ISSN:2229-6093

Multiple Sequence Alignment using Boolean Algebra and Fuzzy Logic: A Comparative Study

Nivit Gill, Shailendra Singh
Department of Computer Science and Engineering,
PEC University of Technology, Chandigarh
nivitgill@gmail.com, sscse@in.com

Abstract

Multiple sequence alignment is the most fundamental and essential task of computational biology, and forms the base for other tasks of bioinformatics. In this paper, two different approaches to sequence alignment have been discussed and compared. The first method employs Boolean algebra which is a two-valued logic whereas the second is based on Fuzzy logic which is a multi-valued logic. Both the methods perform sequence matching by direct comparison method using the operations of Boolean algebra and fuzzy logic respectively. To ensure the optimal alignment, dynamic programming is employed to align multiple sequences progressively. Both the methods are implemented and then tested on various sets of real genome sequences taken from NCBI bank. The processing time for both the methods on these data sets have been computed and compared.

Keywords: Bioinformatics, multiple sequence alignment, Boolean algebra, fuzzy logic.

1. Introduction

In biology, it is a widely accepted theory that all living organisms evolved from a single cell. The living organism cells are composed of genetic codes which are passed from one generation to other. The genetic code can be represented as a sequence of alphabets, such as four base pairs of DNA and RNA, or twenty amino acids of protein. These sequences are called biological sequences, and over time, a lot of changes, called mutations, occur in these sequences. The field of bioinformatics aims to align a large number of biological sequences with the purpose of deriving their evolutionary relationships through comparative sequence analysis.

The field of bioinformatics aims to discover and record the role of genetics in an organism's biological characteristics. The bioinformatics applies computations to the biological sequences in order to analyze and manipulate them. Sequence alignment is the most basic and essential module of computational bio-informatics and has varied applications in sequence assembly, sequence annotation, structural and functional prediction, evolutionary or phylogeny relationship analysis. Sequences with high degree of similarity have similar structure and function, and such sequences are useful in deducing evolutionary or phylogenetic relationships among organisms.

In this paper, two approaches to biological sequence alignment have been compared. The first approach uses Boolean algebra, which is a logical calculus of truth values, i.e. 0 or 1, or truth or false. In this approach, the given DNA sequences are encoded into binary form, and then Boolean operators are applied to determine the percentage of matching of sequences. The second method employs Fuzzy logic, which is a form of multi-valued logic derived from fuzzy set theory. The given biological sequences are compared pair wise so as to determine the number of matches, and mismatches between them. Then these counts are fuzzified using fuzzy membership functions, and then fuzzified counts are put in an aggregate fuzzy function in order to find the fuzzy match value of the two sequences. Both the methods use the computed match value to align the sequences progressively according to the similarity. The most similar pair is aligned first and the rest of the sequences are then aligned to this aligned pair.

The outline of this paper is: Section 2 discusses the basics of sequence alignment and its types and Section 3 reviews the literature related to sequence alignment. The methodology used for sequence alignment of two methods is discussed in detail in Section 4. Experimental results and their discussions are presented in Section 5 and finally Section 6 concludes the paper.

2. Biological Sequence Alignment

Biological sequence alignment is a field of research that focuses on the development of tools for comparing and finding similar sequences of amino acids or DNA base pairs with the help of computers. The degree of similarity is used to measure gene and protein homology, classify genes and proteins,

predict biological function, secondary and tertiary protein structure, detect point mutations, construct evolutionary trees, etc.

A biological sequence is a sequence of characters from an alphabet. For DNA sequence, character alphabet is {A, C, G, T}, for RNA sequence, alphabet is {A, C, G, U}, and for protein sequence, character set is {A, R, N, D, C, Q, E, G, H, I, L, K, M, F, P, S, T, W, Y, V}. A sequence alignment is a method of arranging biological sequences in order to search similar regions in the sequences. These similar regions provide functional, structural, and evolutionary information about the sequences under study. Aligned sequences are generally represented as rows within a matrix. Gaps ('-') are inserted between the characters so that identical or similar characters are aligned in successive columns. Gaps are also called indels, as they represent insertion of a character in or a deletion of a character from a biological sequence [1].

