sorts runtime.txt

Yvonne Cheng

March 15, 2023

1 Introduction

In this paper, I will demonstrate that our hypothesis of order of runtimes of Merge Sort and Quick Sort, binary and tertiary, was correct; I will compare the runtime of the Merge Sort and Quick Sort in 3 different ways in order to prove that the execution time for Tertiary algorithms, O(log3n), is quicker than Binary algorithms, O(log2n) and that for large lists, Quick Sort is more efficient than Merge Sort under the same binary or tertiary algorithm.

2 Evidence&Analysis

2.1 Merge Sort Tertiary vs Merge Sort Binary Runtime

Through graphing the execution time in Merge Sort Tertiary, O(log3n), versus the execution time in Merge Sort Binary, O(log2n), it is shown by Figure 1 that the Merge Sort Tertiary execution time is faster than the Merge Sort Binary execution time. Given a randomly generated list of 2,000 number, it is shown that the Merge Sort Tertiary runtime(.0010s) takes much less time than the Merge Sort Binary runtime(.0032s), thus backing up our hypothesis of ternary algorithms being slightly faster than binary ones.

2.2 Quick Sort Tertiary vs Quick Sort Binary Runtime

Through graphing the execution time in Quick Sort Tertiary, O(log3n), versus the execution time in Quick Sort Binary, O(log2n), it is shown by Figure 2 that the Quick Sort Tertiary execution time is faster than the Merge Sort Binary execution time. Given a randomly generated list of 2,000 number, it is shown that the Quick Sort Tertiary runtime(.0012s) takes much less time than the Merge Sort Binary runtime(.0024s), thus backing up our hypothesis of ternary algorithms being slightly faster than binary ones.

2.3 Fastest to Slowest Runtimes of Merge Sort and Quick Sort(Binary, Tertiary)

Shown in Figure 3, the Quick Sort Tertiary algorithm is the quickest in terms of runtime in seconds followed by Merge Sort Tertiary, Quick Sort Binary, and finally Merge Sort Binary. This supports our hypothesis as we know that when it comes to larger data sets, Quick Sort is usually more efficient at sorting lists than that of Merge Sort. In addition, we also know the runtime in tertiary algorithms of O(log3n) ¡ binary algorithms of O(log2n).

3 Conclusion

In this paper, I have demonstrated that the execution times of Merge Sort and Quick Sort, binary and tertiary, is the fastest to slowest in the following order: Quick Sort Tertiary, Merge Sort Tertiary, Quick Sort Binary, and Merge Sort Binary. Therefore, our hypothesis of order of magnitude estimates of Merge Sort and Quick Sort, binary and tertiary, runtimes was correct.

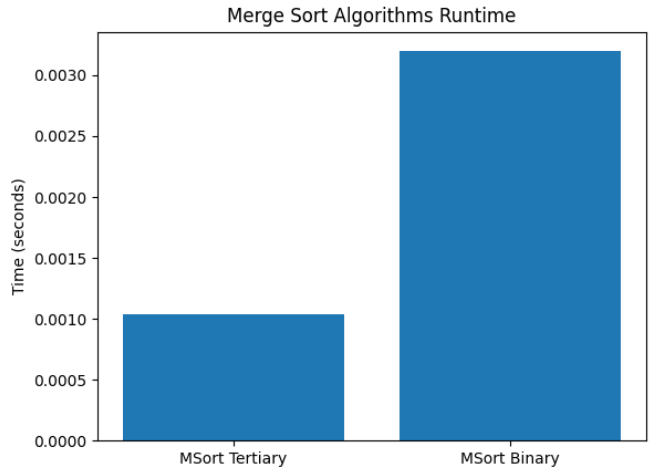

Figure 1: Merge Sort Tertiary vs Merge Sort Binary Runtime Graph

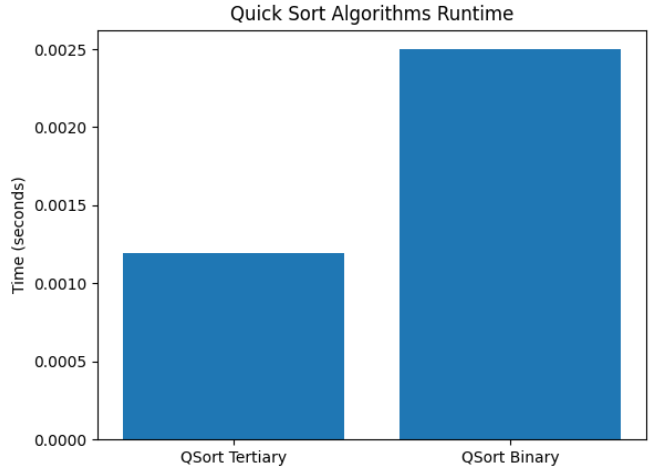

Figure 2: Quick Sort Tertiary vs Quick Sort Binary Runtime Graph

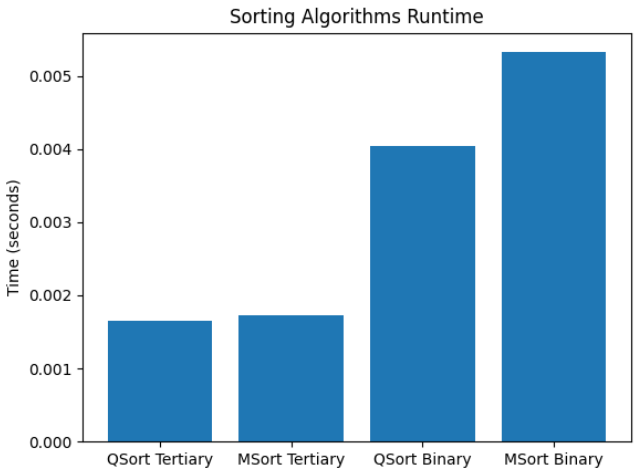

Figure 3: Merge Sort(Binary, Tertiary) vs Quick Sort(Binary, Tertiary) Runtime Graph