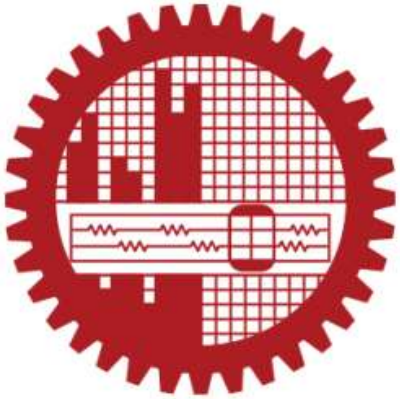


Bangladesh University of Engineering and Technology

Course No: CSE 204

Name: Md. Shahrukh Islam

Student Id: 1805098

Department: CSE

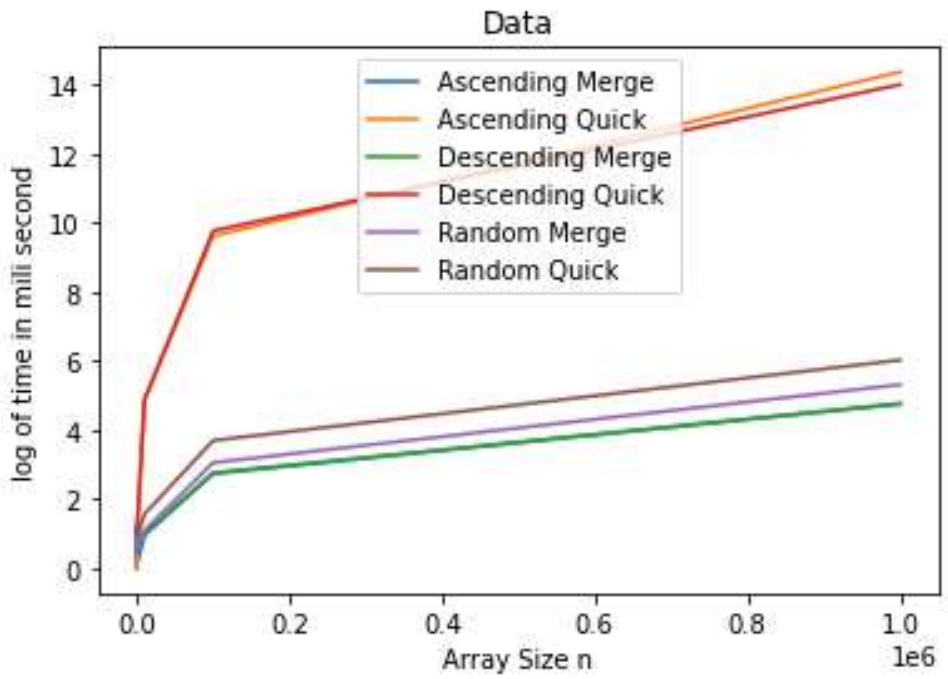
Section: B2

Date: 11/06/2021

Table: Average time for sorting n integers in different input order

Input Order	n = / Sorting Algorithm	10	100	1000	10000	100000	1000000
Ascending	Merge	1	2.2	1.2	2.6	15.4	117
	Quick	1	3.2	3.4	129.6	14916.6	1717980
Descending	Merge	2.6	1.6	2.4	2.8	15.8	116
	Quick	2.8	2.8	4.2	130.6	17346.2	1194900
Random	Merge	1.8	2.6	1.8	3	21.2	202.8
	Quick	2.8	3.4	2.6	4.8	40.2	413.2

Graph Plot:



Machine Configuration

Processor : Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80 GHz

Ram : 12.0 GB

OS: Windows 10

Complexity Analysis

Merge Sort:

Complexity for all three order:

Analysing all the three order for merge sort, we can say that each of the three cases gives almost same amount of time asymptotically. That's why from the above plotting of average runtime vs input size, it shows the almost same growth rate at each curve for all the three orders. We can clearly observe that increasing the input size for 10 times, showing the average running time increasing more than 10 times but not increasing polynomially. So average running time is not increasing linearly with input size. Rather the run time is in between linear growth rate and polynomial growth rate. So the runtime can be f(n) = nlogn asymptotically. So, the complexity of merge sort for all of these three orders is asymptotically, T(n) = O(nlogn).

Quick Sort:

Complexity for Random order:

We can clearly observe increasing the input size for 10 times ,showing the average running time increasing more than 10 times but not increasing polynomially. So average running time is not increasing linearly with input size. So,the run time is in between linear growth rate and polynomial growth rate. So the runtime can bounded by f(n) = nlogn asymptotically. So,the complexity of quick sort for random order is asymptotically, T(n) = O(nlogn).

Complexity for Ascending order:

From the increasing rate of running time with the input size, we can observe that when input size is very large, the running time is increasing at the rate of almost square of the increment of input size. That's why the running time in this case can be bounded by $f(n)=k*n^2$ (k=any constant). For ascending order, the complexity of quick sort is asymptotically, $T(n) = O(n^2)$

Complexity for Descending order:

From the increasing rate of running time with the input size, we can observe that when input size is very large, the running time is increasing at the rate of almost square of the increment of input size. That's why the running time in this case can be bounded by $f(n)=k*n^2$ (k=any constant). For descending order, the complexity of quick sort is asymptotically, $T(n) = O(n^2)$