IJRIT International Journal of Research in Information Technology, Volume 1, Issue 11, November, 2013, Pg. 476-483

International Journal of Research in Information Technology (IJRIT)

www.ijrit.com ISSN 2001-5569

Partitioning Algorithm for Multiple Sequence Alignment

Meghna Mathur[1], Geetika[2]

[1]Student, CSE/IT, ITM University
Gurgaon, Haryana, India
meghna10047@gmail.com

[2]Assistant Professor, CSE/IT, ITM University
Gurgaon, Haryana, India
geetika@itmindia.edu

Abstract

Sequence alignment is an important task in bioinformatics which involves typical database search where data is in the form of DNA, RNA or protein sequence. For alignment various methods have been devised starting from pairwise alignment to multiple sequence alignment (MSA). To perform multiple sequence alignment various methods exists like progressive, iterative and concepts of dynamic programming in which we use Needleman Wunsch and Smith Waterman Algorithm. This paper proposes method based on partitioning algorithm for multiple sequence alignment and its results are also shown. The partitioning approach significantly improves the solution time and quality by utilizing the locality structure of the problem.

Keywords: **Multiple sequence alignment, partitioning algorithm**

1. Introduction

In bioinformatics, a sequence alignment is a way of arranging the biological sequences including DNA (Deoxyribonucleic acid) or RNA (Ribonucleic acid) or protein. DNA sequencing is used to find genes, segments of DNA that code for specific protein or phenotype. Also used if a region has been sequenced then it can be screened for characteristic features of genes.

Alignment can reveal homology between sequences. Similarity is the term that tells about the degree of match between two sequences. The principles include that sequence similarity do not always imply common function and conserved functions do not always imply similarity at sequence level.

Sequence alignments constitute an extremely powerful means of revealing the constraints imposed by the structure and function on the evolution of a protein and nucleic acid family, where as alignment task of multiple sequences requires large amount of computational time. Local alignment is based on completeness. It includes the task of finding and extracting a pair of regions from two given biological sequences that exhibit high similarity. Global alignment is also based on completeness and is done across the entire sequence length to include as many matches as possible up to and including sequence end whereas Pairwise Sequence Alignment is used to identify regions of similarity that may indicate functional, structural and/or evolutionary relationships between two biological sequences (protein or nucleic acid). This type of alignment is based on numbers. Multiple sequence alignment (MSA) is the alignment of three or more biological sequences of similar length and therefore it is included in the alignment based on numbers. From the output of MSA applications, homology can be inferred and the evolutionary relationship between the sequences can be studied.

2. Methods of Alignment

2.1 Dot Matrix Method:

A dot matrix analysis is primarily a method for comparing two sequences to look for possible alignment of characters between the sequences. The methods is also used for finding direct or inverted repeats in biological sequences and for predicting regions in RNA that is self-complementary and therefore have the potential of forming secondary structure through base-pairing. It works by locating regions of similarity between two sequences which provide a great deal of information about the function and structure of the query sequence. Similar structure indicates homology, or similar evolution, which provides critical information about the functions of these sequences. A dot matrix plot is a method of aligning two sequences to provide a picture of the homology between them. The dot matrix plot is created by designating one sequence to be the subject and placing it on the horizontal axis and designating the second sequence to be the query and placing it on the vertical axis of the matrix. At each position within the matrix, a point is plotted if the horizontal and vertical elements are identical. Diagonal lines within the resulting matrix indicate regions of similarity. The advantages of dot matrix methods is that it readily reveals the presence of insertions/deletions and direct and inverted repeats that are more difficult to find by the other, more automated methods. However it has some shortcomings also. These are that most dot matrix computer programs do not show an actual alignment. Also it does not return a score to indicate how 'optimal' a given alignment is [10].

2.2 Dynamic Programming*:*

Dynamic programming (DP) algorithms are a class of algorithms typically applied to optimization problems. DP is applicable on an optimization problem with following key points. Firstly it should have an optimal substructure – an optimal solution to the problem contains within it optimal solutions to sub-problems. Secondly it must contain overlapping sub-problems i.e. the pieces of larger problem have a sequential dependency. DP works by first solving every sub-sub-problem just once, and saves its answer in a table, thereby avoiding the work of re- computing the answer every time the sub-sub-problem is encountered. Each intermediate answer is stored with a score, and DP finally chooses the sequence of solution that yields the highest score [10].

