International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 6 | ISSN : 2456-3307

Improvisation of Global Pairwise Sequence Alignment Algorithm Using Dynamic Programming

Jyoti Lakhani[1], Ajay Khunteta[2], Dharmesh Harwani[*3]
[1]Poornima University, Jaipur & Maharaja Ganga Singh University, Bikaner, Rajasthan, India
[2]Poornima University, Jaipur, Rajasthan, India
[*3]Maharaja Ganga Singh University, Bikaner, Rajasthan, India

ABSTRACT

The global pairwise sequence alignment algorithms based on dynamic programming match each base pair step by step in the sequence under observation from start to end. This approach increases the time complexity which increases further many folds when large sequences are used. Needleman-Wunsch, Smith-Waterman, ALIGN, FASTA, BLAST and many other pair-wise sequence alignment algorithms are based on dynamic programming approach. The present communication is an attempt to provide a method for improvisation in dynamic programming used for pair-wise sequence alignment. The proposed technique is based on look-ahead method which decides whether it is required to continue or stop the processing of alignment steps for the sequence pair under observation from the current point if significant match score is not achieved. A threshold to be set by a user a-priori, indicating minimum percent match per base error to be accepted in sequence alignment process. The present improvisation method of dynamic programming can reduce bulky computational steps and hence save a reasonable amount of time in pairwise sequence alignment process.

Keywords: Improvisation, Pairwise Sequence Alignment, Dynamic Programming, Time Complexity

I. INTRODUCTION

Dynamic programing based sequence alignment becomes more complex as each base pair of the targeted sequences is compared step by step to align. On one side dynamic programming is complicated to apply due to the high number computational steps and on other side it is an important technique to find the best possible match. Other techniques find the optimal solution by compromising the best one. The present communication is an attempt to improvise the dynamic programming approach to find the best suitable par-wise sequence alignment by using a novel look-ahead method. The proposed method removes the unnecessary processing steps of global pairwise sequence alignment.

II. PREVIOUS WORK

Dot Matrix method (A.J. Gibbs and G.A. McIntyre (1970) is the simplest method for sequence matching but it is unable to reveal the best match for the two sequences where insertions and deletions are required. It is difficult to find the best sequence alignment by considering each possible pair-wise match with insertions and deletions. “To find an optimal alignment in which all possible matches, insertions and deletions have been considered to find the best one is computationally so difficult that for proteins of length 300, 1088 comparisons have to be made” (Waterman, 1989, Durbin 1998). Smith and Waterman presented a classical dynamic programming based global pairwise sequence alignment algorithm (1981a,b). Needleman and Wunsch (1970) used progressive building of alignment by comparing two amino acids at the same time. They started at one end of each sequence and then moved ahead one amino acid per pair. The approach is known as a global alignment using dynamic programming in which entire sequence is considered. Smith and Waterman (1981 a,b) identified that the best align sequence regions in DNA and protein sequences are more significant than the other less aligned regions.

CSEIT1726243 | Received : 25 Nov 2017 | Accepted : 20 Dec 2017 | November-December-2017 [(2)6: 909-914]

For this purpose they modified the Needleman-Wunsch algorithm for alignment of sequences locally. The modified version of Needleman-Wunsch algorithm is called local alignment or Smith–Waterman algorithm. In connection to the above, the Smith-Waterman algorithm also considered insertions or deletions that appeared during the evolutionary process. Finally, Smith Waterman algorithm provided a mathematical proof that the dynamic programming provides an optimal alignment between sequences for sure. As per Durbin (1998), dynamic programming methods are slow due to the large number of computational steps, which increase approximately proportional to the square or cube of the sequence lengths. Therefore, it is difficult to use this method for very long sequences.

