Results

Figure 1 and 2 both give interesting results about the affect of sortedness on the time complexity of the quick sort algorithm when the pivot is chosen as the last element in the sequence.

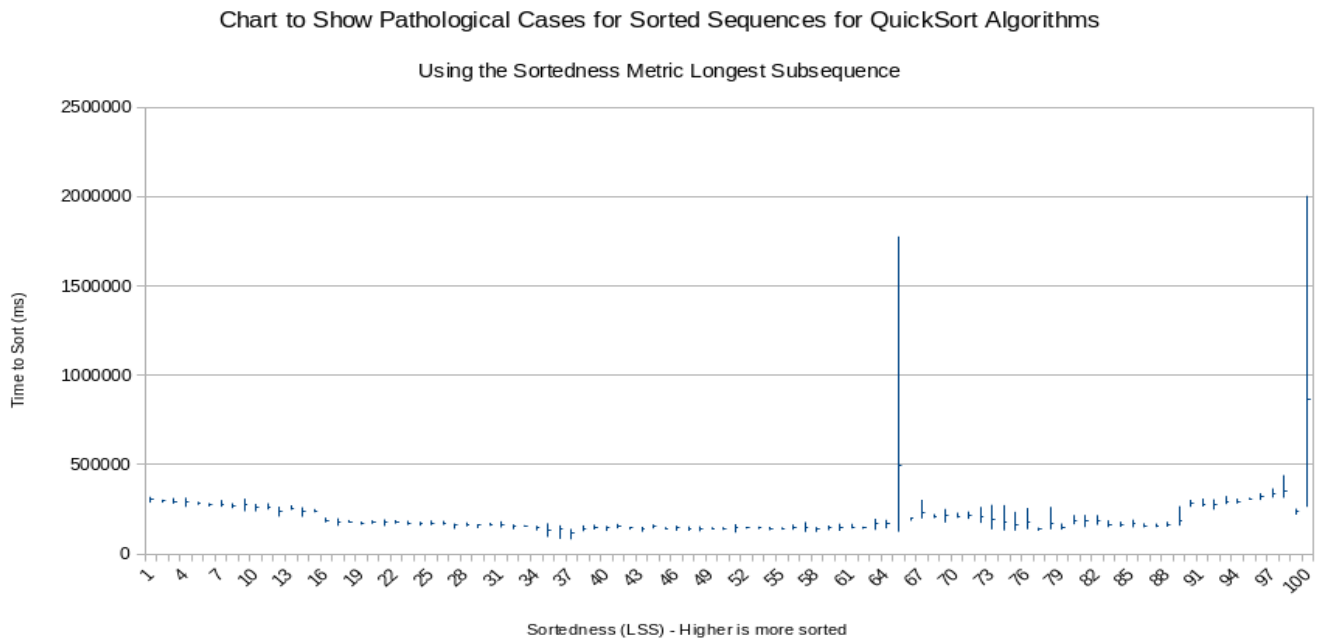

Figure 1: Results for quick sort time complexity for a range of sortedness values using the longest subsequence sortedness metric.

Figure 1 shows an overall trend of reverse-sorted sequences tend to be consistently more inefficient than sequences which are more random. This is shown as reverse-sorted lists give a longest subsequence value of 0. However, it is clear that there is a pathological case when sequences are sorted which gives a LSS value of the sequence size (in this case 100). This average is much higher than all other values, and it's range is also must greater than most other trials. However, an interesting pathological case is at the LSS of 65. This may show that the sortedness algorithm is not ideal, as it allows for another pathological case past the fully sorted list.