Pair-wise sequence alignment is concerned with comparing two DNA or amino-acid sequences – finding the global and local "optimum alignment" of the two sequences. Based on differences between the two sequences, one can calculate the "cost" of aligning the two sequences by using replacements, deletions and insertions, and assign a similarity score. The global alignment attempts to match both the sequences to each other from end to end. A local alignment searches for segments of the two sequences that match well. The classical global alignment technique is the Needleman-Wunsch algorithm, which is based on dynamic programming. The Smith-Waterman algorithm is a general local alignment method also based on dynamic programming. Multiple sequence alignment aims to find similarities between many sequences. It is hard and less tractable than pair wise alignment. Dynamic programming is impractical for a large number of sequences.

In order to quantify the similarity achieved by an alignment, substitution matrices are used [1]. These matrices contain a value (positive, zero or negative value) for each possible substitution, and the alignment score is the sum of the matrix's entries for each aligned pair. For gaps (indels), a special gap penalty score is used--a very simple one is just to add a constant penalty score for each indel. The optimal alignment is the one which maximizes the alignment score. Commonly used matrices are PAM (Percent Accepted Mutations) matrices, BLOSUM (BLOck SUbstitution Matrix), etc.

3. Literature Review

Biological sequences databases are growing exponentially resulting in extensive demands on the implementation of new fast and efficient sequence alignment algorithms. Most of the work in the sequence alignment field has been primarily intended on providing new fast and efficient alignment methods.

Smith and Waterman proposed an algorithm to find a pair of segments one from each of two long sequences such that there is no other pair of segments with greater similarity (homology) [12]. In this local alignment algorithm, similarity measure allowed arbitrary length deletions and insertions. A new algorithm for local alignment of DNA sequences had been proposed by Das and Dey [11]. A dynamic programming algorithm for performing a global alignment of two sequences has been proposed by Needleman and Wunsch [8]. A partitioning approach, based on ant-colony optimization algorithm has been proposed by Y. Pan et.al; the approach significantly improves the solution time and quality by utilizing the locality structure of the problem [19]. Naznin, Sarker and Essam designed an iterative progressive alignment method for multiple sequence alignment by using new techniques for both generating guide trees for randomly selected sequences as well as for rearranging the sequences in the guide trees [2]. Cai, Juedes, and Liakhovitch proposed to combine existing efficient algorithms for near optimal global and local multiple sequence alignment with evolutionary computation techniques to search for better near optimal sequence alignments [4]. Anitha and Poorna suggested an algorithm for global alignment between two DNA sequences using Boolean algebra and compare the performance of the algorithm with Needleman-Wunsch algorithm [13]. Yue and Tang applied the divide-and-conquer strategy to align three sequences so as to reduce the memory usage from O (n^3) to O (n^2) [3]. They used dynamic programming so as to guarantee optimal alignment. Nasser et al. provided a hybrid approach of dynamic programming and fuzzy logic to align multiple sequences progressively [10]. They computed optimal alignment of subsequences based on several factors such as quality of bases, length of overlap, gap penalty. Bandyopadhyay et. al proposed direct comparison methods to obtain global and local alignment between the two sequences; the method proposed an alternate scoring scheme based on fuzzy concept [9]. Chang et al. established fuzzy PAM matrix using fuzzy logic and then estimated score for fitness function of genetic algorithm using fuzzy arithmetic [7]. Their experimental results evidenced fuzzy logic useful in dealing with the

uncertainties problem, and applied to protein sequence alignment successfully.

The sole aim of the researchers has been to develop efficient alignment algorithms based on different and latest techniques.

4. Methodology

The methodology used for aligning multiple sequences is first determining the similarity between the sequences, and then based on their degree of similarity, the sequences are aligned. In this paper, the two sequence matching approaches being compared, firstly find the similarity values for all pairs of involved multiple sequences using their respective procedures. Based on this computed similarity values, both the approaches then align the multiple sequences in a progressive manner using Needleman-Wunsch algorithm for pair-wise alignment of sequences.

4.1 Sequence Matching

Sequence matching refers to computing the degree of similarity or dissimilarity between two sequences. This measure can be used to guide sequence alignment of multiple sequences, searching in databases to find the best match, and deriving evolutionary relationships between the sequences. Different methods are used to measure the similarity / dissimilarity such as calculating the number of matches (identities), the number of gaps or mismatches, score as number of matches minus the number of gaps/ mismatches, etc.

4.1.1. Boolean Logic Approach.

Boolean algebra (or Boolean logic) is a logical calculus of truth values (true or false), developed by George Boole in the 1840s. The operations in Boolean algebra are conjunction xΛy (AND), disjunction xVy (OR), and complement or negation ¬x (NOT) [15].