2.2.1 Needleman Wunsch Algorithm for Global Alignment:

It performs a global alignment on two sequences. This algorithm is suitable when the two sequences are of similar length, with a significant degree of similarity throughout. It focuses on the best alignment over the entire length of the two sequences [1]. Steps performed are shown below:

1. Initialization: Create a matrix with X +1 Rows and Y +1 Columns
   The 1st row and the 1st column of the score matrix are filled as multiple of gap penalty

2. Scoring: The score of any cell C (i, j) is the maximum of:

   Scorediag = C (i-1, j-1) + S (i, j)

   Scoreup = C (i-1, j) + g

   Scoreleft = C(i, j-1) + g

   Where S(I, j) is the substitution score for letters i and j, and g is the gap penalty

3. Traceback: The trace back step determines the actual alignment(s) that result in the maximum score. There are likely to be multiple maximal alignments.Trace back starts from the last cell, i.e. position X, Y in the matrix gives alignment in reverse order. There are three possible moves: diagonally (toward the top-left corner of the matrix), up, or left .It takes the current cell and looks to the neighbor cells that could be direct predecessors. This means it looks to the neighbor to the left (gap in sequence #2), the diagonal neighbor (match/mismatch), and the neighbor above it (gap in sequence #1). The algorithm for trace back chooses as the next cell in the sequence one of the possible predecessors.

Example:

Sequence1: AGCTACTG

Sequence2: TCGCATACGC

- Match Score = +1
- Mismatch Score = -1
- Gap penalty = -1

Substitution Matrix

	A	G	C	T
A	1	-1	-1	-1
G	-1	1	-1	-1
C	-1	-1	1	-1
T	-1	-1	-1	1

Initialization step:

		A	G	C	T	A	C	T	G
	0	-1	-2	-3	-4	-5	-6	-7	-8
T	-1								
C	-2								
G	-3								
C	-4								
A	-5								
T	-6								
A	-7								
C	-8								
G	-9								
C	-10								

Scoring matrix:

		A	G	C	T	A	C	T	G
	0	-1	-2	-3	-4	-5	-6	-7	-8

T	-1	1	0	-1	-2	-3	-4	-5	-6
C	-2	0	2	1	0	-1	-2	-3	-4
G	-3	-1	1	3	2	1	0	-1	-2
C	-4	-2	0	2	4	3	2	1	0
A	-5	-3	-1	1	3	5	4	3	0
T	-6	-4	-2	0	2	4	6	5	4
A	-7	-5	-3	-1	1	3	5	7	6
C	-8	-6	-4	-2	0	2	4	6	8
G	-9	-7	-5	-3	-1	1	3	5	7
C	-10	-8	-6	-4	-2	0	2	4	6

The last right most cell gives the value of alignment of the sequences. Here the score is 6 .

Dynamic programming provides optimal alignment for a given set of scoring function which is its advantage. But it is slow due to the very large number of computational steps; computer memory requirement also increases as the square of the sequence lengths. Moreover the complexity is O (n^2).Therefore it is difficult to use the method for very long sequences.

2.2.2 Smith Waterman Algorithm for Local Alignment:

It performs the local alignment on two sequences and find out the best alignment over the conserved domain of two sequences [1]. The steps of Smith waterman algorithm are same as that of Needleman Wunsch i.e. initialization, scoring and trace back. In the initialization step we fill the first row and column of the matrix by zeros. In the scoring step we do not include negative scores so we put zeros instead. Finally in trace back step we start with the cell that has the highest score and work back until a cell with a score of zero is reached.

2.3 Progressive Methods:

Progressive methods are used for multiple sequence alignment. It is used to align three or more sequences. Using the standard dynamic programming algorithm on each pair, we can calculate the (N*(N-1))/2 (N is total number of sequences) distances between the sequence pairs. Distance matrix is obtained by applying the clustering algorithm and then constructing a guide tree. From the tree obtained we align the first node to the second node. After fixing the alignment, addition of another sequence or the third node takes place. After this we iterate the step until all the sequences are aligned. When a sequence is aligned to a group or when there is alignment in between the two groups of sequences, the alignment is performed that had the highest alignment score. The gap symbols in the alignment replaced with a neutral character [10].

