$ elements less than m^*
- Hence, at most $7n/10$ elements are greater than m^*

Quickselect with median-of-medians pivot

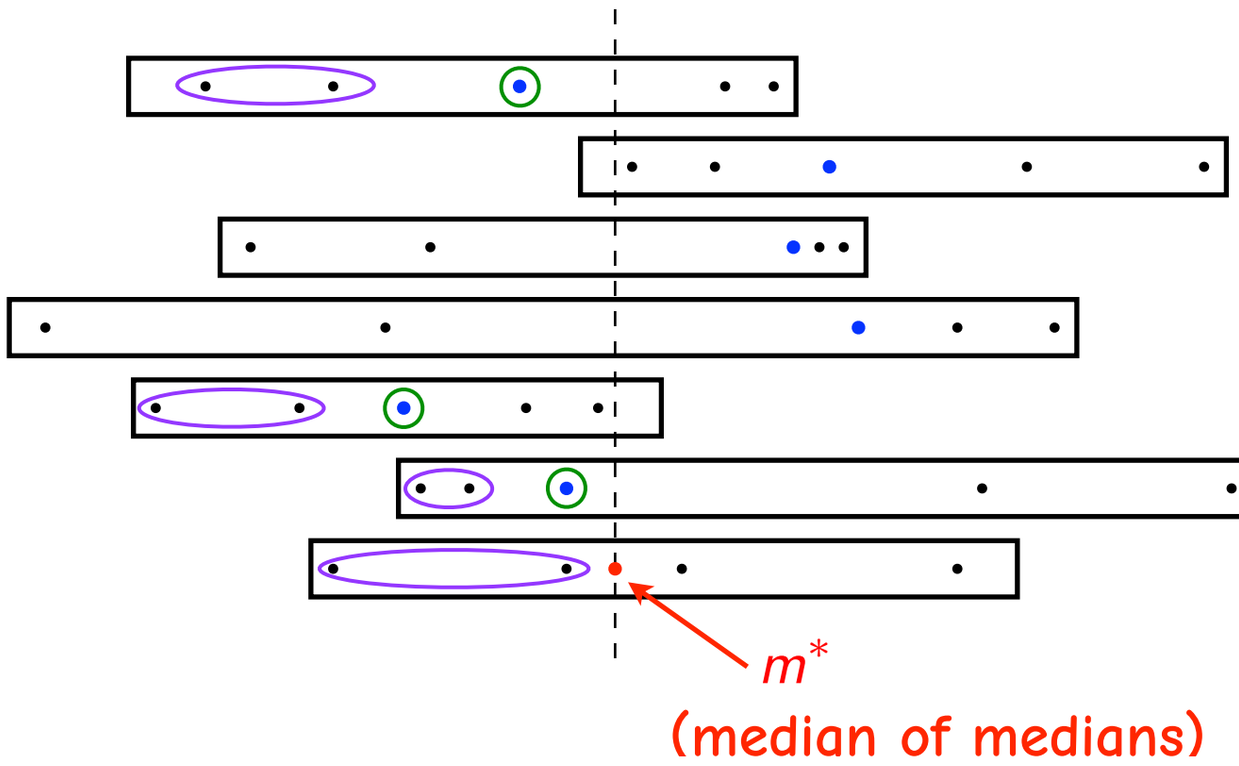
- Is median of medians a good pivot, i.e. is it a β-approximate median?



- $n/5$ medians, of which $n/10$ of are less than or equal to m^*
- For each such median, 2 more elements are less than m^*
- At least $(3n/10) - 1$ elements less than m^*
- Hence, at most $7n/10$ elements are greater than m^*
- By symmetry, at most $7n/10$ elements are less than m^*

Quickselect with median-of-medians pivot

- Is median of medians a good pivot, i.e. is it a β-approximate median?



- $n/5$ medians, of which $n/10$ of are less than or equal to m^*
- For each such median, 2 more elements are less than m^*
- At least $(3n/10) - 1$ elements less than m^*
- Hence, at most $7n/10$ elements are greater than m^*
- By symmetry, at most $7n/10$ elements are less than m^*

m^* is a β-approximate median (for $\beta = 7/10$)

Quickselect with median-of-medians pivot

- How to compute median of medians?
- Idea! Recursively call Quickselect(medians, (n/5)/2)
- Can this really work?
  - Original sequence was of length n
  - Sequence of medians is of length only n/5
  - Seems like a divide-and-conquer strategy

