Time Complexity and Space Complexity Trade-offs

Space complexity refers to how much memory an algorithm uses, and it often has a trade-off with time complexity. In many cases, optimizing for time can result in higher space usage, and vice versa. Understanding this balance is crucial when designing efficient algorithms.

Example: Merge Sort vs Quick Sort

- **Merge Sort**: While Merge Sort has an optimal time complexity of O(n log n) for sorting large datasets, it requires additional space—O(n)—for auxiliary arrays used during the merging process.
- **Quick Sort**: Quick Sort also has an average time complexity of O(n log n), but its space complexity is usually O(log n), because it uses less memory due to its in-place sorting method. This makes Quick Sort more space-efficient than Merge Sort, especially for larger datasets.

Balancing Time vs Space:

Sometimes, it might be acceptable to use more memory if it leads to faster execution. In contrast, in environments with limited memory (like embedded systems or mobile apps), space complexity might need to be prioritized over time complexity.