The Sequence Alignment Problem

Time Analysis:

The time complexity of the dynamic programming algorithm for sequence alignment is O(mn), where m and n are the lengths of the input sequences. This is achieved by using a two-dimensional matrix to store intermediate scores and iteratively filling in the matrix based on the recurrence relation. The matrix has dimensions (n+1) x (m+1), and each entry requires constant time to compute.

Approach Explanation:

The goal of sequence alignment is to find the optimal alignment between two sequences, maximizing the total alignment score. The scoring is defined by a substitution matrix that assigns scores to matches, mismatches, and gaps.

Initialization:

Create a matrix to store scores for subproblems. Initialize the first row and column to represent the scores of aligning prefixes of the sequences with gaps.

Dynamic Programming Filling:

Iterate through the matrix, computing the score of each subproblem based on the scores of its neighbors. The recurrence relation is used to calculate the optimal score at each position.

Traceback:

Once the matrix is filled, trace back from the bottom-right corner to reconstruct the optimal alignment. At each step, determine the direction (diagonal, up, or left) based on the optimal score.

Alignment Construction:

Build the aligned sequences by following the traceback path. Insert gaps as needed to match the optimal alignment.

Total Score Calculation:

Calculate the total score of the alignment by summing the scores from the substitution matrix for each matched pair.

Summary:

The dynamic programming approach for sequence alignment optimally solves the problem by breaking it down into overlapping subproblems and efficiently computing their solutions. The time complexity of O(mn) ensures scalability for larger sequences. The algorithm's correctness is guaranteed by considering all possible alignments and selecting the one with the maximum total score.