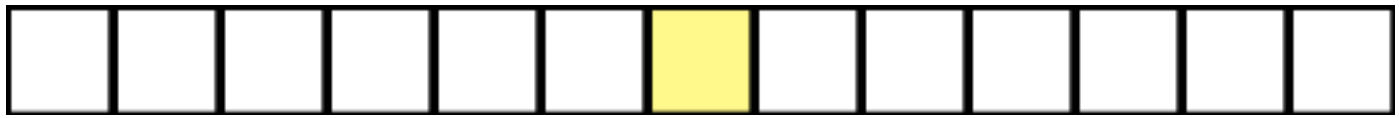
Linear Algorithms for the Selection Problem

Pedro Ribeiro

DCC/FCUP

2018/2019



Selection Problem

Selection Problem

Given an unordered array with n elements, find its k-th smaller item

- One example is to find the **median**
- How quickly can we do this? Can we do it more quickly than by sorting?

Selection Problem vs Sorting

- We talked about an $\Omega(n \log n)$ lower bound for a comparison-based sorting algorithm
- Suppose that now the problem is *"order the input array so that the smallest k items come before the largest $n-k$ items"*.
- For the median: *"order the input array so that the smallest $n/2$ items come before the largest $n/2$ items"*.
- Does the $\Omega(n \log n)$ lower bound still hold for this problem? **No!**
  - It breaks down because any input will have multiple possible permutations as correct answers
  - Ex: For input $[3, 2, 4, 1]$ we could output any of $[1, 2, 3, 4]$, $[2, 1, 3, 4]$, $[1, 2, 4, 3]$, or $[2, 1, 4, 3]$.
  - In fact we are now going to show how to solve this problem in linear time, by solving the selection problem in linear time.

Randomized Selection

- Supose we partition around a specific pivot. What do we know about the position of the k-th element? Do we know in which partition it is?
- **Yes!** Just by looking at the partition sizes, we know where the element we are looking for is!
- Ex: Suppose we are looking for the 67-th smallest element:
  - Let LESS be the set of elements smaller than the pivot, and GREATER the set of elements bigger than the pivot
  - If we get a LESS of size 100, then we just need to find the 67-th element of LESS
  - If we get a LESS of size 40, then we need to find the $67 - 40 - 1 = 26$-th element of GREATER
- We will only recurse on one partition instead of two! What is the **complexity** of something like this?

Randomized Selection

Randomized Selection (QuickSelect)

Given an array of size n and an integer $k \leq n$

1. Pick a pivot p at random from the array
2. Split the array into LESS and GREATER by comparing each element to p as in QuickSort. Let L be the number of elements in $LESS$.
3. a) If $L = k - 1$ then output p

   b) If $L > k - 1$ then output QuickSelect(LESS, k)

   c) If $L < k - 1$ then output QuickSelect(GREATER, $k - L - 1$)

Randomized Selection

Theorem

The expected number of comparisons for QuickSelect is $O(n)$.

Before a formal proof, let's start with some intuition.

If we split a candy bar at random, what would the expected size of the largest piece be? 3/4 of the bar!

If the size of the largest partition was always 3/4 of the array, what would the recurrence be? $T(n) = (n-1) + T(3n/4)$

What would this recurrence solve to? $T(n) \leq 4n$!

$1, 3/4, 3^2/4^2, 3^3/4^3, \ldots$ (geometric series!)

And we have $s = \sum_{i=0}^{\infty} (3/4)^i = 4$ $\quad [s - (3/4)s = 1 \leftrightarrow s = 4]$!

Randomized Selection

Theorem

The expected number of comparisons for QuickSelect is $O(n)$.

In our case, $3/4$ is not the actual size of the largest partition, but only its *expected* value.

What we really want is $\mathbf{E}[T(i)]$ rather than $T(\mathbf{E}[i])$

However, since the answer is linear... the average of $T(i)$ is the same as the T(average of i)!

Let's see this more formally.

Randomized Selection

Theorem

The expected number of comparisons for QuickSelect is $O(n)$.

It takes $n-1$ comparisons to split the array

These pieces are equally likely to have size 0 and $n-1$, 1 and $n-2$, 2 and $n-3$, etc

The piece we choose depends on k but since we are trying to obtain an upper bound, we can imagine recursion always on the larger piece.

$$\begin{aligned} T(n) &\leq (n-1) + \frac{2}{n} \sum_{i=n/2}^{n-1} T(i) \\ &= (n-1) + avg[T(n/2), \ldots, T(n-1)] \end{aligned}$$

Randomized Selection

Theorem

The expected number of comparisons for QuickSelect is $O(n)$.

