Sequence Alignment Algorithm Report

The Sequence Alignment method introduces gaps and maximizes a scoring scheme based on character similarity in order to get the optimal alignment between two sequences. To compute the best alignment in an efficient manner, this approach uses a dynamic programming (DP) paradigm.

Time Analysis:

Time Complexity: The time complexity of the algorithm is O(m * n), where m and n represent the lengths of the input sequences. The reason for this is that in order to fill the DP table, the nested loops iterate through every character in both sequences.

Space Complexity: The DP table also contributes to the O(m * n) space complexity. For every potential alignment of subsequences, the intermediate scores must be stored in a 2D array.

Algorithm Explanation:

Initialization: The algorithm initializes a DP table with dimensions (m+1) x (n+1), setting initial scores based on gaps and mismatches.

Scoring Matrix: A scoring matrix is constructed to assign scores for matches, mismatches, and gaps. Each character comparison yields a score based on the matrix.

Dynamic Programming: The DP table is filled iteratively, considering three possibilities at each cell: match/mismatch, deletion, or insertion. The maximum score among these options is recorded in the cell.

Backtracking: Following table completion, backtracking starts from the bottom-right cell to reconstruct the actual alignment. It traces the path through the DP table, determining matches, insertions, or deletions to form the aligned sequences.

Algorithm Explanation:

Initialization: Based on gaps and mismatches, the algorithm sets initial scores in a DP table with dimensions of (m+1) x (n+1).

Scoring Matrix: To allocate points for matches, mismatches, and gaps, a scoring matrix is created. Based on the matrix, a score is obtained for every character comparison.

Dynamic programming: Three options are taken into account at each cell of the DP table: match/mismatch, deletion, or insertion. This process fills the table iteratively. The cell has the highest possible score among these possibilities.

Backtracking: To recover the real alignment after the table is completed, backtracking begins in the bottom-right cell. In order to create the aligned sequences, it follows the path through the DP table, looking for matches, insertions, or deletions.

Output: Based on the scoring matrix, the algorithm returns the two aligned sequences as well as the alignment score, which shows how similar the sequences are.

In conclusion, the Sequence Alignment algorithm computes the ideal alignment between two sequences in an efficient manner by utilizing dynamic programming. Because of its O(m * n) time complexity, it can be used in real-world scenarios and allows for precise sequence comparisons.