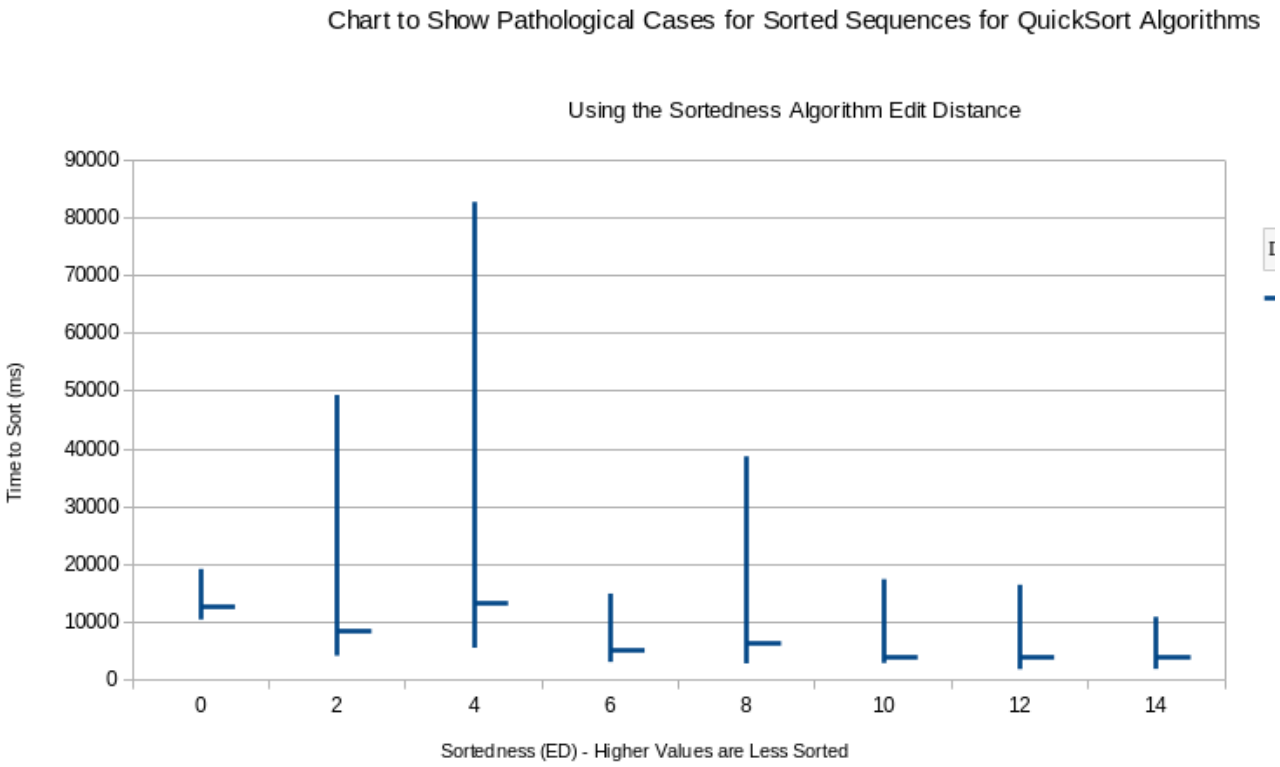

Figure 2: Results for quick sort time complexity for a range of sortedness values using the edit distance sortedness metric.

Figure 2 mostly agrees with figure 1 in it's findings. As lower values are more sorted, we can see that the two highest values are at the 0, and 4 ED values. A sortedness of 0 means that the list is fully sorted here and we can see that, despite the variation of values, the average is still highest here. Semi-sorted lists at 4 appears to be the same pathological case that is being displayed in figure 1.

Overall, my results show that the sortedness metrics I have used are mostly useful in determining whether or not a sequence will likely be efficient as it can be hard to determine what a semi-sorted list, or very unsorted list looks like until we get a metric value. We can also then detect the pathological cases as we could map values to the graphs generated from my dataset.

The difference between the two sortedness metrics I used can be seen in the less sorted sequences for both. LSS shows an increase in time taken to sort what are deemed less sorted by LSS, but there is a consistent decline in time taken for the ED metric.

The ED metric appears to be better at determining what sequences will be pathological, as we can usually see that lower ED values equate to less efficiency. For the LSS, because of the difference explained above, it would require more work. However, the inefficiency of the algorithm to determine the ED value is an important issue to raise.

Methodology

I am confident that my results are accurate. I collected my results using algorithmically generated sequences which were aimed to generate a wide range of sortedness values. However, an improvement that could be made is to the ED metric, as determining the level of sortedness was unfeasible due to it's time complexity when sequences over the size of 14 were used on my machine. The results from the LSS data does however support the claims made from the limited ED data.

To limit the effect of machine differences, I added data-ranges to both of my graphs which was possible due to the addition of repeated tests. I then found the mean average for each sequence, and added the minimum and maximum values to the graphs.