Sorting Algorithms - Data Structures

Eric Wasserman

December 2019

1 Bubble Sort - Run time: .289 $O(n^2)$

2 Insertion Sort - Run time: .169 $O(n^2)$

3 Selection Sort - Run time: .177 $O(n^2)$

4 Quick Sort - Run time: .032 $O(logn)$

5 Analysis

The specific run times of each algorithm did not surprise me. Bubble sort, the easiest to implement, took the longest. The next longest was selection sort, however, that was only .08 seconds slower than insertion sort. My sorting algorithms only sorted around 100 doubles, so the run times for each are very low. That being said we can use percentages to easily compare the relative run times. Selection sort was around 41 percent faster than bubble sort - this number is not what I expected. I thought the relative differences would be a lot slimmer. Insertion sort was around 43 percent faster than bubble sort, and insertion sort is just barely faster than selection sort. I was most surprised by how fast quick sort was. It makes sense because of it's time complexity but I didn't realize the difference would be so extreme.

Based on my times you can see the advantages in terms of run time of each algorithm. That being said the fastest, quick sort, was also the hardest to implement. The run time for insertion sort can reach O(n) if the data is already partially sorted. Although bubble sort and selection sort are both $O(n^2)$ selection sort has less swaps and therefore has a faster run time. One of the short comings using empirical analysis in this situation is that it is not time effective.

In general all these times are very fast because we used C++. It is a compiled language so it is much faster than interpreted languages such as Python or Java.