How does the Smith-Waterman alignment algorithm differ from the Needleman-Wunsch algorithm?

The Smith-Waterman and Needleman-Wunsch algorithms are both used for sequence alignment, but they differ in their approach. The Needleman-Wunsch algorithm is a global alignment algorithm, meaning that it aligns the entire length of two sequences. It assigns a score to each possible alignment and finds the alignment with the highest score. This algorithm is useful for aligning two sequences that are similar in length and have similar patterns.The Smith-Waterman algorithm, on the other hand, is a local alignment algorithm, meaning that it aligns only the most similar regions of two sequences. It also assigns a score to each possible alignment, but it allows for gaps to be introduced in the alignment if it improves the score. This algorithm is useful for aligning sequences that have regions of similarity within larger, dissimilar sequences.Overall, the main difference between the Needleman-Wunsch and Smith-Waterman algorithms is that the former aligns the entire length of two sequences, while the latter aligns only the most similar regions of two sequences.

The aim of this work is to apply the global and local alignment besides implementing the parallel computing to speed up the process of comparing sequences without compromise with incomplete results, like missing some optimal results. All problems are analysed and studied its characteristic as well as to choose the significant parameters that affect the result. The research methodology as depicted in Figure 2 is designed to produce an effective optimal alignment by reducing the gaps, mismatch and length of sequences optimal alignment without sacrificing the result of accuracy, efficient in time and space complexity, and practical for sequence alignments of large genomic regions. This work designs two proposed algorithms highlighted as, Needle (Needleman-Wunsch algorithm) and Smith (Smith-Waterman algorithm) programs. The programs aligning sequence 1, seqA with lenA length (subject sequence) against sequence 2, seqB with lenB length (query sequence). To implement the algorithms, a size of matrix (lenA x lenB) is calculated.

**NeedleArray[i][j] = max { NeedleArray [i −1][j −1] + sub (seqA [i], seqB [j] ), NeedleArray [i −1][ j] + del (seqA [i] ), NeedleArray [i][ j −1] + ins (seqB [j] ) }**

**SmithArray[i][j] = max { 0 , SmithArray [i −1][j −1] + sub (seqA [i], seqB [j] ), SmithArray [i −1][ j] + del (seqA [i] ), SmithArray [i][ j −1] + ins (seqB [j] ) }**