ALGORITHM COMPLEXITY ANALYSIS

Algorithm Name: Wagner-Fischer Algorithm

TIME COMPLEXITY

• **Best Case:** The best-case time complexity of the Wagner-Fischer algorithm is O(n*2), where n is the length of the input strings. This occurs when the two input strings are identical, and no operations are required to transform one into the other.

• **Worst Case:** The worst-case time complexity of the Wagner-Fischer algorithm is also O(m*n), where m and n are the lengths of the input strings. This occurs when the two input strings are completely different, and the algorithm has to consider all possible operations (insertion, deletion, and substitution) for each character.

• **Average Case:** The average-case time complexity of the Wagner-Fischer algorithm is also O(mn), where m and n are the lengths of the input strings. This is because, on average, the algorithm needs to consider a portion of the matrix of size mn to find the minimum edit distance between the two strings.

Note: The time complexity of the Wagner-Fischer algorithm is based on the size of the dynamic programming matrix used by the algorithm, which is of size (m+1) x (n+1), where m and n are the lengths of the input strings. Each cell in the matrix requires constant time to compute, leading to a total time complexity of O(m*n) for the algorithm.

SPACE COMPLEXITY

The space complexity of the Wagner-Fischer algorithm is O(m*n), where m and n are the lengths of the input strings. This space is required to store the dynamic programming matrix used by the algorithm to compute the minimum edit distance between the two strings.

Explanation:

The algorithm uses a 2D matrix of size (m+1) x (n+1) to store the minimum edit distance values for each pair of prefixes of the input strings.

Each cell in the matrix corresponds to a subproblem in the dynamic programming approach, and its value is computed based on the values of adjacent cells.

Since there are (m+1) rows and (n+1) columns in the matrix, the total space required is O(m*n).