### CS 5633: Analysis of Algorithms

### Homework 4

1. a) **Array with n equal keys:**

If we consider the last element as a pivot, each time the partition function will return the last element’s index as the partition point. So, the recurrence relation will be:

T(n) = T(0) + T(n-1)+Ө(n).

Steps will be n -> n-1 -> n-2->.......->1

This will produce the worst-case scenario and the runtime will be Ө(n2).

In the case of randomized quicksort, the pivot is chosen randomly. But the same scenario will happen here too as each element is the same. The runtime will be Ө(n2).

b)

If we change A[j] ≤ x to A[j] < x in the pseudocode for partition and select the last element as pivot, the partition function will return the first elements’ index as the partition point. Again the recurrence relation will be:

T(n) = T(0) + T(n-1)+Ө(n). Its runtime will be Ө(n2).

c)

In deterministic quicksort with just two distinct keys, after the first partition, there will be two subproblems in the average case. If one subproblem has k elements then the other will have n-k-1 elements. After this step, all next partitions will produce one subproblem with 1 lesser element from the two children. So from no a) the runtime will be Ө(n2).

d)

Partition(A, p, r)

pivot = A[r]

i = p

while p <= r

if A[p] < pivot

exchange A[i] with A[p]

i = i+1

p = p+1

else if A[p] == pivot

p = p+1

else if A[p] > pivot

exchange A[p] with A[r]

r = r-1

return (i-1, p) // Equal elements of pivots are from index i-1 and to index p

e)

For this, consider an example 10, 10, 10, 30, 30, 10, 30

Here only 2 elements are distinct. Considering the last element as the pivot, after the first partition second level will have only 10, 10, 10, 10 and after this level, it is sorted.

Steps = n -> n-k where k is the number of equal elements as the pivot.

And the running time = n+n-k = 2n-k

So, the worst-case runtime = O(n)

f)

In an array of n elements with d distinct values, consider the last element as the pivot, and the array is sorted in increasing order. So, after each partition, only the pivot is sorted that is only one element is sorted. In one partition n-d elements will be sorted. This could be the worst case.

n -> n-1 -> n-2 ->....->n-k-(n-d)-> …… 1 where k is a constant

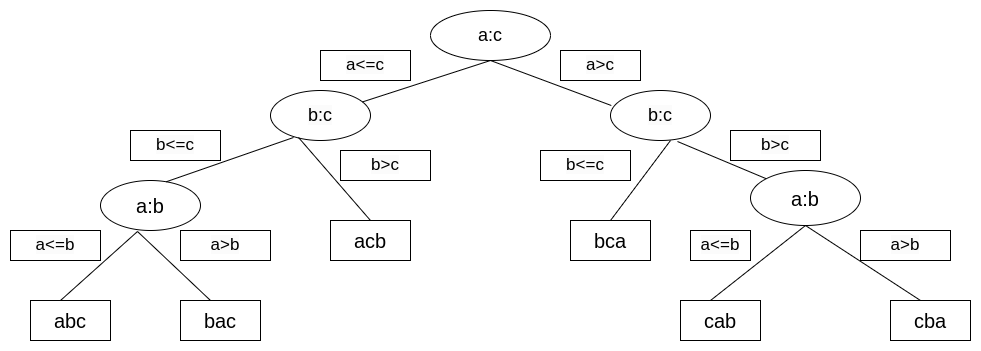
The running time = n+n-1+n-2+....+n-k-(n-d)+1

For d distinct elements running time = dn-e where e is a constant

So, the worst-case runtime = O(dn)

2.

Let the 3 inputs be a,b,c



3.

**Counting Sort:**

Let’s assume that there are two equal elements in an array whose indices are k1 and k2. In the sorted array, one will be in A[i1] and another will be in A[i2] where A is the sorted array. The counting sort works in reverse order during the final insertion in the sorted list from the original list. So, k2 will be inserted in A[i2] and k1 will be in A[i1]. As the two elements preserve their order, the Counting sort is stable.

**Insertion Sort:**

Insertion sort is a stable sorting algorithm. During the sorting process, we compare each element with all the elements to its left and only swap two elements if the right element is smaller than the left one. Therefore the ordering of two equal elements will always be preserved. As the two elements preserve their order, the Insertion sort is stable.

**Merge Sort:**

Merge sort is a stable sorting algorithm. In merge sort, two elements will swap if the right element is smaller than the left one. In this way, two equal elements won’t swap their position and their order will be preserved always. As the two elements preserve their order, the Merge sort is stable.

**Quick Sort:**

Quick sort is an unstable sorting algorithm. In the case of the quick sort, we swap elements according to the pivot’s position. We don’t consider their original position. Hence original order may be hampered according to the pivot chosen. As two equal elements can’t preserve their order, Quick sort is unstable.

For example, [10, 10, 20, 30, 10] is unstable. Considering the first element as the pivot after the partition first 10 will go after the second 10. So, this is unstable.