COMP3711: Design and Analysis of Algorithms

Tutorial 4

HKUST

Question 1

Recall the randomized selection algorithm. We pick a pivot randomly, divide the array into 3 segments: left, pivot, and right, and then either stops immediately or recursively solve the problem in the left or the right part of the array. As in the analysis of quicksort, we denote by $z_1, \ldots, z_n$ the elements in sorted order. Suppose the randomized selection algorithm is given the task of finding the k-th smallest element (namely, z_k). What is the probability that z_i and z_j $(i < j)$ are ever compared by the algorithm? [Hint: consider the following 3 cases separately: $k < i, i \leq k \leq j, k > j$.] Then use the indicator variable technique to show that the expected running time of the randomized selection algorithm is $O(n)$.

Question 2

The analysis of the expected running time of randomized quicksort in lecture note assumes that all element values are distinct. In this problem, we examine what happens when they are not.

(a) Suppose that all element values are equal. What would be randomized quicksort's running time in this case?

(b) The PARTITION procedure returns an index q such that each element of $A[p...q-1]$ is less than or equal to $A[q]$ and each element of $A[q+1...r]$ is greater than $A[q]$. Modify the PARTITION procedure to produce a procedure PARTITION$'(A, p, r)$, which permutes the elements of $A[p...r]$ and returns two indices q and t, where $p \le q \le t \le r$, such that

- all elements of $A[q...t]$ are equal,
- each element of $A[p...q-1]$ is less than $A[q]$, and
- each element of $A[t+1...r]$ is greater than $A[q]$

Like PARTITION, your PARTITION$'$ procedure should take $\Theta(r-p)$ time.

(c) Modify the QUICKSORT procedure to produce QUICKSORT′(A, p, r) that calls PARITITION′ and recurses only on partitions of elements not known to be equal to each other.

(d) Using QUICKSORT′, how would you adjust the analysis in lecture note to avoid the assumption that all elements are distinct?