Selection and Merge Sort Summary

Selection sort should have a complexity of O(n). In the trials ran, selection sort does seem to exponentially grow, however I expected it to be more convex than it appears. As shown in figure 1, when n^2 is scaled and placed on top of the selection sort trials, n^2 has a greater growth in operations than the results for selection sort had.

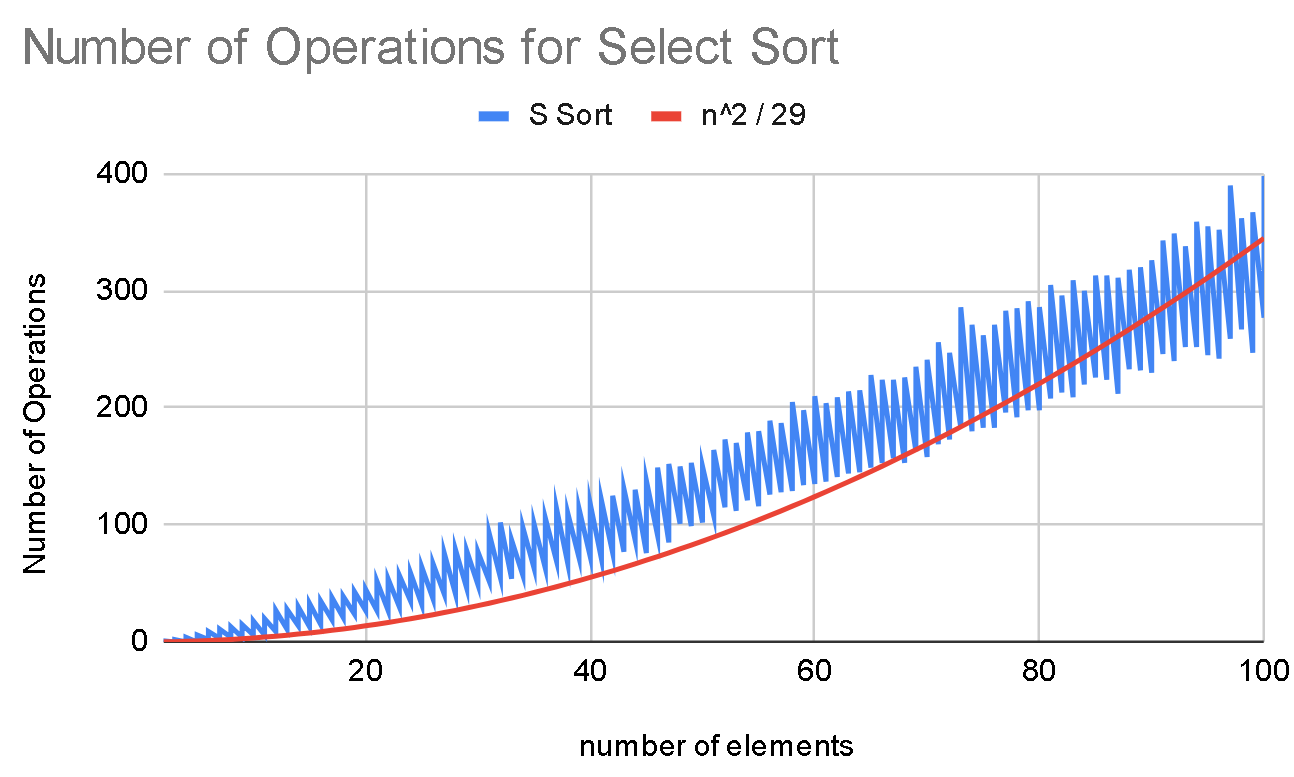


Figure 1

Merge sort should have a complexity of O(n log(n)). In the trials ran, merge sort almost perfectly matches this time complexity. Figure 2 shows n log(n) scaled and placed on top of the merge sort trials. The trials are visibly very similar, and appear as expected.