Now let's use guess and prove. Assuming inductively that $T(i) \leq 4i$ for all $i < n$, then:

$$\begin{aligned} T(n) &\leq (n-1) + avg[4(n/2), 4(n/2+1), \ldots, 4(n-1)] \\ &\leq (n-1) + 4(3n/4) \\ &\leq 4n \end{aligned}$$

□ And we have verified our guess :)

Deterministic Selection

- Randomized selection works well in practice, but can we have a deterministic linear time selection?
- For a long time it was thought to be impossible but in 1972 a deterministic algorithm was developed (by Blum, Floyd, Pratt, Rivest, and Tarjan).
- The main ideia is to try to find deterministically a good pivot, i.e., one that produces a good split.
- Ideal? Median! But finding the median IS the problem...
- We will give ourselves some leeway by allowing the pivot to be any element that is "roughly" in the middle: at least 3/10 of the array below the pivot and at least 3/10 of the array above.

Deterministic Selection

Deterministic Selection (Median of Medians)

Given an array of size n and an integer $k \leq n$

1. Group the array into $n/5$ groups of size 5 and find the median of each group.
2. Recursively, find the true median of the medians. Call this p.
3. Use p as a pivot to split the array into subarrays LESS and GREATER.
4. Recurse on the appropriate piece.

Deterministic Selection

Theorem

Deterministic Selection makes $O(n)$ comparisons to to find the k-th smallest element in an array of size n

How many comparisons for finding the median of 5 elements? A constant number (finding the exact minimum will be a good exercise).

Step 1 takes $O(n)$ time then.

Step 2 takes time $T(n/5)$.

Step 3... We claim that at least $3/10$ of the array is $\leq p$ and at least $3/10$ of the array is $\geq p$. Assuming this is true we get:

$T(n) \leq cn + T(n/5) + T(7n/10)$

Deterministic Selection

Let's prove our claim about the pivot.

First an example. Imagine an array of 15 elements that breaks into 3 groups of 5 like this:

$\{1, 2, 3, 10, 11\}, \{4, 5, 6, 12, 13\}, \{7, 8, 9, 14, 15\}$

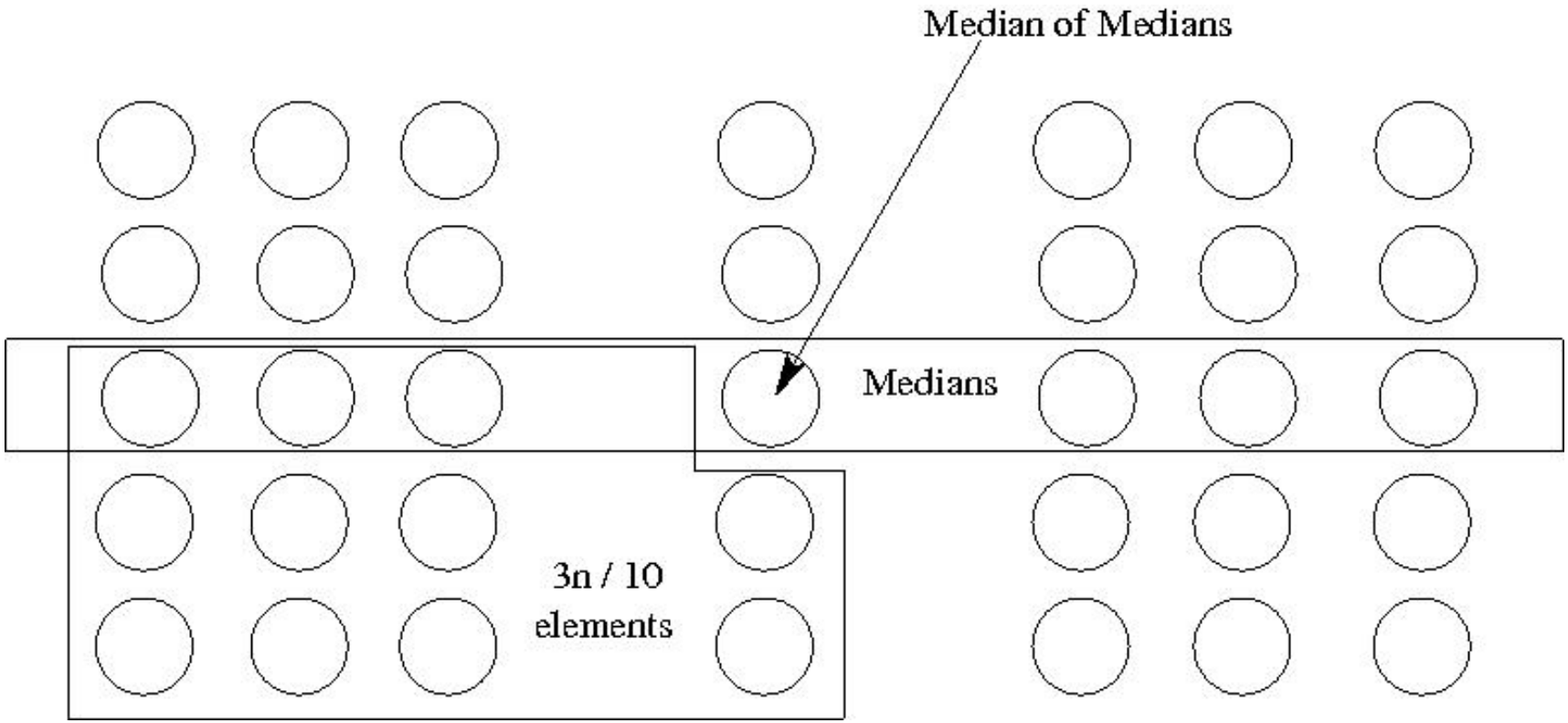
The medians are 3, 6 and 9. The true median of the medians is 6.

