DATA STRUCTURE & INTRO TO ALGORITHMS

COMPLEXITY ANALYSIS FOR SORTING ALGORITHMS

Student Name: Sewar Khalifeh Student No.:20160147 Dr. Khalid Mahmoud

November,4th,2018

Introduction

A sorting algorithm is an algorithm made up of a series of instructions that takes an array as input, performs specified operations on the array, sometimes called a list, and outputs a sorted array [1]. In this report, I aim to analyze each of Insertion, Selection & Bubble sorting algorithms, by comparing the Big O notation of each one & the execution time for each while changing the size of the inputs.

Method

To begin with, The code of each algorithm was written in C++, in the main program there was an option to print out data retrieved randomly,then the sorting methods will be applied and the arrays will be sorted “while showing the execution time”, I tried a small number of inputs “20 for example” then printed it to check that the algorithms are working with no bugs.

Furthermore, I stopped the printing method to work on larger inputs, I started with 10000 array size, then 100000 & finally 1000000. The execution time for each sorting algorithm was shown on the screen so I can start the analysis & the comparison between each algorithm. Finally, I took screenshots of the results.

Results

From the tables below, we can see how each algorithm varies in the execution time, as you can see the insertion algorithm seems to be the fastest among them,”I’ll discuss the Big O notations in the comparison section below” ,Bubble sort is the slowest, and between the selection & the improved selection sort, the improved one is faster and the growth rate must be slower,”Sorted Data” generates the best case where we can see huge differences between the insertion & bubble sort , selection2 is better on the average case from the selection sort,” Reversed Data” gives the worst case, “Random Data” generates random inputs in the array(Average case), ”nearly sorted data” is somehow more of a best case for each algorithm.

10000 inputs analysis:

Exec.Time(ms)/Algorithm	Insertion	Selection	Bubble	Selection2
Sorted Data	0	246	230	122
Reversed Data	133	264	8541	115
Random Data	67	300	4364	121
Nearly Sorted Data	0	280	239	132

100000 inputs analysis:

Exec.Time(ms)/Algorithm	Insertion	Selection	Bubble	Selection2
Sorted Data	0	26288	22792	11991
Reversed Data	12765	25282	781307	12001
Random Data	6389	29699	424204	11929
Nearly Sorted Data	0	30004	22942	11791

1000000 inputs analysis:

Exec.Time(ms)/Algorithm	Insertion	Selection	Bubble	Selection2
Sorted Data	2	2.23782e+06	2.60445e+06	1.38947e+06
Reversed Data	1.29673e+06	2.27116e+06	8.05631e+07	1.27408e+06
Random Data	654789	2.20077e+06	4.48781e+07	1.29424e+06
Nearly Sorted Data	3	2.06634e+06	2.30077e+06	1.3333e+06

Comparison Between Different Algorithms:

To choose a sorting algorithm for a particular problem, consider the running time, space complexity:

BigO/Algorithm	Insertion	Selection	Bubble	Selection2
Best Case	n	n^2	n	n^2
Worst Case	n^2	n^2	n^2	n^2
Average Case	n^2	n^2	n^2	n^2
Space Complexity	1	1	1	1

Note that Big O notation is a form of asymptotic notation that means "the running time grows at most this much, but it could grow more slowly.". In this case algorithms with the same big O complexities can vary when we reconsider the constants & less significant terms, that is the reason why some algorithms run faster while they have the same Big O " Selection & Selection2 for example", Insertion sort performs quite well on nearly sorted sequences.
It can be seen, that for the already sorted array it needs just n steps, Bubble sort is Inefficient on large sized arrays but on the other hand it 's a straightforward algorithm to implement, Selection sort improves the performance of bubble sort but also slow for big inputs.

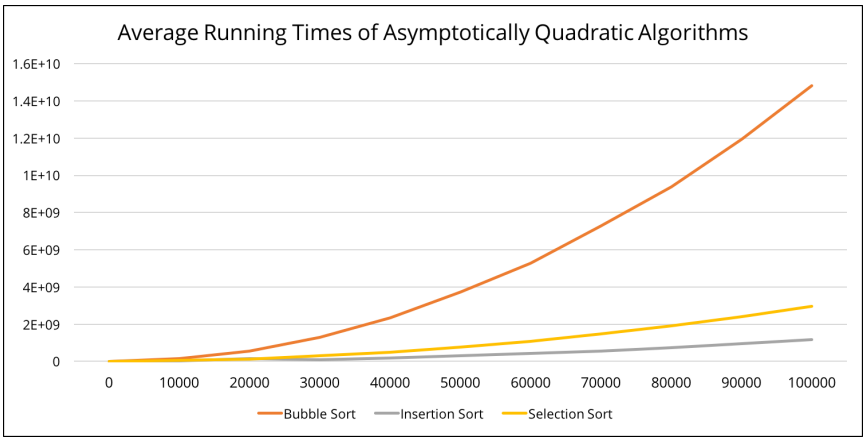

Figure (1):Analysis of Sorting Algorithms

Conclusion

In conclusion, we can see by the implementation that each algorithm varies in execution & hence, in optimality. It depends on the problem we are solving but in general, every algorithm does its work & between those 3 algorithms, Insertion sort has the best performance according to time consumption, we tested the best, worst & average case for each one & observed that bubble sort was the slowest running algorithm, selection sort & improved selection sort are in the middle, Big O notation can tell a lot about the algorithm used & helps a lot in the analysis process.

References

[1].Sorting Algorithms, Brilliant.org.

[2].Analysis of Sorting Algorithms, Hashan Punchihewa.