Ryan Kennedy

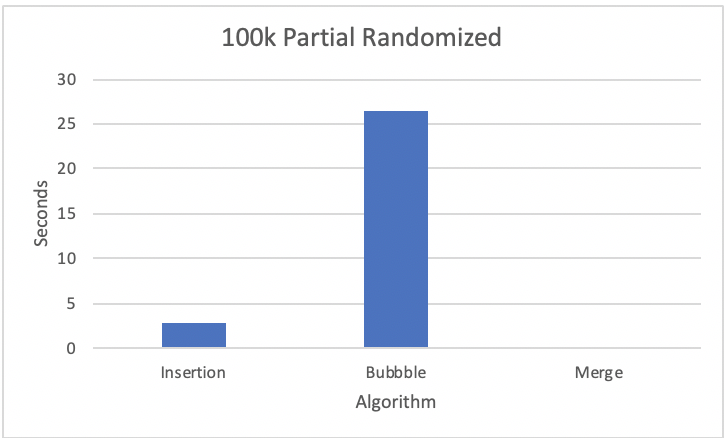
Professor Clark

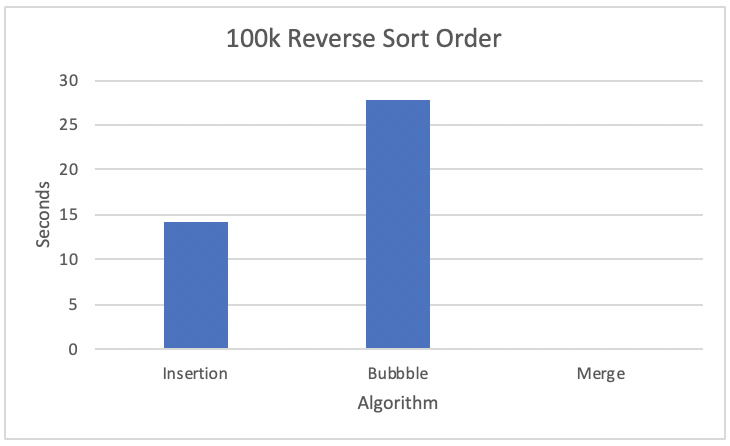
CS 3353

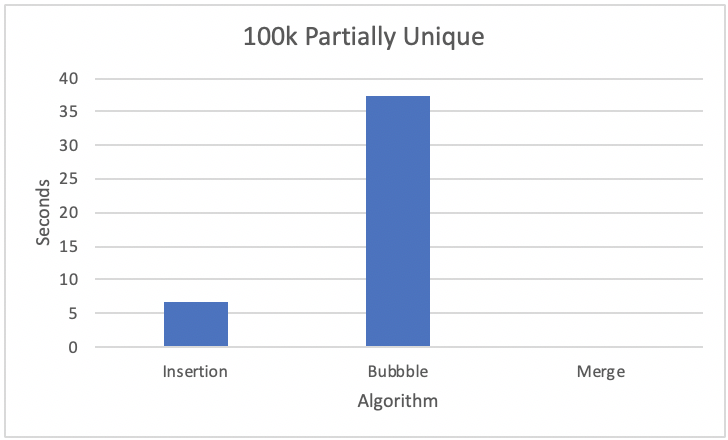
September 12, 2019

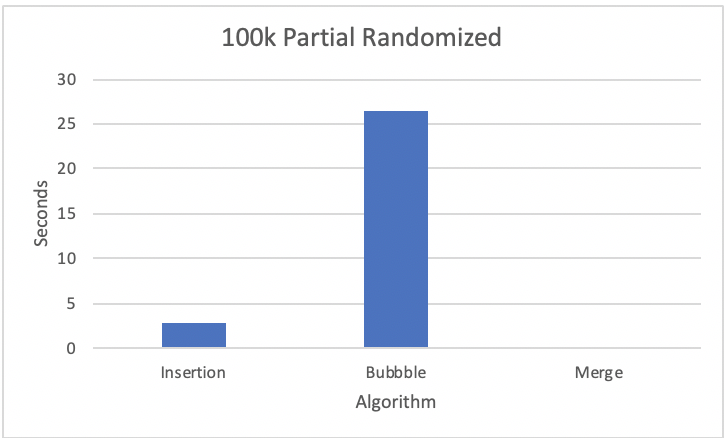
Analysis of Insertion, Bubble and Merge Sort

In this lab I explored the differences of 3 sort algorithms, insertion, bubble, and merge and found there were vast differences in the time it took the algorithms to sort the exact same set. I used 4 different types of data sets with 4 different data sizes of each set type to test the algorithms and ran the algorithms 15 times each to generate data. Through this analysis I found there were some data set types that were sorted faster than other types. Overall, I found merge sort to perform much better than the other two algorithms. On average across all data set types I found merge to 2,119 times faster than bubble sort and 550 times faster than insertion sort. This is why merge sort appears to not be visible on my graphs[[1]](#footnote-1).

