CS 161	**Handout for Lecture 4**
Winter 2021	Mon, Jan 25

This handout was written originally by Mary Wootters and has possibly been modified by Moses Charikar and Nima Anari.

Please direct all typos and mistakes to Moses Charikar and Nima Anari (2021).

In this handout, we give a formal proof by induction that the Select algorithm that we saw in class is correct, for any choice of pivot.

1 Selection

Recall that the k-selection problem is to find the k-th smallest number in an array A.

Input: array A of n numbers, and an integer $k \in \{1, \cdots, n\}$.

Output: the k-th smallest number in A.

In class, we saw the following algorithm, where ChoosePivot is a method that chooses a pivot. In class we talked about how to choose the pivot intelligently so that the running time of Select was $O(n)$; but in fact the algorithm is correct (that is, returns the correct answer) for any way of choosing a pivot.

For clarity, we'll assume all elements are distinct from now on, but the idea generalizes easily.

Algorithm 1: Select(A, k)

```
n ← |A|
if k > n or k < 1 then
    throw an error; the input is not allowed
if n = 1 then
    return A[0]
p ← ChoosePivot(A, n)
L ← {A[i] | A[i] < p}
R ← {A[i] | A[i] > p}
if |L| = k − 1 then
    return p
else if |L| > k − 1 then
    return Select(L, k)
else if |L| < k − 1 then
    return Select(R, k − |L| − 1)
```

In plain words, what the algorithm does is as follows. In each call to Select, we use the pivot value p to partition the array into two parts: all elements smaller than the pivot and all elements larger than the pivot, which we denote L and R, respectively. This can be done in $O(n)$ time using the Partition algorithm discussed in class. Then depending on the size of $|L|$, we either return the pivot value p, or we recurse on L, or we recurse on R.

2 Formal proof that Select is correct

Here, we prove formally, by induction, that Select is correct. We will use *strong induction*. That is, our inductive step will assume that the inductive hypothesis holds for *all* n between 1 and $i-1$, and then we'll show that it holds for $n = i$.

Remark 1. You can also do this using regular induction with a slightly more complicated inductive hypothesis; either way is fine.

Inductive Hypothesis (for n). When run on an array A of size n and an integer $k \in \{1, \ldots, n\}$, Select returns the k-th smallest element of A.

Base Case ($n = 1$). When $n = 1$, the requirement $k \in \{1, \ldots, n\}$ means that $k = 1$; that is, Select(A, k) is supposed to return the smallest element of A. This is precisely what the pseudocode above does when $|A| = 1$, so this establishes the Inductive Hypothesis for $n = 1$.

Inductive Step. Let $i \geq 2$, and suppose that the inductive hypothesis holds for all n with $1 \leq n < i$. Our goal is to show that it holds for $n = i$. That is, we would like to show that

> When run on an array A of size i and an integer $k \in \{1, \ldots, i\}$, Select(A, k) returns the k-th smallest element of A.

Informally, we want to show that assuming that Select "works" on smaller arrays, then it "works" on an array of length n.

We do this below:

Suppose that $1 \leq k \leq i$, and that A is an array of length i. There are three cases to consider, depending on $p =$ ChoosePivot(A, i). Notice that in the pseudocode above, p is a value from A, not an index. Let L, R, p be as in the pseudocode above.

- **Case 1.** Suppose that $|L| = k-1$. Then by the definition of L, there are $k-1$ elements of A that are smaller than p, so p must be the k-th smallest. In this case, we return p, which is indeed the k-th smallest.
- **Case 2.** Suppose that $|L| > k-1$. Then there are more than $k-1$ elements of A that are smaller than p, and so in particular the k-th smallest element of A is the same as the k-th smallest element of L. Next we will use the inductive hypothesis for $n = |L|$, which holds since $|L| < i$. Since $1 \leq k \leq |L|$, the inductive hypothesis implies that

Select(L, k) returns the k-th smallest element of L. Thus, by returning this we are also returning the k-th smallest element of A, as desired.

- **Case 3.** Suppose that $|L| < k - 1$. Then there are fewer than $k - 1$ elements that are less than p, which means that the k-th smallest element of A must be greater than p; that is, it shows up in R. Now, the k-th smallest element in A is the same as the $(k - |L| - 1)$-st element in R. To see this, notice that there are $|L| + 1$ elements smaller than the k-th that do *not* show up in R. Thus there are $k - (|L| + 1) = k - |L| - 1$ elements in R that are smaller than or equal to the k-th element. Now we want to apply the inductive hypothesis for $n = |R|$, which we can do since $|R| < j$. Notice that we have $1 \leq k - |L| - 1 \leq |R|$; the first inequality holds because $k > |L| + 1$ by the definition of Case 3, and the second inequality holds because it is the same as $k \leq |L| + |R| + 1 = n$, which is true by assumption. Thus, the inductive hypothesis implies that Select$(R, k - |L| - 1)$ returns the $(k - |L| - 1)$-st element of R. Thus, by returning this we are also returning the k-th smallest element of A, as desired.

Thus, in each of the three cases, Select(A, k) returns the k-th smallest element of A. This establishes the inductive hypothesis for $n = i$.

Conclusion. By induction, the inductive hypothesis holds for *all* $n \geq 1$. Thus, we conclude that Select(A, k) returns the k-th smallest element of A on any array A, provided that $k \in \{1, \ldots, |A|\}$. That is, Select is correct, which is what we wanted to show.