Run-time Analysis

$$T(n) = T(n/5) + T(7n/10) + cn$$

We could use substitution method to analyze complexity

Instead, let's use "stack of bricks" view of the recursion tree

Set $\alpha = 1/5$ and $\beta = 7/10$

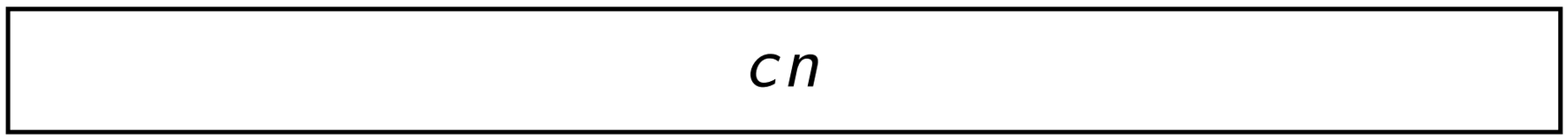
Run-time Analysis

$$T(n) = T(n/5) + T(7n/10) + cn$$

We could use substitution method to analyze complexity

Instead, let's use "stack of bricks" view of the recursion tree

Set $\alpha = 1/5$ and $\beta = 7/10$



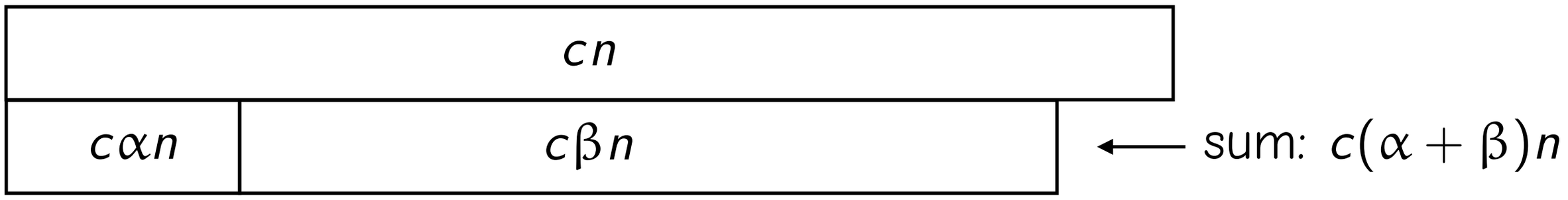
Run-time Analysis

$$T(n) = T(n/5) + T(7n/10) + cn$$

We could use substitution method to analyze complexity

Instead, let's use "stack of bricks" view of the recursion tree

Set $\alpha = 1/5$ and $\beta = 7/10$



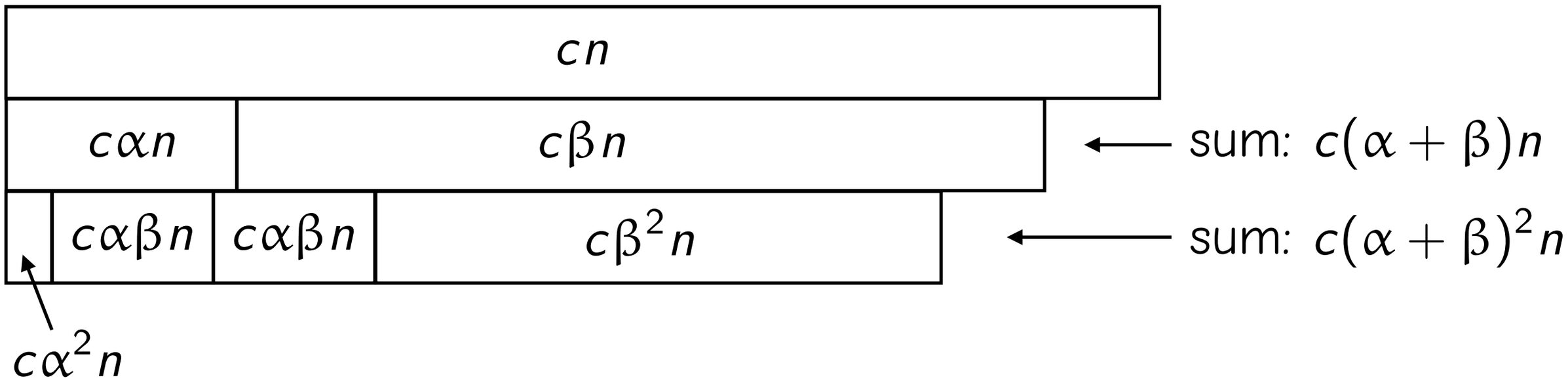
Run-time Analysis

$$T(n) = T(n/5) + T(7n/10) + cn$$

We could use substitution method to analyze complexity

Instead, let's use "stack of bricks" view of the recursion tree

Set $\alpha = 1/5$ and $\beta = 7/10$



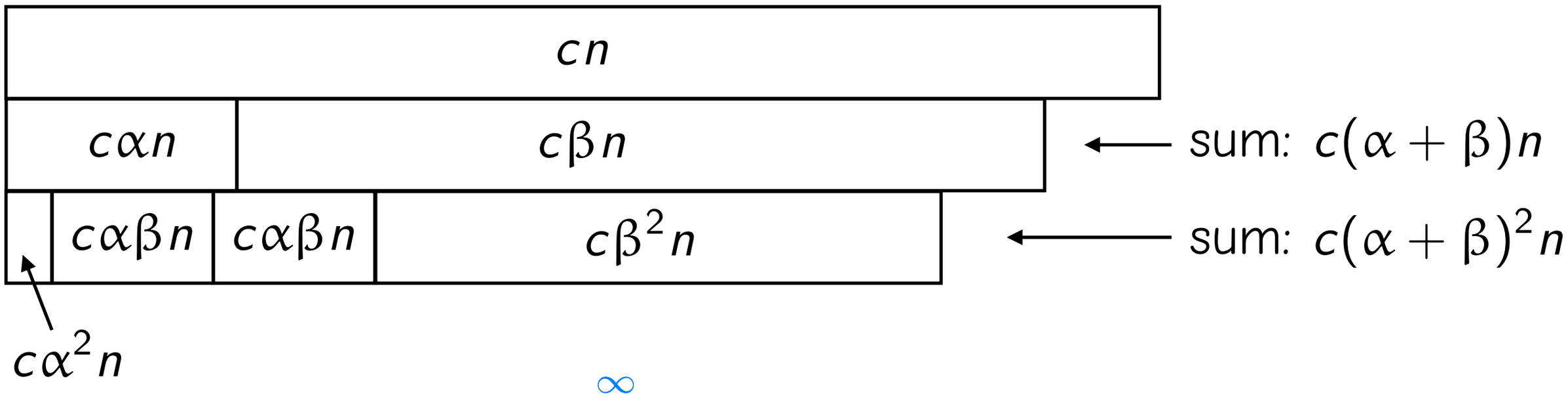
Run-time Analysis

$$T(n) = T(n/5) + T(7n/10) + cn$$

We could use substitution method to analyze complexity

Instead, let's use "stack of bricks" view of the recursion tree

Set $\alpha = 1/5$ and $\beta = 7/10$



$$T(n) \leq O(1) + cn \sum_{j=0}^{\infty} (\alpha + \beta)^j = \frac{cn}{1 - (\alpha + \beta)} = 10cn = O(n)$$

Upper bound on runtime of Quickselect with median of medians pivot

Theorem
Quickselect(A, k) using the median of medians pivot returns the k^{th} order statistic in time at most $O(n)$.