2.4 Iterative Methods

These algorithms use iterative approach in which existing alignment can be realigned during addition of more sequences to multiple sequence alignment. Iterative methods can return to previously calculated pairwise alignments or sub-MSAs incorporating subsets of the query sequence as a means of optimizing a general objective function such as finding a high-quality alignment score [10]. Iterative methods are DIALIGN which takes an unusual approach of focusing narrowly on local alignments between sub-segments or sequence motifs without introducing a gap penalty. The alignment of individual motifs is then achieved with a matrix representation similar to a dot-matrix plot in a pairwise alignment. The most popular iteration-based method called MUSCLE (multiple sequence alignment by log-expectation) improves on progressive methods with a more accurate distance measure to assess the relatedness of two sequences. The distance measure is updated between iteration stages [6].

3. Evaluation of Sequence Alignment Algorithms

A sequence can be evaluated based on various factors like complexity of an algorithm, probability, accuracy and definiteness of an algorithm. Based on the complexity factor we can evaluate an algorithm .The algorithm that takes less time is more useful to the one that takes much time. This is so because our ultimate goal is always to solve a problem in as much less time as possible. In the case of ClustalW algorithm, the multiple sequence alignment of N sequences of length O (n), the complexity is O (N^2n^2).While by using the heuristics to this algorithm the complexity reduces to O ($N\log_2 Nn^2$). So it can be concluded that complexity helps in evaluating the use of an algorithm of a sequence alignment.
Second evaluating factor can be probability. Posterior probability is applied on pair of characters to find out the matching score. This helps in obtaining accurate results and higher speed [2].
The third evaluating factor can be accuracy of an existing algorithm. Accuracy can be defined as the correctness of an algorithm in terms of the output obtained on applying accurate inputs. An algorithm should always give one output to the number of inputs applied. So if that's the case then it can be said that the algorithm is accurate [3].
Fourth factor can be definiteness. By the term definiteness we mean that the algorithm should have finite number of steps. If an algorithm is not having a finite number of steps then that one cannot provide us with the correct results or the desired results.

4. Proposed Methodology

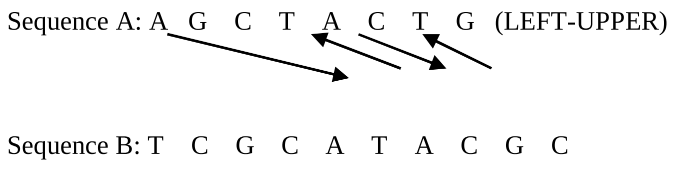
In partitioning algorithm the overall problem is partitioned into smaller sub problems that can be easily solved exponentially. Suppose we have a family S = (S1,S2,.......SN) of N sequences, we partition the sequences into subsets of segments vertically. When we cut the sequence Si at a position near the midpoint, we obtain two new families of shorter sequences. If we align the two new families of sequences optimally, we can concatenate our resulting alignments of the original sequences. [11]
Steps of the algorithm are:
1. Estimation of matching score
2. Searching for optimal cut-off points
3. Bisection
4. Partition.

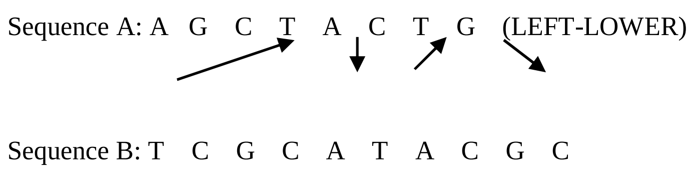
Estimation of matching score:
Let A and B be two sequences of lengths L and M. The score of their alignment can be found out using dynamic programming algorithm .To reduce the time complexity score estimation algorithm is designed to find approximate score of two sequence alignment. This algorithm consists of four steps which scans sequences from different directions. A and B are upper and lower sequences respectively whose scores are to be found out.

