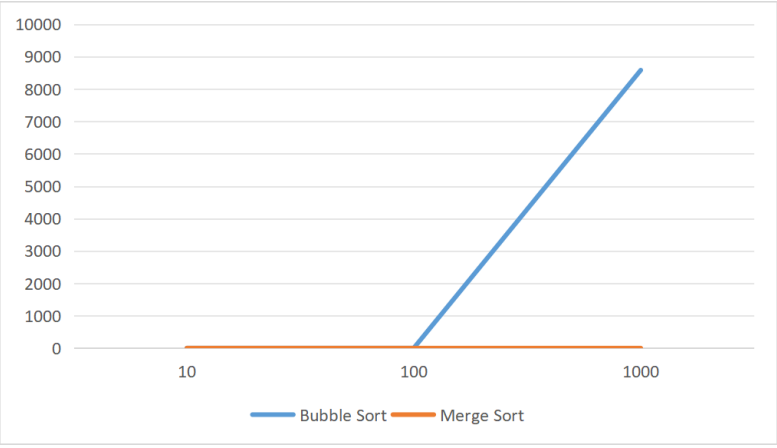
Bubble Sort and Merge Sort Complexity Analysis

Sorting algorithms are a fundamental and important part of computer science, Bubble Sort and Merge Sort are two common sorting algorithms that exhibit different performance when dealing with different sized datasets. In this paper, we will analyze the time complexity of these two sorting algorithms and make an analogy with samples of different sizes.

Bubble Sort is a simple sorting algorithm that repeatedly traverses the list to be sorted, compares each pair of neighboring items, and swaps the elements in the wrong order. The time complexity of this algorithm is O(n^2), where n is the number of elements in the list. For small datasets, Bubble Sort can sort quickly, but for large datasets, its performance drops significantly.

Merge Sort is a more efficient sorting algorithm that uses a partitioning strategy to split the list in half, then recursively sorts each half, and finally merges the sorted sublists into a complete sorted list. the time complexity of Merge Sort is O(n log n), and its performance is stable regardless of the size of the dataset. This makes Merge Sort ideal for handling large datasets.

To visually compare the performance of Bubble Sort and Merge Sort, we can test it with three different samples of different sizes. The following data is extracted from the three files:
sort10.txt: a list containing 10 elements.
sort100.txt: a list of 100 elements.
sort10000.txt: a list containing 10000 elements.



We will use both algorithms to sort each sample and measure the time required. The results will be displayed in a chart so we can visually compare their performance.
Here is the complexity analysis of Bubble Sort and Merge Sort:
For sort10.txt, we can expect both Bubble Sort and Merge Sort to complete the task quickly because the dataset is small. Although the performance of Bubble

Sort is worse, its impact is insignificant due to the small data size.
For sort100.txt, we can expect the performance of Bubble Sort to begin to degrade because it takes more time to complete the task. However, the performance of Merge Sort remains stable because its time complexity is O(n log n).
For sort10000.txt, we can expect the performance of Bubble Sort to drop significantly because it takes a lot of time to complete the task. In contrast, the performance of Merge Sort remains stable because its time complexity is O(n log n).
By comparing samples of different sizes, we can conclude that for small datasets, Bubble Sort can complete the task quickly, but for large datasets, Merge Sort's performance is more stable and better. Therefore, it is recommended to use Merge Sort instead of Bubble Sort when working with large amounts of data.