There are 5 elements less than p and 9 elements greater.

And in general?

How many groups? $g = n/5$. In at least $\lceil g/2 \rceil$ of them (groups with median $\leq p$) at least 3 out of 5 elements are $\leq p$. Therefore , the total number of elements $\leq p$ is at least $3\lceil g/2 \rceil \geq 3n/10$. Similarly, the number of elements $\geq p$ is also at least $3\lceil g/2 \rceil \geq 3n/10$.

Deterministic Selection



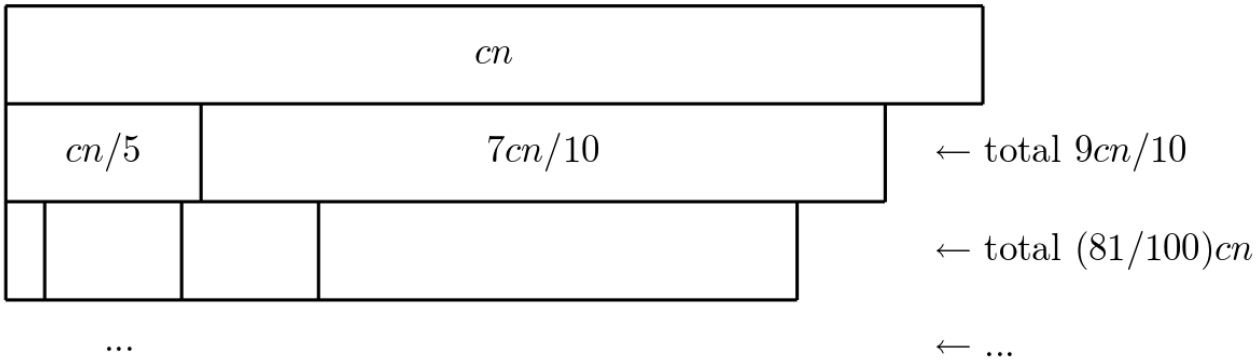
Deterministic Selection

$T(n) \leq cn + T(n/5) + T(7n/10)$

We could again solve by the "guess and prove" method.

Can we however look at the recurrence and see it is linear?

(recurrence where "weight is on top")



Even if we extend this pattern "forever", we will have:

$T(n) \leq cn + T(9n/10)$ whose sum is $cn \sum_{i=0}^{\infty} (9/10)^i = 10cn$

(again a geometric series)

So, $T(n) \leq 10cn$ and we're done! $[T(n) = O(n)]$

An "extension" to the master theorem

Generally, if we have a problem of size n and we can solve it by performing recursive calls on pieces whose total size is at most $(1 - \epsilon)n$ for some constant $\epsilon > 0$ (plus some additional $O(n)$ work), then the total time spent will be just linear in n. This gives us:

"Extension" to Master Theorem

For constants c and $a_1, \ldots, a_k$ such that $a_1 + \ldots + a_k < 1$, the recurrence:

$$T(n) = T(a_1 n) + \ldots + T(a_k n) + cn$$

solves to

$$\mathbf{T(n) = \Theta(n)}$$