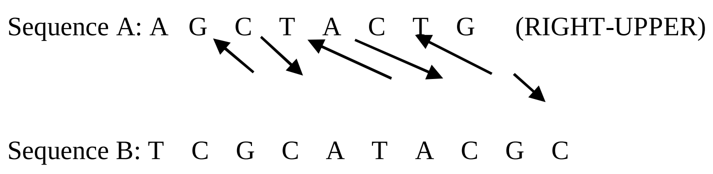
The steps are as follows:

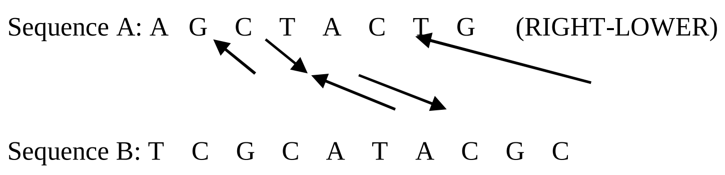
Left-Upper: in this step we start from first character in A sequence, suppose A[0] and search for the first matching character in B from left to right. If no matching character is found in B matching A[0] then we restart the scan to find for the character matching A[1] in B. If such character is found say B[j], the algorithm then searches for the first character matching with B[j] in the rest part of A from left to right. If no such character is found the it again start the search for the character matching B[j+1] in B. When such character is found say A[i], the algorithm searches for the first character matching with A[i] in rest of B sequence from left to right only. We keep on repeating these steps till we reach the end of the sequences. As a result we can find the matching characters in the sequences. Similarly we repeat the steps for Left-Lower, Right-Upper and Right-Lower.

Example: Explanation of first step


Sequence A: A G C T A C T G (LEFT-UPPER)
Sequence B: T C G C A T A C G C

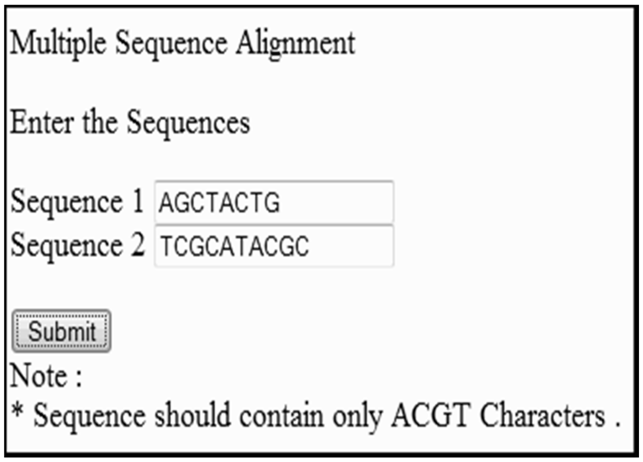

Sequence A: A G C T A C T G (LEFT-LOWER)
Sequence B: T C G C A T A C G C


Sequence A: A G C T A C T G (RIGHT-UPPER)
Sequence B: T C G C A T A C G C


Sequence A: A G C T A C T G (RIGHT-LOWER)
Sequence B: T C G C A T A C G C

Maximum count gives the score of alignment between sequences. Here maximum score is 6.

Implementation:



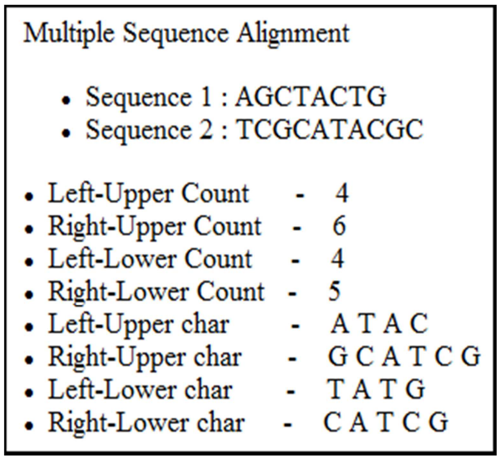
Multiple Sequence Alignment

Enter the Sequences

Sequence 1 AGCTACTG
Sequence 2 TCGCATACGC

Note :
* Sequence should contain only ACGT Characters .

Fig1. This figure shows the input taken in the form of sequence1 and sequence2.



```
Multiple Sequence Alignment

    • Sequence 1 : AGCTACTG
    • Sequence 2 : TCGCATACGC

• Left-Upper Count    -   4
• Right-Upper Count   -   6
• Left-Lower Count    -   4
• Right-Lower Count   -   5
• Left-Upper char     -   A T A C
• Right-Upper char    -   G C A T C G
• Left-Lower char     -   T A T G
• Right-Lower char    -   C A T C G
```

Fig2. Figure shows the output obtained giving the score of Left-upper,

Right-upper, Left-upper and Right-upper respectively.

