**Analysis and Comparison of Sorting Algorithms:**

**Time Complexity of Bubble Sort:**

* Best-case time complexity: O(n) (when the input array is already sorted).
* Average-case time complexity: O(n^2).
* Worst-case time complexity: O(n^2).

**Advantages of Bubble Sort:**

* Simple to understand and implement.
* Requires minimal additional memory space (in-place sorting).
* Performs well on small datasets or partially sorted arrays.
* Best-case time complexity is linear (when the array is already sorted).

**Disadvantages of Bubble Sort:**

* Inefficient for large datasets due to its quadratic time complexity.
* Not recommended for practical use on large or unsorted datasets.
* Performance degrades quickly with an increase in the number of elements.

**Comparison with Other Sorting Algorithms:**

* Bubble Sort: Performs well on small datasets but is inefficient for larger ones.
* Insertion Sort: Similar to Bubble Sort in terms of time complexity, but generally performs slightly better in practice.
* Selection Sort: Similar time complexity to Bubble Sort, but typically performs slightly worse in practice.

**Performance Analysis:**

The performance analysis shows that Bubble Sort's time complexity makes it inefficient for larger arrays compared to other sorting algorithms like Merge Sort, which has a better time complexity (O(n log n)) and performs significantly faster on large datasets.

**Conclusion:**

Bubble Sort is a simple sorting algorithm suitable for small datasets or partially sorted arrays. However, its quadratic time complexity makes it inefficient for large arrays. When dealing with large datasets, other sorting algorithms like Merge Sort, Quick Sort, or even Insertion Sort and Selection Sort tend to perform better. Bubble Sort's main advantage lies in its simplicity and minimal space requirements, but its performance is inferior to more advanced sorting algorithms for large datasets.