Vaishakhi Kulkarni Net Id:vpk140230

PROJECT REPORT

Problem Statement:

Comparison of randomized vs deterministic Quick sort:

(a) Randomized version of partition

(b) Median-of-three partitioning method.

Compare the algorithms on large arrays (n > 10 million).

Do not use randomly generated input data. Try inputs that are sorted, reverse sorted, or almost sorted.

Results:

Following table shows the performance of deterministic and randomized Quick Sort.

Part 1:

Size: 10000000

Elements are sorted

Quick Sort	Deterministic	Randomized
1	791	1586
2	1015	1605
3	698	1451
4	857	1518

Runtime complexity for randomized quicksort is more compared to Deterministic Quick sort.

Part 2:

Size: 10000000

Elements are in reverse order

Quick Sort	Deterministic	Randomized
1	1342	1458
2	1287	1607
3	1202	1781
4	1266	1396

Runtime complexity for randomized quicksort is more compared to Deterministic Quick sort.

Comparison of runtime complexity of randomized vs deterministic Quick sort

Part 3:

Size: 50

Elements are sorted.

Quick Sort	Deterministic	Randomized
1	1	0

When the size is small the randomized quicksort has very less runtime complexity compared to Deterministic to perform quicksort.

Note: All the reading are calculated after taking average of 5 outputs.

Conclusion: Running time complexity of Deterministic Quick Sort is less compared to Randomized Quick Sort when the size of array is very huge. If array size is small randomized Quick Sort have less runtime complexity.

References:

Class notes - Balaji

Introduction of Algorithms - Thomas Cormen, Charles Leiserson, Ronald Rivest , Clifford Stein

http://www.java2s.com/Tutorial/Java/0140__Collections/Quicksortwithmedianofthreepartitioning.htm

http://www.java-tips.org/java-se-tips/java.lang/quick-sort-implementation-with-median-of-three-partitioning-and-cutoff-for-small-a.html