Assignment 6 Report

James Kistner
CPSC 350-01
Rene German
13 December 2018

During this program I noticed some expected and unexpected results over the four different algorithms. At first, I ran the sorting algorithms using a relatively small data set of about 1000 double numbers. To my surprise all of the run times were 0 seconds. So, I increased the amount of numbers to 200,000. This is where I started to actually notice a difference. I knew that merge sort and quick sort were going to be the fastest, but I didn't expect that a run time of $nlog(n)$ would be much different from n^2. When I sorted 200,000 numbers the run time for bubble sort was roughly 117 seconds, insertion sort was 20 seconds, quick sort and merge sort were both 0 seconds. There were two things that surprised me with these results. First off, I didn't realize how insertion sort could perform 5 times as quick as bubble sort. The data, to the best of my knowledge, was unsorted. Even then there were substantial boosts in performance. The second surprising fact was that merge sort and quick sort ran in 0 seconds. That is insane! I wanted to test how many numbers I could sort with using quick sort and merge sort while their sorting time remained < 0 seconds. Unfortunately, there was a limit on how many numbers I could sort without getting a "segmentation fault" error, so I'll save that test for another time. I'm not sure exactly how using C++ to create these algorithms impacts the performance of the algorithms, but I will say that I don't think I would know how to code quick sort and merge sort in any language other than C++. I mean, the shortcomings of this assignment are very clear. We knew before hand that the sorting algorithms in terms of speed should go bubble sort, insertions sort both with $O(n^2)$, followed by either quick sort or merge sort with $O(nlog(n))$, so programming these functions were wasted time. However, there are some unmentioned perks that come with doing empirical analysis. It is very rewarding and compelling to actually verify these theorized run times. Also, the results are more detailed. This assignment was insightful as it finally proved these Big O theories we have been talking about all semester!

References:

https://www.geeksforgeeks.org/merge-sort/
https://www.geeksforgeeks.org/quick-sort/
https://www.geeksforgeeks.org/insertion-sort/
https://www.youtube.com/watch?v=eWr7etWd3Zo