There are some alternative methods that have greatly reduced the time and space requirements of dynamic programming method for sequence alignment. Some shortcut methods have also been developed to speed up the alignment process. Such methods are used in FASTA and BLAST algorithms. The word and k-tuple methods are used by FASTA and BLAST algorithms. FASTA was developed by W. Pearson and D. Lipman (1988) which performs a database scan for sequence similarity in a short time. FASTA break down a sequence into short words of a few characters long, and these words are then organized into a table indicating where they are in the sequence. If one or more words are present in both sequences, then the sequences must be considered similar for those regions. Pearson (1990, 1996) continued to improve the FASTA method for similarity searches in sequence databases. BLAST developed by S. Altschul et al. (1990) has been considered faster than the FASTA algorithm. Like FASTA, BLAST prepares a table of short sequence words for each sequence, but it also determines which of these words are most significant and good indicators of similarity between two sequences and finally it confines the search to these words (and related ones). This confinement fastened the alignment process. Recent improvements in BLAST include PSI-BLAST and GAPPED-BLAST which is threefold faster than the original BLAST.

III. IMPROVISATION METHOD

Improvisation of dynamic programming algorithm is an approach where computational steps have been reduced for pairwise sequence alignment process and performed to find the best sequence match. A threshold is set for sequence alignment process which indicates minimum percent base pair match error to be accepted for pair-wise sequence alignment.

The dynamic programming approach searches each possibility of alignment in order to search the best solution. Different algorithms omit some of the steps (possibilities of alignments) by setting threshold or by implementing word search e.g. BLAST. Although it is a time consuming approach but dynamic programming is useful as it can find the best possible pair-wsie sequence alignment rather than by just giving an optimal solution.

In the present communication we are presenting an improvisation method of dynamic programming approach for pairwise alignment of two sequences. The 100 percent match of two sequences under observation is the best alignment. But, it is not always possible to get 100 percent match therefore an error threshold is placed which indicates percent mismatches ignorable during the matching process. Let us assume that error threshold is 20 percent. Then it indicates that error up to 20 percent of the sequence length in the sequence alignment is accepted. Suppose sequence1 and sequence2 are of same length, indicated by n. The sequence pair span is from 0 to n. Threshold Th is the limit of error that can be ignorable in pair-wise sequence alignment process. Mf (Match factor) is an indication for the required percentage for matching in order to declare two sequences similar. It is required to set error threshold Th before the process of alignment is getting started. Match factor Mf can be calculated by formula Mf =(n-(n*Th/100)). Sp is the (Shift parameter) can be positive or negative. Shift parameter indicates shifting of sequence 2 with reference to sequence 1, in order to align the both sequences.

By implementing all shifting combinations total 2n combinations can be found. Each of these combinations is processed for alignment purpose. If shift parameter is a non –zero value, it is an indication of a shrink in the sequence pair. The reason is that if a sequence is shifted, the singleton end trails are no more paired to each other and there is no way to align them. In that case, these end trails of sequence does not keep importance and can be removed directly. It is clear here that as these end trails cannot be paired with other bases in sequence hence the deletion of this trail will not affect results. The end trails are shown in blue color in Figure 1.

If the end trails of sequence are not considered this results in shrink of sequence pair length to n-sf (Shift factor) for each combination. Here sf is considered as an absolute value. Assuming that if it requires one unit of time to compare one base pair for matching, then by removing end trails total 2n*(n-(n-sf)) units of time are saved.

Let us assume that there are two sequences with length n=10 and Th=20. Subsequently it requires 80 percent matches (Mf=8) or at-least 8 base pair match, out of 10 to declare the sequence pair are aligned. So, all possible sequence pair with n-sf < Mf can be discarded directly without processing further. This is shown in the yellow triangle in Figure 1(a) highlighting the ignored sequence base pairs for alignment process. Now, a total of 2*(n-Mf) number of sequence will be considered for alignment process and remaining 2n-(2*(n-Mf)) number of sequence can be discarded without processing further. 2n-(2*(n-Mf)) number of sequence has been shown below the yellow triangle in Figure (a) that can be discarded directly. In this case, only 2*(n-Mf) sequence pairs are considered for further processing. Here 2n-(2*n-Mf)) unit of computation time can be saved.

