Sorting Evaluation

Rose Ramirez

I. Introduction

The goal of this assignment is to compare various sorting functions and observe their efficiency and effects on CPU and RAM. I implemented four different sorting algorithms, Bubble Sort, Insertion Sort, Quick Sort, and Bogo Sort. These algorithms are all vastly different and will have contrasting results. In order to test the sorting functions I implemented, I used four different data sets. With sizes 10, 100, 1000, and 10000 elements each. I timed each execution of a sorting function, and compiled my results here.

Why I Decided to Evaluate Bogo Sort

Upon research of other sorting algorithms not discussed in class, I came across Bogo Sort, or Monkey Sort that has incredible inefficiency. I thought it would be interesting to test and see how it performs.

II. Expectations and Results for Time Differences

From what we studied in class, I expect Quick Sort to have the fastest sort time, followed by Insertion Sort, Bubble Sort, and Finally Bogo Sort.

Observed Time Differences

1) Bubble Sort: Bubble Sort's Algorithm compares each value to each value. Bubble Sort always has an efficiency of $O(n^2)$, since it has nested for loops. As with most sorting algorithms, the time to sort is less when the data size is less.

2) Insertion Sort: Insertion Sort's algorithm has a marker that has items to the left partially sorted and inserts the marker in the left portion. The average performance for insertion sort is $O(n^2)$, and the best case is $O(n)$ for presorted data, since there is a while loop that wouldn't get executed.

3) Quick Sort: Quick Sort's Algorithm is a divide and conquer algorithm that partitions the array at a pivot point where all items higher than the pivot are on one end of the array and all items lower than the pivot point are on the other end. The efficiency of Quick Sort is $O(nlogn)$ since it divides the data in half at each execution.

4) Bogo Sort: Bogo Sort's algorithm randomizes the data until the data is sorted. The Average case for Bogo Sort Efficiency is $O(n * n!)$, and the worst case for Bogo Sort is infinite, which makes it extremely inefficient. Bogo sort works best for smaller data sets, ranging from size 0 to 5.

III. Trade offs

As Expected, Quick Sort was the most efficient sorting algorithm of those tested. The trade-off in sorting algorithms is that as the complexity of the algorithm increases, the efficiency becomes more optimized.

IV. Choice of Programming Language

I chose C++ as the programming language for this assignment because it is the programming language that we used in class and it is a relatively fast programming language that allows the programmer to control memory allocation.

V. Shortcomings of Empirical Analysis

Empirical Analysis is beneficial in that it gives concrete values for execution time, but is time consuming since we have to fully implement and execute the algorithms in order to observe their performance. I was not able to full test all possible data sets and therefore, my conclusions are not complete. I only tested four different data sets, so I was unable to observe outlying data sets. Empirical Analysis also depends on the environment it is executed in, so performance depends on the computer you are using to test it.

References

[1] GeeksForGeeks.org
[2] IEEE.org
[3] CPlusPlus.com