OFFLINE 7

Merge Sort and Quick Sort

Name: Fardin Anam Aungon

Student ID: 1805087

Department: CSE

Section: B1

Submitted on: 11th June, 2021

Procedure: For each of every order and every array size, ten random arrays are generated and the times required to sort those arrays using both the merge sort and quick sort are calculated. Then the average time of the ten different values of each sorting algorithm is calculated separately. Hence, I have got the following data table.

Table: Time complexity data from code

Input order	n =	10	100	1000	10000	100000	1000000
	Sorting Algorithm						
Ascending	Merge	3.53E-06	7.35E-06	9.45E-05	8.56E-04	0.005218	0.073522
Ascending	Quick	3.25E-06	1.75E-05	3.89E-04	0.047314	4.516625	461.1316
Descending	Merge	4.65E-06	7.92E-06	7.42E-05	3.90E-04	0.004236	0.057581
Descending	Quick	2.05E-06	1.54E-05	3.62E-04	0.0408	3.628709	361.1963
Random	Merge	4.40E-05	2.26E-05	1.22E-04	0.001334	0.012367	0.105432
Random	Quick	1.83E-06	1.28E-05	5.47E-05	5.83E-04	0.006076	0.071656

Graphs: Graph of array size vs time taken to sort the array using both merge sort and quick sort are given below.

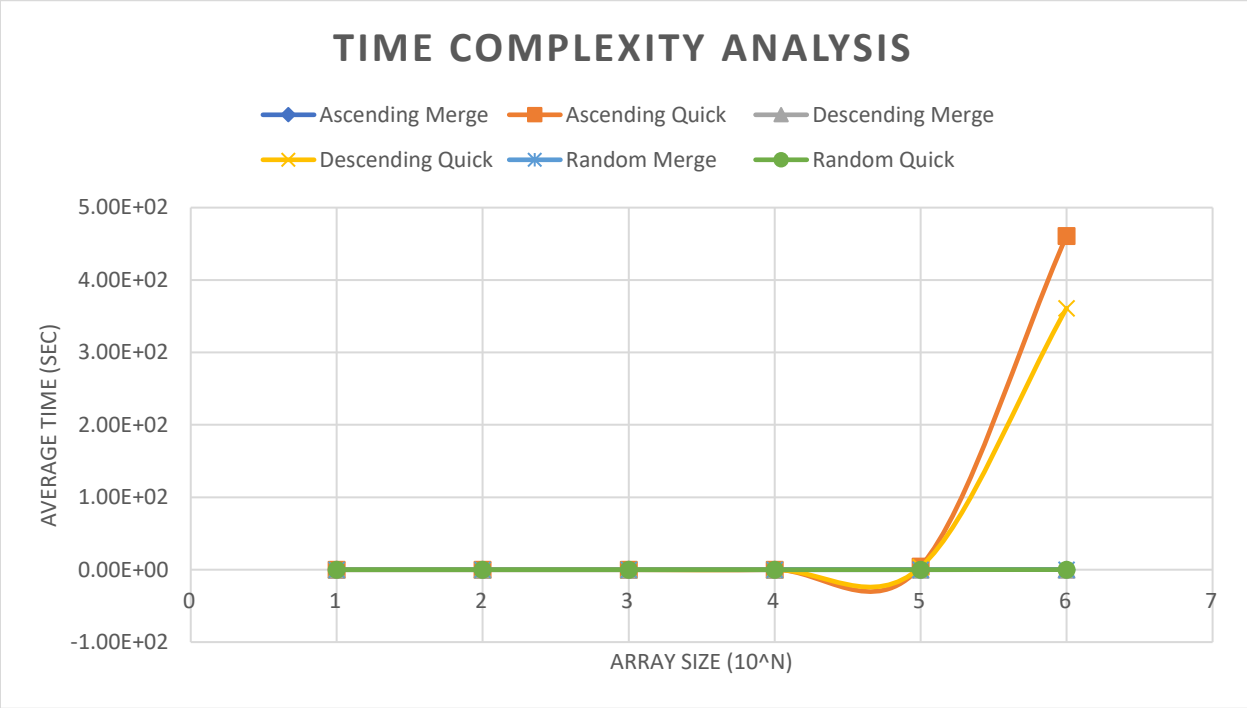

Fig 1: Sorting algorithms applied on randomly generated arrays containing values in all order

Since the time to sort an already sorted array by quick sort takes a very large time compared to other orders, the curves of lower values have become a straight line. So, for a better understanding of the curve, I have shown the curves of different orders separately.

