A REPORT ON

COMPARISON BETWEEN QUICK SORT AND MERGE SORT

SUBMITTED BY: KAWSHIK KUMAR PAUL

STUDENT ID: 1705043

SECTION: A2

DEPARTMENT: CSE

COURSE NO: CSE-204

Machine Configuration

Processor: Intel Core i3 6th Gen (Up to 2.2GHz)

Ram: 4GB DDR3

Operating system: Windows 7

Data And Complexity Analysis

Merge Sort:

Best Case: O(n*logn)

Average Case: O(n*logn)

Worst case: O(n*logn)

That does mean that the time complexity for merge sort is (n*logn). As a result, the time to sort an array doesn't really depend on what kind of array we give as a input.

Quick Sort:

Best Case: O(n*logn)

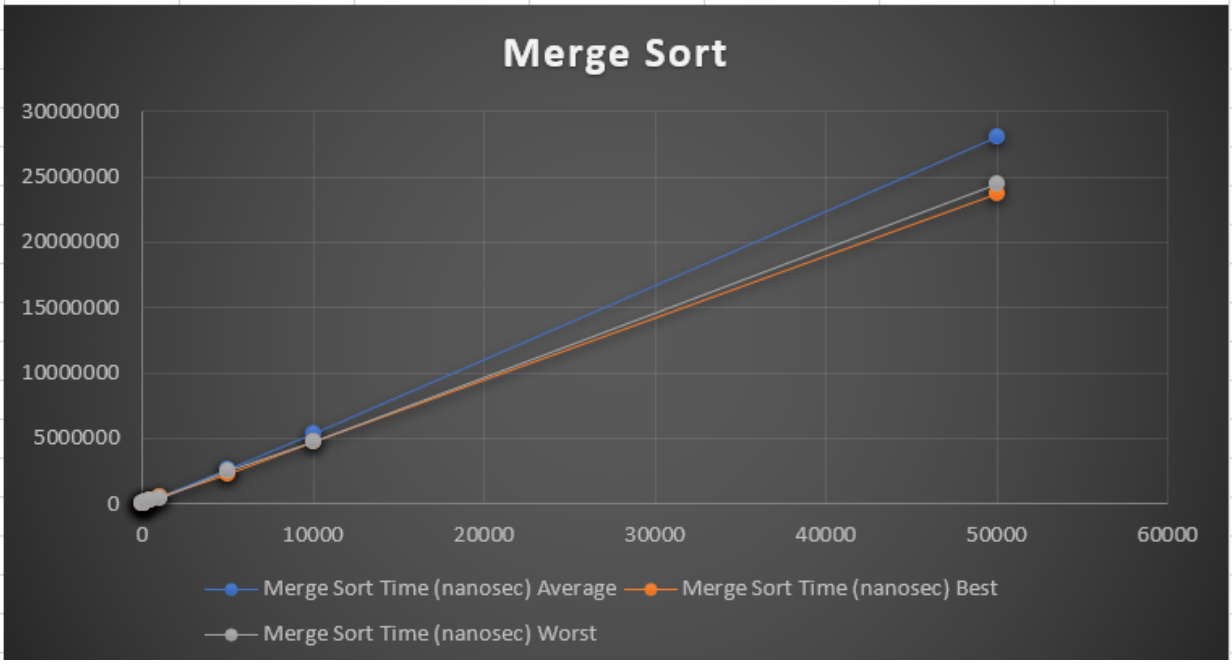
Average Case: O(n*logn)

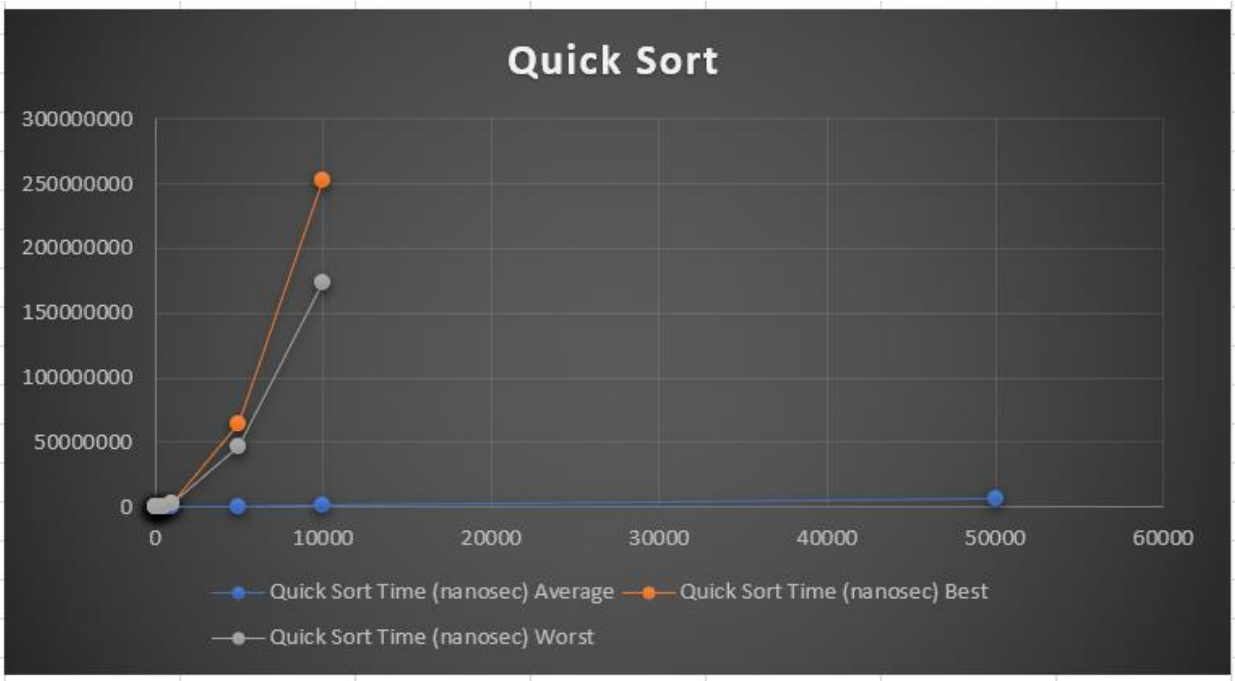
Worst Case: O(n^2)

