Difference between Merge Sort and Quick Sort

Merge sort and quick sort are two popular sorting algorithms used to sort an array or list of elements. Both are efficient sorting techniques that have their strengths and weaknesses. Here are some of the key differences between merge sort and quick sort:

Algorithm

The algorithm used by merge sort is a divide-and-conquer algorithm, while quick sort uses a partitioning algorithm.

Merge Sort: 1. Divide the unsorted list into n sub-lists, each containing one element. 2. Repeatedly merge sub-lists to produce new sorted sub-lists until there is only one sub-list remaining. This will be the sorted list.

Quick Sort: 1. Choose a pivot element from the array. 2. Partition the array such that all elements smaller than the pivot are to its left, and all elements larger than the pivot are to its right. 3. Recursively apply quick sort to the left and right sub-arrays.

Time Complexity

The time complexity of merge sort and quick sort is different.

Merge Sort: - Worst case: O(n log n) - Best case: O(n log n) - Average case: O(n log n)

Quick Sort: - Worst case: O(n^2) - Best case: O(n log n) - Average case: O(n log n)

Space Complexity

The space complexity of merge sort and quick sort is different.

Merge Sort: - Worst case: O(n) - Best case: O(n) - Average case: O(n)

Quick Sort: - Worst case: O(n) - Best case: O(log n) - Average case: O(log n)

Stability

Merge sort is a stable sorting algorithm, while quick sort is not.

Implementation

Merge sort requires additional memory to store the temporary arrays during the merging process. Quick sort does not require additional memory.

Conclusion

In summary, merge sort is more efficient than quick sort in terms of worst-case time complexity and stability. However, quick sort is faster in practice for small arrays and has a lower space complexity than merge sort. The choice between merge sort and quick sort depends on the specific requirements of the problem at hand.