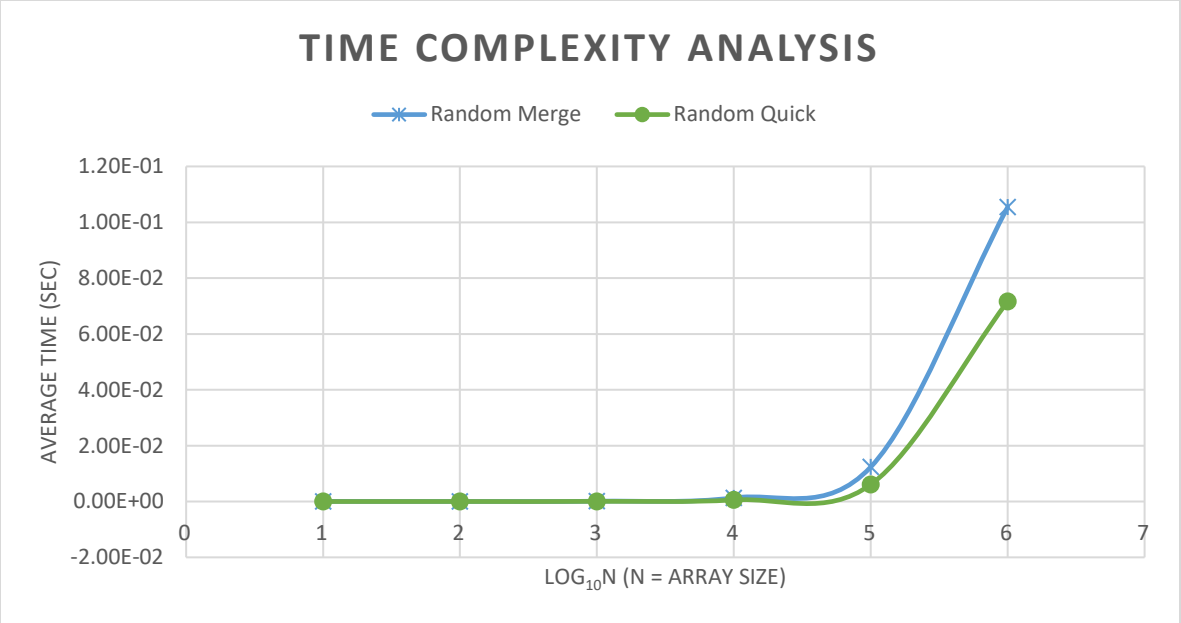

Fig 2: Sorting algorithms applied on randomly generated array containing values in random order

