Heaven’s Light is Our Guide

RAJSHAHI UNIVERSITY OF ENGINEERING AND TECHNOLOGY

Dept. of Computer Science & Engineering

Course Code : *CSE 2202*
Course Title : *Sessional based on CSE 2201*
Experiment NO : *02*
Experiment Name : *Complexity analysis of Finding Maximum and Minimum Value and various sorting algorithms [Quick Sort and Merge Sort].*

Date of Experiment : *17/10/2022*
Date of Submission : *24/10/2022*

MACHINE CONFIGURATION:
Intel(R) Core(TM) i5-10210U CPU
@ 1.60GHz 2.11 GHz, 20.0 GB of RAM 1 TB of Hard disk

Submitted By

Name: Md. Golam All Gaffar Tasin
Roll : 1903114
Section: B
Year: 2nd Year(Even)
Dept. of CSE

Submitted to

Rizoan Toufiq
Assistant Professor, Dept. of CSE
Rajshahi University Of Engineering And Technology.

Name of the Experiment: Complexity analysis of Finding Maximum and Minimum Value and various sorting algorithms [Quick Sort and Merge Sort].

Objective:

Getting knowledge about Finding Maximum and Minimum algorithm and various sorting algorithm like quick sort and merge sort, also getting knowledge about the complexity analysis of these algorithms.

Theory:

Quick sort, sometime called partition exchange algorithm, is an efficient sorting algorithm serving as a systematic method for placing the elements of random access file or any in order. Merge sort is a sorting technique based on divide and conquer technique. Merge sort first divide the array into equal halves and then combines them in a sorted manner. Here both type of sorting are example of divide and conquer technique. Time complexity-

Quicksort	Merge sort
Worst case: $\theta(n^2)$	Worst case: $O(n\log n)$
Best case : $\theta(n\log n)$	Best case: $O(n\log n)$
Avg case: $\theta(n\log n)$	Avg case: $O(n\log n)$

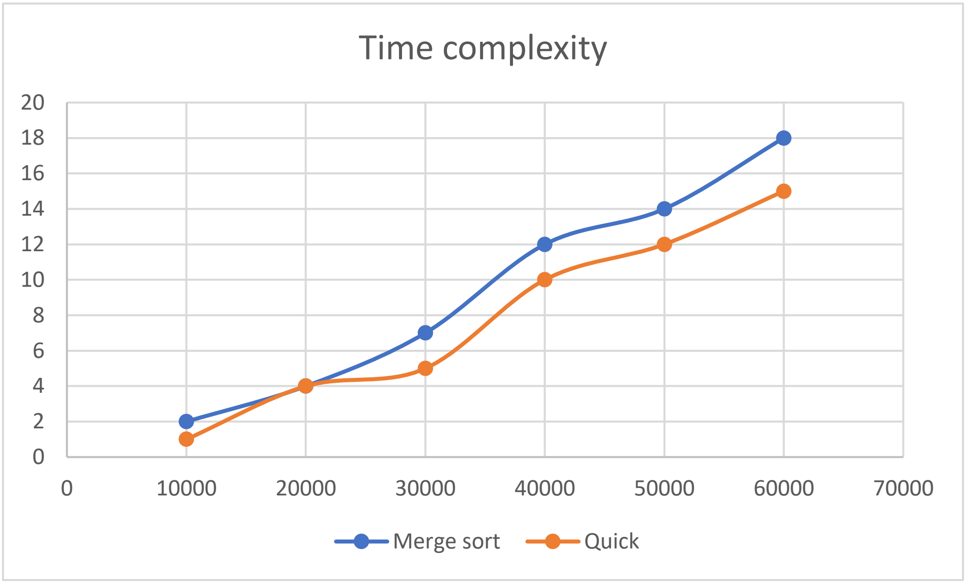
Procedure:

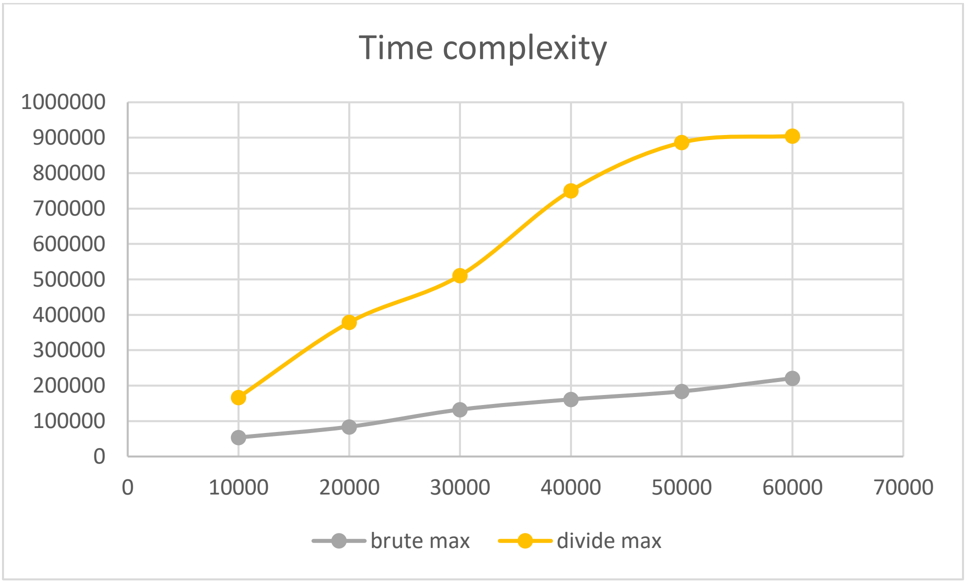
```
MERGE-SORT(A, p, r)
1. if p<r
2. q=⌊(p+r)/2⌋
3. MERGE-SORT(A,p,q)
4. MERGE-SORT(A,q+1,r)
5. MERGE(A,p,q,r)
```

```
Quick Sort(A,p,r)
1. if p<r
2. q = PARTITION(A, p, r)
3. QUICKSORT(A,p,q-1)
4. QUICKSORT(A,q+1,r)
```

Data Table:

NO of input	10000	20000	30000	40000	50000	60000
Quick sort(milli s)	1	4	5	10	12	15
Merge sort(milli s)	2	4	7	12	14	18
Brute maximum(ns	53781	83991	132513	161268	183889	221052
Divide maximum(n	167087	378652	510948	750415	886264	905012





Discussion:

From the experiment we can see that merge sort slightly perform better than quicksort in average case. In the best case both sorting algorithm has the same time complexity but in the worst case quicksort has time complexity of O(n^2) that is bigger than O(nlogn) for merge sort. So merge sort can perform better in all case.

From the graph we can see that for every size of input finding maximum is taking more time than brute maximum finding. The reason behind is that calling of recursive function takes some extra time. Putting in stack calling the function all takes some extra time. That’s why the time is more than brute max. When the array is sorted then the time complexity of divide and conquer maximum is O(logn). But we didn’t use a sorted array.