5. Applications of Multiple Sequence Alignment

Multiple sequence alignment has emerged to have a lot of applications in the field of bioinformatics such as Sequence alignment helps in pattern recognition i.e. regions responsible for functional site can be identified by looking at the conserved regions, profiles can be extracted using the file provided in multiple sequence alignment so that these can be used against databases, DNA regulatory elements can be located using multiple sequence alignment moreover tree reconstruction can take place by picking u related sequences which are used in multiple sequence alignment. Also decision about membership of protein to belong to a particular family can be inferred from multiple sequence alignment.

6. Result and Conclusion

Based on literature survey of all sequence alignment algorithms the table gives the summarized observation of all algorithms.

Table1

ALGORITHM	ADVANTAGES	DISADVANTAGES
Dot Matrix Method	It readily reveals the presence of insertions/deletions and direct and inverted repeats that are more difficult to find by the other, more automated methods.	Dot matrix computer programs do not show an actual alignment. It does not return a score to indicate how 'optimal' a given alignment is. Dot plots do not provide statistical analysis.
Dynamic Programming i) Needleman Wunsch Algorithm ii)Smith Waterman Algorithm	It is guaranteed in mathematical sense to provide an optimal alignment for a given set of scoring function. It does not require gap penalty.	It becomes slow as there are large computation steps. The memory requirement also increases as alignment sequences get large. It requires gap penalty for efficient working.
Progressive Alignment	These are efficient for aligning large set of sequences. Progressive	Progressive alignment is not optimal

	alignment services are provided on web servers so the user need not install it locally.	globally. The errors generated at any stage are propagated to next stages which go till the end. When the sequences in the set are distantly related then the performance degrades.
Iterative Method i. DIALIGN ii. MUSCLE	They improve the accuracy of MSA. These can be repeated a number of times or until convergence.	

By studying various sequence alignment algorithms it can be concluded that posterior probability when applied on pairs of characters in a given sequence to avoid occurring of errors in early stage of progressive alignment helps in obtaining accurate results along with it if ant colony optimization technique is used to find matching score of character then higher speed can be achieved compared to other progressive alignment methods. ClustalW algorithm when applied on pair of sequences helps in reducing time complexity and also makes the process of alignment more dynamic.

7. References

1. Ankit Agrawal and Siddhartha Kumar Khaitan (2008),A New Heuristic for Multiple Sequence Alignment, *IEEE* Iowa State University, pp. 214-217.
2. Ling Chen, Welliu, Juan Chen (2007), Ant Colony Optimization Method for Multiple Sequence Alignment, Intl. Conf. Machine Learning and Cybernetics , Hong Kong, pp. 914-919.
3. Tristan Cazenve (2007), Overestimation for Multiple Sequence Alignment, in proceedings of the 2007 *IEEE* Symposium on Computational Intelligence in Bioinformatics and Computational *Biology*, University Paris 8, pp.159-164.
4. C. Notredame (2002), recent progress in multiple sequence alignment: a survey, *Pharmacogenomics*, **vol. 3**, pp. 131-144.
5. Carsten Kemena, C.Notredame (2009), Upcoming Challenges for multiple sequence alignment methods in high-throughput area, Oxford Journal, **vol.25**, issue no.19, pp.2455-2465.
6. Robert C. Edger (2004), MUSCLE: multiple sequence alignment with high accuracy and high throughput, Nucleic Acids Research, **vol. 32**, pp. 1972-1977.
7. Mohamed Radhouene Aniba, Olivier Poch and Julie D. Thompson (2010), Issues in bioinformatics benchmarking: the case study of multiple sequence alignment, Nucleic Acids Research, **vol. 38**, pp.7353-7363.
8. EMBL-EBI,WellcomeTrustGenome Campus,Hinxton,Cambridge,CBIO,1SD,UK[Online] Available: http://www.ebi.ac.uk/Tools/msa.
9. David Mount, Bioinformatics sequence and genome analysis, cold spring harbor laboratory press, pp.238-260.
10. "Multiple Sequence Alignment :Progressive methods and HMMS", Terry Speed's computational Biology Course at UC Berkeley.[Online]Available:http://www-stat.stanford.edu/~nzhang/345_web/sequence_slides3.pdf
11. Yixin Chen, Yi Pan, Ling Chen, Juan Chen. Partitioned optimization algorithms for multiple sequence alignment, Washington University, Georgia University, Yangzhou University.