Assignment 11 - Question 6 - Complexity Analysis

Omar Abdelhady

900222873

May 26, 2025

Complexity Analysis

Time Complexity

The time complexity of the provided backtracking algorithm can be analyzed as follows:

- The algorithm explores all possible ways to divide the input set into two subsets, considering each element for either subset.
- For a set of size n, there are 2^n possible ways to assign each element to a subset.

The recurrence relation for the algorithm is:

$$T(n) = 2T(n-1) + 1$$

Expanding this recurrence:

$$\begin{aligned} T(n) &= 2T(n-1) + 1 \\ T(n-1) &= 2T(n-2) + 1 \\ T(n-2) &= 2T(n-3) + 1 \\ &\vdots \end{aligned}$$

Continuing this pattern, we get:

$$T(n) = 2^n T(0) + 2^{n-1} + 2^{n-2} + \ldots + 2^0$$

The sum of the geometric series simplifies to:

$$T(n) = 2^n T(0) + (2^n - 1)$$

Since $T(0)$ is a constant, we can denote it as c. Thus, the time complexity is:

$$T(n) = O(2^n)$$

Space Complexity

The space complexity of the algorithm can be analyzed as follows:

- The algorithm uses a recursive approach, which requires space for the call stack.
- The maximum depth of the recursion is n, leading to a space complexity of $O(n)$ for the call stack.
- Additionally, the algorithm uses space for the subsets and other variables, but these are bounded by $O(n)$ as well.
- Therefore, the overall space complexity is $O(n)$.

Conclusion: The time complexity of the backtracking algorithm is $O(2^n)$, and the space complexity is $O(n)$.