The Boolean logic approach to sequence matching translates the given pair of biological sequences (DNA, for example) into their binary forms, so that Boolean logic can be applied to them. The four nucleotides A, C, G and T are represented by 000, 001, 010, and 011 respectively, and the gaps as 100. Exclusive NOR (XNOR) function (see Table I) is a Boolean operator that produces true if both the inputs are same, otherwise false [16]. XNOR function is applied on two sequences encoded as binary strings. In the resultant string, replace the three consecutive ones by 1, otherwise replace by 0. Thus, in the final resultant string, 1 will correspond to a match and 0 to a mismatch.

Table 1. XNOR gate

A	B	A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

The Boolean approach can be implemented in two steps – first encoding the given biological sequences into binary form (toBinary() procedure) and then calculating the percentage of similarity by applying Boolean operations on the binary encoded biological sequences (BA_SequenceMatch() procedure).

pseudo code for toBinary(SeqA)

```
// input: character sequence SeqA
 // output: binaryA, the binary form of SeqA
i. initialize binaryA = [ ]                 //empty string
ii. for each character m in the sequence SeqA
     if (m == 'A' || m=='a') suffix=[0 , 0 , 0]
     elseif (m=='C' || m=='c') suffix=[0 , 0 , 0]
     elseif (m=='G' || m=='g') suffix=[0 , 0 , 0]
     elseif (m=='T' || m=='t') suffix=[0 , 0 , 0]
     else  suffix=[1 , 0 , 0]
     endif
iii. append suffix to end of binaryA
iv. return binaryA
```

pseudo code for BA_SequenceMatch(SeqA, SeqB)

```
// input: character sequences SeqA and SeqB
//output: the percentage of similarity of given sequences SeqA and SeqB
 i. binaryA = toBinary (SeqA)
 ii. binaryB = toBinary (SeqB)
 iii. compute result = XNOR (binaryA, binaryB)
 iv. initialize final-result = [ ]        // empty string
 v. for i = 1 to length (result), step-increment=3
      if (result[i] == result[i+1]== result[i+2] == 1)
                    append 1 to final-result
    else
                    append 0 to final-result
    endif
 vi. compute count = number of 1's in final-result
 vii. match-percent = (count * 100) / length (final-
          result)
 viii. return match-percent
```

4.1.2. Fuzzy Logic Approach.

Fuzzy logic is a form of multi-valued logic that deals with reasoning that is approximate rather than fixed and exact. In contrast with "crisp logic" i.e. Boolean logic, where binary sets have two-valued logic: true or false, fuzzy logic variables may have a truth value that ranges in degree between 0 and 1

[17]. The operations for OR and AND operators are max and min, respectively. For complement (NOT) operation, NOT (A) is evaluated as (1-A) [14].

The Fuzzy logic approach to sequence matching uses three input variables – match-count (#match), mismatch-count (#mismatch), and calculated-score (#score – calculated using substitution matrix). These inputs are then fuzzified using following membership functions:-

μ(match) = { 0, if #match=0
1, if #match=lenSeq -- (1)
[0,1] (#match / lenSeq)
}

μ(mismatch) = { 0, if #mismatch=0
1, if #mismatch=lenSeq -- (2)
[0,1] (#mismatch/ lenSeq)
}

μ(score) = { 0, if #score <=0
1, if #score = perfectScore -- (3)
[0,1] #score / perfectscore
}

In these equations 1, 2 and 3, lenSeq is the length of the shorter sequence of the two sequences being matched, and perfectScore is the score of matching the two candidate sequences, if there are no indels or replacements.

These Fuzzy values are put into an aggregate Fuzzy function, Fuzzy-match-score, given in equation 4.

Fuzzy-match-score = W1 * μmatch + W2 * μmismatch + W3 *μscore ---- (4)

The weights W1, W2, and W3 are the fixed weights assigned to three Fuzzy measures respectively. The Fuzzy logic approach has been implemented using FLSequenceMatch() procedure.

pseudo code for FL_SequenceMatch(SeqA,SeqB)

// input: character sequences SeqA and SeqB
//output: the fuzzy similarity measure of given sequences SeqA and SeqB

i. initialize matchcount=0, mismatchCount=0, score=0, I=0
ii. compute len=min(length(A), length(B))
iii. a. for I = 1 to len
if (A[I] == B[I])
matchCount=matchCount+1
else
mismatchCount=mismatchCount+1
endif
b. compute s=SubstitutionMatrix(A[I],B[I])
c. score= score + s
iv. compute three fuzzy similarity measures using equations 1, 2, and 3
v. compute fuzzy-match-score using aggregate fuzzy function given in equation 4
vi. return fuzzy-match-score

4.2 Multiple Sequence Alignment

Both the methods align the multiple sequences using progressive approach, i.e. first aligning the most similar pair then aligning another sequence to the aligned pair and so on until all the sequence in the given set are aligned. The sequence once aligned is not changed afterwards rather new sequences are aligned to the already aligned ones. The pair-wise alignment is being done using the traditional Needleman-Wunsch algorithm.