During the process of matching, it is possible to keep track of number of matches at a given point of time. This tracking record can be used to predict the possible sequence match. The prediction can be made by calculating matching score at that particular point, length of the sequence remaining for matching process (remaining sequence length rsl) and the number of required matches i.e. matching factor. Suppose both sequences are of length n=10 and we are at i^{th} position in pair-wise sequence alignment process and has obtained matching score ms =4. After the point i=5, rsl= n-i = 5 bases of sequence remain which are yet to be processed. If matching factor is mf = 8 then we require (mf - ms) match score to declare same sequence pair. This will be called remaining match score (rms) which is 4 here. In this case match score 8 will be needed and we have score 4 at the current position i, so we four more matches will be required. We still have rsl=5 length of sequence to be matched further. If there are 4 more matches then sequence can be declared as similar. The possibility to find similarity if rsl-rms >= 1 then prob=1 otherwise prob=0 will be considered. If prob is 1 then the matching process will continue. At the point where prob become 0, the matching process will stop. Now suppose n=10, ms=3, i=7, rsl= n-i = 10-7=3, rms= Mf-ms= 8-3=5. Here, at point i=7 match score is only 3 and length remained = 3. If all the three bases are getting matched, the match score will be ms+3 =6. There is now, no possibility to get the required match score i.e. 8. Consequently, the sequence matching process will jump down or stopped immediately.

IV. METHODOLOGY

A. Algorithm

```
Step 1. Set Look Ahead Pointer *l at 0th position.
Step 2. For Look Ahead Pointer *l from 0 to n
        - calculate ms
        - calculate rsl
        - calculate rms
        - calculate probability of matching
            if rsl < rms then
                probability = 0
            else
                probability = 1
Step 3. If probability of matching is 0 then stop the process otherwise continue.
```

