Tirgul 4

Randomized & Average Complexity

Selection

- The selection problem is defined as follows:
  - Input: An array $A[1..n]$ of elements in an arbitrary (random) order and an index k
  - Output: The k^{th} smallest element of A
- When k==**1**, we are looking for the ***minimum***
- When k==n, we are looking for the ***maximum***
- When k==n/**2** we are looking for the ***median***

Quick Select

- The algorithm idea is very simple:

1. We choose a pivot and split the array around it (like in quicksort)
2. The desired element is now on one side of the pivot (we know, which side) and we recursively apply the procedure on that side

Quick Select

- QuickSelect(A,p,r,i)
  1. if $p=r$
  2. return $A[p]$
  3. $q\leftarrow$RandomizedPartition(A,p,r)
  4. $k\leftarrow q-p+1$
  5. if $i\leq k$
  6. then return QuickSelect(A,p,q,i)
  7. else return QuickSelect($A,q+1,r,i$)

Quick Select

- What is the worst case running time of the algorithm?
- In the worst case, the partition always partitions the input to n-1 and 1, and the k^{th} element is always in the larger partition
- The recurrence equation in that case is:
- $T(n) = T(n\text{-}1) + n = O(n^2)$
- Since the algorithm is random (we have a random partition), no particular input will yield that running time

Quick Select

- Randomized complexity analysis:

$$T(n)=\frac{1}{n}\sum_{k=1}^{n-1}T(k)+n$$

Let us assume that $T(k)<ck$ for some k

$$T(n)=\frac{1}{n}\sum_{k=1}^{n-1}ck+n=\frac{c}{n}\sum_{k=1}^{n-1}k+n=\frac{c}{n}\frac{n(n-1)}{2}+n$$

$$=\frac{c}{2}(n-1)+n\leq cn \qquad \text{For } c>3$$

Insertion Sort – Average complexity

- We have seen that the worst case complexity of insertion sort is $O(n^2)$, but what is its average complexity?
- To do that we need to average over all the possible insertion in each step of the algorithm
- We define E_i as the average number of comparisons needed for inserting of the i^{th} element
- When inserting the i^{th} element, we already have i-1 sorted elements, so on an average we get

$$E_i = \frac{1}{i}\sum_{k=1}^{i} k = \frac{i-1}{2}$$

Insertion Sort – Average complexity

- The total complexity of the algorithm is the combined complexity of all the insertions, therefore:

$$T(n) = \sum_{i=1}^{n} E_i = \sum_{i=1}^{n} \frac{i-1}{2} = \frac{1}{2}\left(\sum_{i=1}^{n} i - \sum_{i=1}^{n} 1\right) = \frac{n(n-1)}{4} - \frac{n}{2} = O(n^2)$$

- So the average complexity is polynomial as well

Insertion Sort – Average complexity

- The only thing we need to justify is why the i^{th} element has an equal probability to be placed in each of the i possible locations
- But since the elements are a random permutation, the first i elements are also a random permutation