Lab 5 Assignment [Ushna Ijaz- 2019-CE-39]

May 5, 2021

Question 1

Suppose we are given an array A of length n with the promise that there exists a majority element (i.e. an element that appears ¿ n / 2 times). Additionally, we are only allowed to check whether two elements are equal (no ¿ or ¡ comparisons). Design an $O(nlog(n))$ algorithm to find the majority element, using divide and conquer. Informally, explain the correctness and runtime of your algorithm. [We are expecting actual code, an English description of the main idea of the algorithm, as well as an informal explanation of correctness and runtime]

Solution

We split the array A into 2 subarrays arr1 and arr2 of half the size. We choose the majority element of arr1 and arr2. The recurrence:

$$T(n) = 2T(n/2) + O(n)$$

We then use the divide-and-conquer method. Basically we have to pair up the elements so we get $n/2$. If the two elements in a pair are difference, it is discarded. If one of the parts has a majority element we count the number of times it is repeated. And if both parts have the same majority element, it is automatically the majority element for array A. My code counts the occurrences for the majority elements in each half of the list.

Question 2

For the sorted arrays, we count while we compare the elements in the given arrays. But first we need to merge both the arrays. For that we took help from the Merge Sort algorithm. Its basically finds the medians of the two sorted arrays and second step is that it compares them.When count becomes n we got our answer. For the Unsorted arrays, the procedure is almost the same but the only difference made is that I sorted the arrays.

Question 3 Table

Time Comparison			
Array Size	Insertion Sort	Merge Sort	Quick Sort
10	0.0	0.0	0.0
100	0.0	0.0	0.007989883
1000	0.124124526	0.0100557804	0.010042428
10000	15.5295608	0.08614063	0.56423211
100000	1788.871452	2.18674635	29.0143427