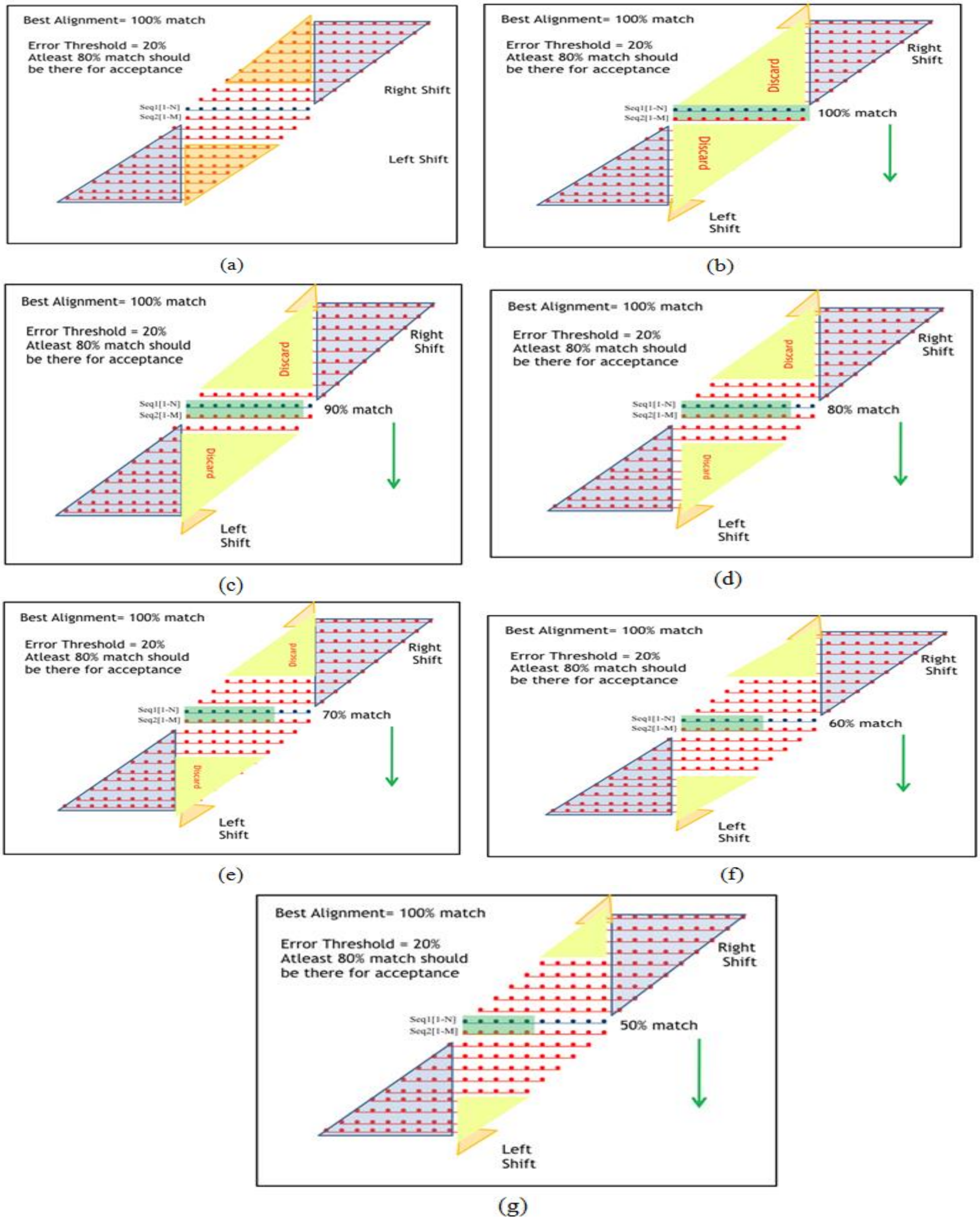

Figure 1. Improvisation of Dynamic Programming

1(a). Shifting of base pairs (left or right shift) until query and target pairs get aligned. Un-aligned base pairs are shown in blue triangle. As per threshold, 80 percent match is required hence sequence given in orange colour at both sides can be omitted. 1(b) to (g) Possibilities to omit steps of matching base-pairs during shifting. Since a 20 percent error threshold of will be accepted and if there is a perfect match at 10th base-pair, all other possibilities of sequence alignment steps could be omitted. As shown in (c), 9 out of 10 base pairs are already matched then remaining steps in sequence alignment can be omitted. All omitted steps of sequence alignment not to be considered or processed using the present method are shown in yellow colour in the 1(b)-(f).

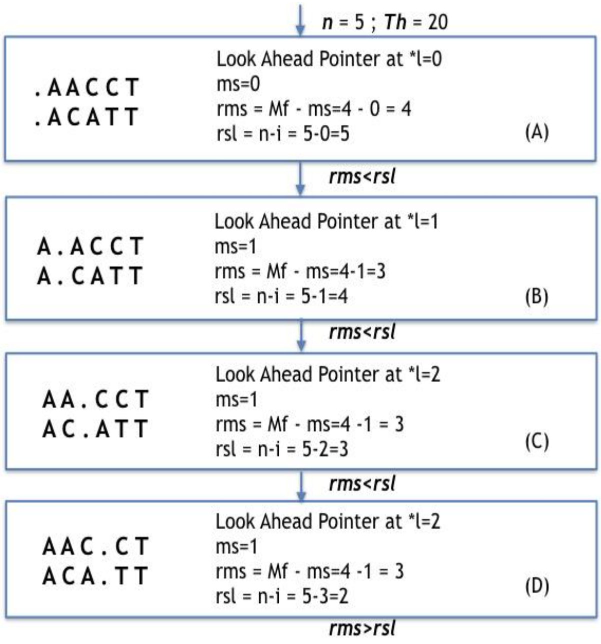

Figure 2. Improvisation in Dynamic Algorithm

Abbreviations used in Algorithm

length of sequence = n

Sequence pair span = 0-n

Threshold limit of error = Th

Match Factor = Mf

MF = n-(n*Th/100)

shift parameter = sp

Total possible Combination of sequences = 2n

Shrink factor= sf= absolute value of shift parameter

Shrink in sequence pair length by removing end trails=n-sf

Time saved by removing end trails= 2n*(n-(n-sf))

Sequences that can be discarded directly if n-sf<Mf

Number of sequences used for similarity search= 2*(n-Mf)

Number of sequences discarded= 2n-(2*(n-Mf))

Remaining match score = rms= Mf-ms

current position = i

Remaining sequence length = rsl= n-i

prob=if (rsl-rms >= 1) then prob=1 otherwise prob=0

The probability of finding a match at the particular point is calculated by tracking the match factor, match score, remaining sequence length and remaining match score. If there is a sufficient number of base pair match in the sequences under observation, the matching process will continue. Otherwise, the process of the pairwise sequence alignment will be stopped immediately. Hence the method proposed in the present communication will enhance the performance of pairwise sequence alignment by improvising dynamic programming algorithm.

