CMPT435 Assignment2

Brian Sprague

September 30, 2020

1 Introduction

This is my LaTeX submission for Assignment 2. In this document you will find a table showing my comparison numbers for each of the four sorts, as well as a list of the asymptotic run times for each sort with an explanation for why.

Figure 1: Java

2 Comparisons

Selection	Insertion	Merge	Quick
3,459	114,309	5,413	3,643

Here in this table I have listed the number of comparisons that were done in each of the four sorting algorithms that I created. From this info you can make assumptions about which are the most efficient sorting algorithms.

3 Asymptotic Run Time: Selection Sort

The asymptotic run time of selection sort is O(n2) ...(n squared). This is because there are three parts to look at in selection sort: index, sort, and the rest of the run time. The run time for the indexing is O(n2) because we don't care about constants. The run time for the sorting is O(n). The run time for the rest of selection sort is also O(n). We then take the mist significant one which is O(n2) and that is now the asymptotic run time for selection sort.

4 Asymptotic Run Time: Insertion Sort

The asymptotic run time of insertion sort is O(n2) ...(n squared). The formula we get for insertion sort is:

c $\cdot(n-1+1)((n-1)/2) = cn2/2 - cn/2c(n1+1)((n1)/2) = cn2/2cn/2$

That isn't the most clear formula, but it can be broken down into O(n2) because we ignore the constants. There are however some cases where the run time is not O(n2). Sometimes, the run time can be O(n) if the list that we are sorting is already sorted and the part that we are adding is being added into the correct place.

5 Asymptotic Run Time: Merge Sort

The asymptotic run time of merge sort is O(n x log2(n)) This is because you need to examine the sort for its divide side and its conquer side, to use the example from class. The run time of the divide side is O(log2(n)) and the run time of the conquer side is O(n). You need to look at both run times because for merge sort you are not conquering while you are dividing, each step occurs on its own. Now, you combine the two run times and you get O(n x log2(n)) which is the total run time for merge sort.

6 Asymptotic Run Time: Quick Sort

Quick sort is different from merge sort in that it divides and conquers at the same time. There are two examples of run time for quick sort that I have. One is the worst case and the other is avoiding the worst case. For the worst case, the run time would be O(n2) and this would occur when your pivot value consistently creates one side of the list being of size 1, while the other side becomes the largest size possible. This is obviously the worst case because it would result in the longest run time due to it needing to constantly call itself through recursion. The run time for avoiding the worst case is O(nlogn). This is the same value as that of merge sort, which makes sense as the two sorts are similar.