Assignment 6

Katherine Hansen

May 17, 2019

1 Introduction

For this assignment I ran a set of 50,000 unsorted values through a bubble sort, an insertion sort, a quick sort, and a cocktail sort, also known as a cocktail shaker sort. I then looked at how long each sort took on the same set of data to understand how each big O runtime works.

2 Conclusion

My conclusions were not entirely what I expected. For bubble sort, it took 5539 ms to sort the 50,000 numbers. For insertion sort, it took 3954 ms to sort, better than the bubble sort, but certainly not O(n), because the numbers were not presorted. I thought quick sort would be the fastest. However, during some of the tests, my computer was not able to complete the quick sort. My guess is that my CPU was not able to handle the quick sort. Quick sort is a better option for a machine that can handle using more CPU. The cocktail sort ran in 4129 ms. A cocktail sort is when you start at the front and swap as you go, and once you get to the end, you turn around and sort the other way. This has a big O of O(n*n), so it is not surprising that the speed wasn't very good.

I was surprised that my computer couldn't handle a quick sort of 50,000 values. During a test with 1,000 values, quick sort did run faster than the other three, but the times weren't drastic due to the small sample size. Empirical analysis can be problematic, because, as I discovered personally, every machine will run the same code differently due to hardware limitations. Although it is good for visualizing how different runtimes can be, it is not data that is easy to replicate. Even when I ran the tests just a few times, the sorts always took a slightly different amount of time.

If your computer can handle quick sort, then that is the best one to choose. However, if your machine is not built to handle that, then insertion sort is a better choice, due to the potential for a slightly faster runtime.