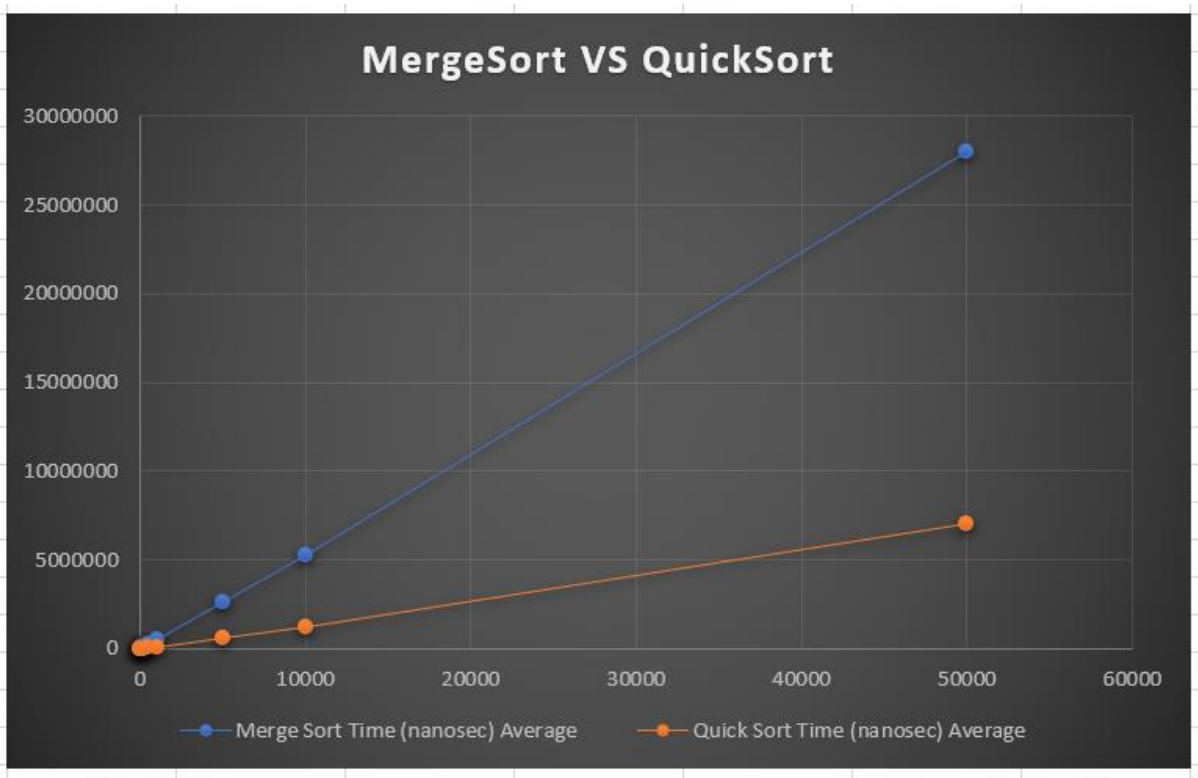
The best case of merge sort is the worst case for Quick sort. As we always take the last element of the array as the pivot element, so if the array is sorted or reversely sorted, the pivot element always stays as the last or the first element of the array. That increases the time complexity of the sorting algorithm as the pivot element has to be compared with all the elements left or right to it.

Data Table:

	Merge Sort Time (nanosec)			Quick Sort Time (nanosec)		
Array Size (n)	Average	Best	Worst	Average	Best	Worst
10	3803	3702.3	3602.3	200.6	300.3	350.2
50	21015	20264.5	30524.5	1504	800.5	5507
100	43032	41532	41029	4504	28022	21016
200	88058	86064	96076	8008	108070	84064
500	232665	220165	220145	30035	642945	500380
1000	495360	450290	440370	80050	2556890	2031520
5000	2602150	2301750	2451500	600350	64469700	46431950
10000	5303600	4703300	4754000	1200800	253276400	174066200
50000	28021000	23766000	24516500	7003500		






MergeSort VS QuickSort
30000000
25000000
20000000
15000000
10000000
5000000
0
0
10000
20000
30000
40000
50000
60000
Merge Sort Time (nanosec) Average
Quick Sort Time (nanosec) Average