In Figure 2, a look-ahead pointer *l (represented by a dot in figure) is used to predict the possibility to find a match between two sequences under observation with length *n* =5 and threshold Th=20. This means that 20 percent error (1 mismatch) can be ignored and remaining 80 percent sequence length should be matched and aligned with each other. In step (A) *l is at position 0 of the sequence. At this initial stage match score *ms* is 0 and the Match Factor Mf is 4 (Length of the Sequence - Th= 5-1=4). The required match score (rms) is 4 and remaining sequence length (rsl) is 5. As the rsl >=rms, the process will continue and *l will be moved to the next position (B). At B, ms=1, rms=3 and rsl=4. As *rsl* is still greater than the rms, possibility to find match is 1. The process will remain continue and *l will be moved to the next position (C). At stage (C) ms =1, rms=3 and rsl=3. Probability of finding a base pair match is still there if all remaining three base pairs get matched so the process will continue and position (D) will appear. At this stage ms is ms=1, rms=3 and rsl=2. Here, rsl<rms so the possibility to find match will be no more. So the process will stop automatically. Hence using the present method unnecessary steps in pairwise sequence alignment are not being processed and that results in the reduction of overall time complexity in pair wise sequence alignment.

V. CONCLUSION

Dynamic programming approach for sequence alignment provides the best solution. Due to the high computation cost, researchers are using other parallel techniques which help them to get the optimal solutions. In the present communication an attempt has been made to provide an improvisation method in dynamic programming for pairwise sequence alignment. The present method uses a look-ahead method and removes unnecessary processing steps in dynamic programming if appropriate base pair matches have not been met.

VI. REFERENCES

[1]. R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. "Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids",

Cambridge: Cambridge University Press (1998). DOI:10.1017/CBO9780511790492

[2]. A.J. Gibbs and G.A. McIntyre. "The diagram, a method for comparing sequences: Its use with amino acid and nuceotide sequences.” European Journal of Biochemistry (1970). 16, 1-11.

[3]. B. Needleman, Saul & D. Wunsch, Christian. "A General Method Applicable to Search for Similarities in Amino Acid Sequence of 2 Proteins” Journal of molecular biology(1970) 48. 443-53. DOI: 10.1016/0022-2836(70)90057-4.

[4]. W. R. Pearson and D. J. Lipman. "Improved tools for biological sequence comparison". Proceedings of the National Academy of Sciences of the United States of America(1988) 85 (8): 2444-2448. doi:10.1073/pnas.85.8.2444. PMC 280013. PMID 3162770

[5]. W.R. Pearson. "Rapid and sensitive sequence comparison with Fastp and Fasta”. Methods Enzymol (1990), 183: 63-98.

[6]. W.R. Pearson. "Effective protein sequence comparison”. Meth Enzymol. (1996) 266: 227-258.

[7]. S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and D. J. Lipman.(1990) J. Mol. Biol., vol. 215, pg. 403-410.

[8]. S. F. Altschul, T.L.Madden, A. A. Schaffer, et al., "Gapped BLAST and PSI- BLAST: A new generation of protein database search programs”. (1997) Nucleic Acids Res., Vol. 25, pp. 3389-3402.

[9]. T. F. Smith and M.S. Waterman. "Comparison of biosequences”. (1981a) Advances in Applied Mathematics Volume 2, Issue 4, December 1981, Pages 482-489. DOI: https://doi.org/10.1016/0196-8858(81)90046-4

[10]. T. F. Smith and M.S.Waterman. "Identification of common molecular subsequences". (1981b)Journal of Molecular Biology, Volume 147, Issue 1, 25 March 1981, Pages 195-197. DOI: https://doi.org/10.1016/0022-2836(81)90087-5