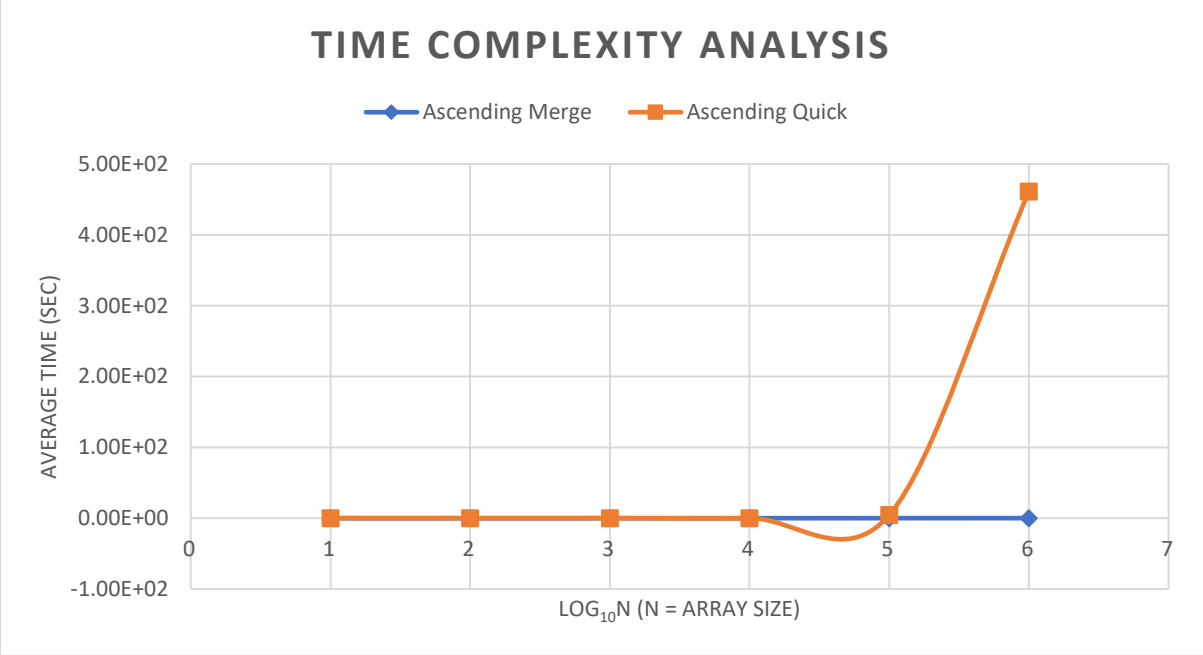

Fig 3: Sorting algorithms applied on randomly generated array containing values in ascending order

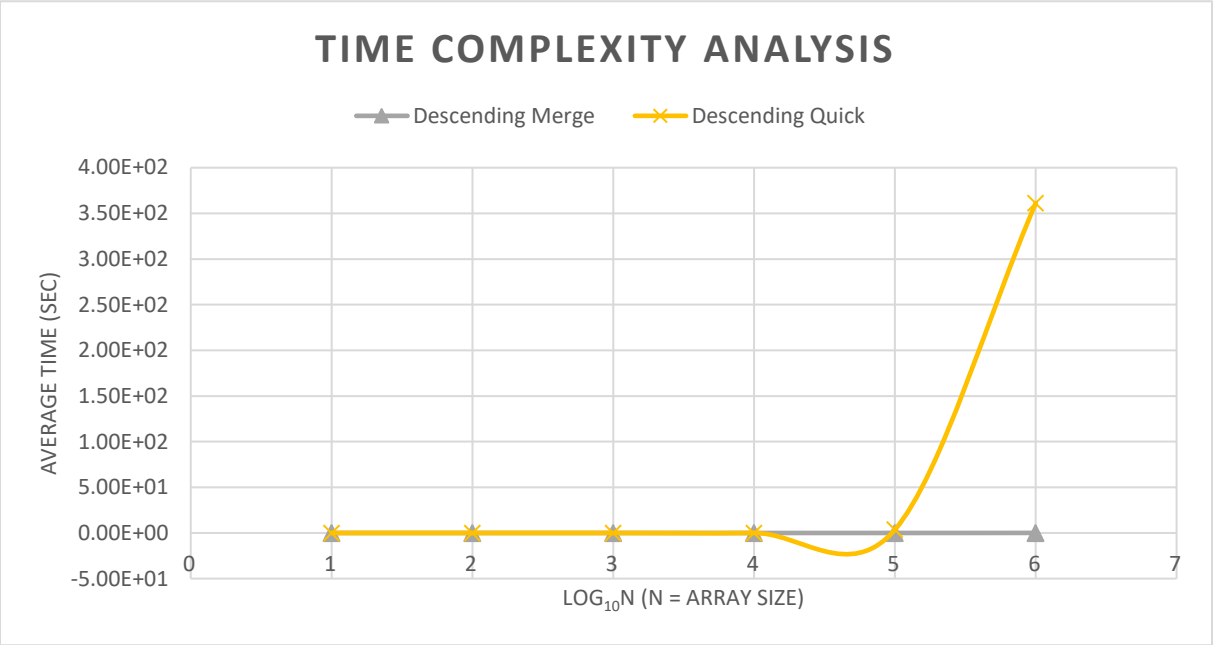

Fig 4: Sorting algorithms applied on randomly generated array containing values in descending order