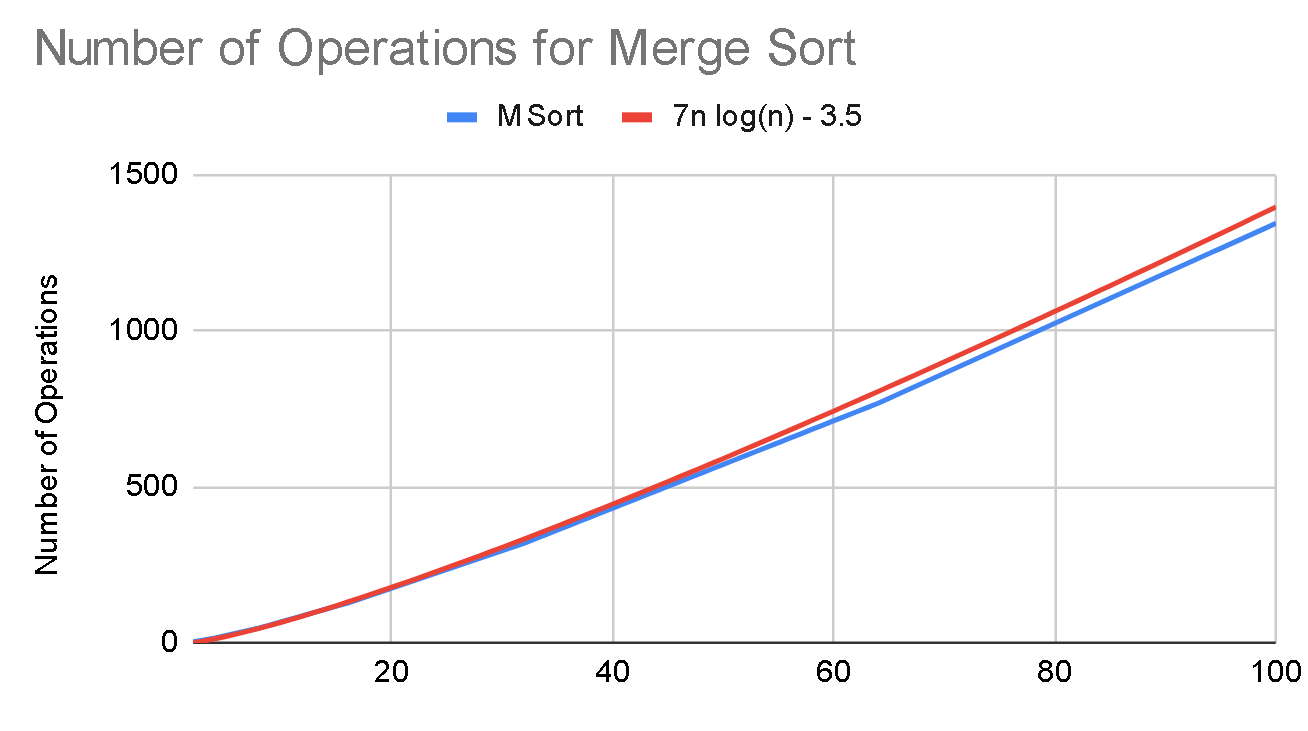


Figure 2

# Merge Sort

Merge sort works by recursively splitting the array into two pieces, until all elements are separated into their own array. Then, each array will be merged together with another array, sorting the combined array as they are merged. The combination works by repeatedly taking the lowest element of each array, comparing them, and moving the lowest of the two to the new array. Once all arrays have been merged with another array, the merging process repeats for these new arrays. This process repeats until one sorted array is left, this array is the sorted array.

# Selection Sort

Selection sort works by iterating through the array and saving the index of the lowest element. Once this index is found, the first element and the lowest element are swapped. This completes the first index. Now the array will be iterated through again, skipping the completed index, to find the next lowest element and swap it with the second element. This process repeats for each index of the array until the array is sorted least to greatest, fully sorting the array.