A lower bound

- Suppose Bob tells Alice he has an algorithm that can select the k^{th} order statistic in sublinear time.
- Alice is dubious that Bob's algorithm is correct, because a sublinear algorithm cannot look at all n elements.
- Alice cooks up a length-n sequence of n distinct integers and observes which value Bob's algorithm does not look at.
- She then changes that value so that it becomes the k^{th} order statistic, rendering Bob's algorithm wrong on this adjusted input.

Selecting the k^{th} order statistic takes time $\Omega(n)$

A lower bound

- Example: n^{th} order statistic (the maximum).
  - Alice observes that on her current input, Bob's algorithm does not look at the first element.
  - Alice adjusts A via A[0] = 1 + max{A[1], A[2], …, A[n]}

Optimality of Quickselect with median of medians

Worst-case runtime for selecting the kth order statistic

Quickselect with the median of medians pivot has worst-case runtime of $O(n)$

The worst-case lower bound for *any* algorithm is $\Omega(n)$

Quickselect with the median of medians pivot has worst-case runtime $\Theta(n)$, and this is optimal

Asymptotic optimality isn't everything

- Quickselect with median of medians pivots is clever and asymptotically optimal in the worst-case

- BUT: In practice, Quickselect with a random pivot can be much faster. Why?
  - Quickselect spends a lot of time computing its pivot

- The constants hidden by the $O(n)$ actually matter.