 The fully randomized data set was on average the slowest method for both bubble sort and merge sort but was not the slowest method for insertion sort. As you can see from the graph on the right, bubble sort took significantly longer than insertion sort and merge sort. On average bubble sort took 42.47 seconds to sort 100,000 numbers fully randomized which is 14% slower than bubble sorts next fastest time, the 100,000 partially unique data set. It only took Insertion sort 7.869 seconds to sort the same data set which is 79% faster on average than insertions sort’s slowest time, the 100,000 reverse sort order set. Merge sort only took .02 seconds to sort the list, which was only 5% slower than the next fastest data set for merge, the 100,000 partially unique data set. From this data it is apparent that merge sort’s time is similar to bubble sort’s if you compare their percent differences to the next fastest time, but insertion sort does not follow this pattern.

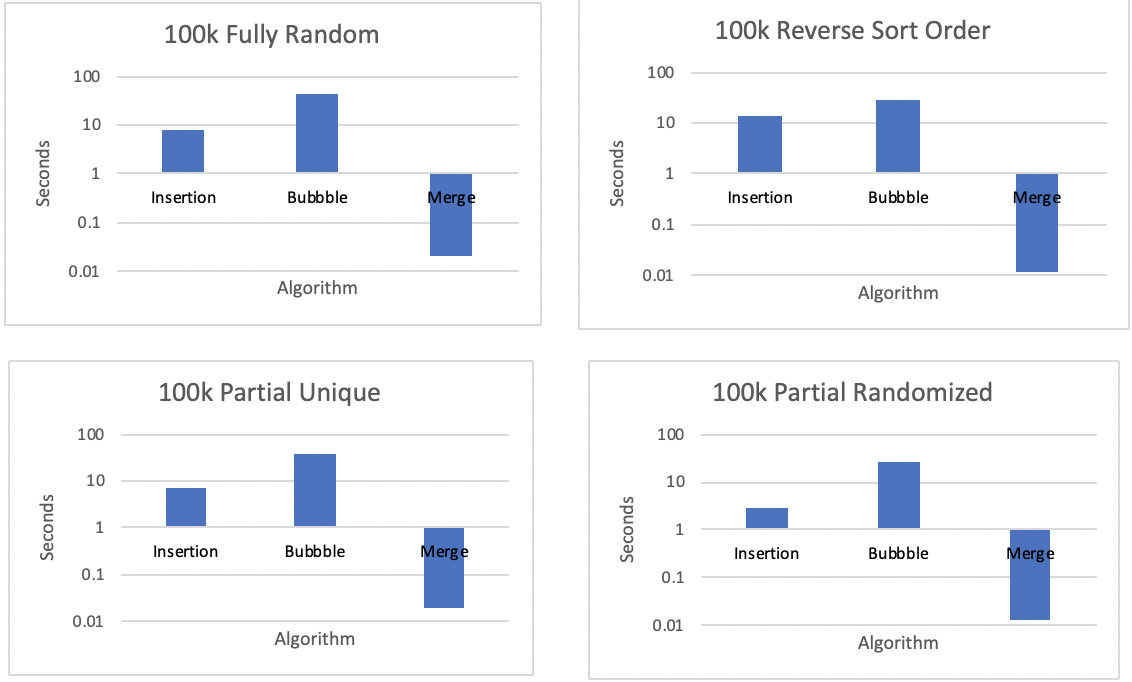
 Secondly, the reverse sort order data set was the slowest data set for insertion sort, but the fastest data set for merge sort and the 3rd fastest for bubble sort. The graph on the right shows how long each algorithm took for this data set. This data set took bubble sort 27.82 seconds on average to sort, which is only 5% slower than the fastest data set for bubble sort, the partially randomized set. Insertion sort took 14.09 seconds on average to sort, which is 79% slower than the next fastest data set, the fully randomized set. Merge sort only took .011 seconds which is about 20% faster than the next fastest data set, the partially randomized set.

 Next, the partially unique data set was the 3rd fastest data set for the bubble sort and merge sort, but the 2nd fastest data set for insertion sort. From the graph on the right it is apparent that insertion sort is remarkedly faster than bubble sort. Bubble sort took 37.39 seconds on average to sort this data set, which is 34% slower than the next fastest data set for bubble, the reverse sort order set. Insertion sort only took 6.86 seconds to sort the data which is 15% faster than the next fastest data set for insertion, the fully randomized set. Merge sort on average took .019 seconds to sort the set, which is only 5% faster than the fully randomized data set, the slowest data set for merge sort.

 Finally, the partially randomized data set was the fastest data set for insertion and bubble sort, but the 2nd fastest for merge sort. Similar to the partially unique data set graph, the graph on the right shows that insertion sort was much faster than bubble sort. On average bubble sort took 26.47 seconds to sort the data set, which is 5% faster than reverse order data set, the 2nd fastest data set for bubble. Insertion sort only took 2.79 seconds on average to sort the list, which was 146% faster than the next fastest data set, the partially unique set. Merge sort on average took .013 seconds to sort, which is 20% slower than the fastest data set for merge, the reverse sort order.

From this data it is apparent that merge sort is remarkably faster than the other algorithms. I can also tell that insertion sort and bubble sort are best for partially randomized sets, while merge sort is best for reverse sort order lists. Another conclusion I can draw from the data is that merge sort is the most consistent for sort times compared to the other 3 algorithms, while insertion sort is not that consistent and bubble sort is just barely less consistent than merge sort.

Extra Logarithmic Graphs:



1. For this reason, I have included graphs with a logarithmic scale at the bottom of this document and on my excel sheet. [↑](#footnote-ref-1)