4.2.1. Progressive Alignment of Multiple Sequences.

In order to align multiple sequences in a progressive fashion, the sequences are compared pair-wise to find the degree of similarity between all the pairs of sequences. From these computed similarity values, find the most similar pair. This most similar pair of sequences is then aligned using Needleman-Wunsch algorithm. These two aligned sequences are taken as reference alignment to which all other remaining sequences are aligned. Now, from the remaining sequences find a sequence which has got the highest similarity with the already aligned sequences, and align it to the reference sequences using Needleman-Wunsch algorithm.

To align multiple sequences progressively, procedure progressiveSequenceAlignment() has been used.

pseudo code for progressiveSequenceAlignment (SeqFile, N, mode)

// input: a FASTA file SeqFile, number of sequences, N, mode =1 or 2
// output: a FASTA file of aligned sequences – alignedSeqDB

i. read the sequences from SeqFile using 'readfasta' function into SeqDB
ii. if (mode == 1)
MATCHER = BA_SequenceMatch
else
MATCHER = FL_SequenceMatch
iii. initialize SimilarityMatrix as a matrix of zeroes of order N
iv. for i = 1 to N-1
for j = i+1 to N
calculate match-value=
MATCHER(SeqDB{i}, SeqDB{j})
SimilarityMatrix [i, j] = match-value
v. find I and J such that SimilarityMatrix[I, J] has the maximum value in the matrix.

vi. [RefA, RefB] = NWalignment(SeqDB{I}, SeqDB{J}). Store these aligned sequences in alignedSeqDB.
vii. find the next sequence K from the remaining sequences to be aligned
viii. KA = NWAlignment(K, RefA)
ix. KB = NWAlignment(K, RefB)
x. compute
    a. matchKA=MATCHER(RefA, KA)
    b. matchKB=MATCHER(RefB, KB)
xi. if(matchKA>matchKB)
    store aligned sequence KA in the alignedSeqDB
    else
    store aligned sequence KB in the alignedSeqDB
xii. Repeat the steps vii - xi until all the sequences in SeqDB have been aligned

4.2.2. Needleman-Wunsch Algorithm.

The Needleman-Wunsch algorithm is a classical dynamic programming- based algorithm for global alignment of two biological sequences. This algorithm first calculates a scoring matrix for the two given sequences A and B, by placing one sequence along row side and another column side [18]. The size of the matrix is (M+1)*(N+1) (M and N are the lengths of the two sequences). The optimal score at each matrix (i, j) position is calculated by adding the current match score to previously scored positions and subtracting gap penalties, which may evaluate to either a positive, negative or 0 value.

A matrix F(i, j) indexed by residues of each sequence is built recursively, such that

$$F(i, 0) = F(0, j) = 0$$

$$F(i, j) = \max \{ F(i-1, j-1) + S(xi, yj), \; F(i-1, j) + G, \; F(i, j-1) + G \} \quad \text{--- (5)}$$

subject to boundary conditions; here, S(i, j) is the substitution score for residues i and j, and G is the gap penalty.

An alignment is computed using the F-matrix (calculated above): start from the bottom right cell, and compare the cell value with the three possible sources ((i-1, j-1) i.e. a Match, (i, j-1) i.e. an Insert, and (i-1, j) i.e. a Delete) to see which it came from. If it is same as Match, then Ai and Bj are aligned, if same as Delete, then Ai is aligned with a gap, and if same as Insert, then Bj is aligned with a gap.

5. Experiments, Results and Discussion

Both the methods have been implemented using MATLAB as it provides toolboxes for bioinformatics as well as Fuzzy logic. To apply Fuzzy logic to match two sequences, we have designed a Fuzzy inference system named 'FLSeqMatcher.fis' using the GUI – based MATLAB Fuzzy Logic Tool [5]. The fuzzified input variables – matchP, mismatchP, and score are supplied to the FIS. The FIS then calculates the corresponding Fuzzy match value based on the defined rules. Figure 1 gives the overview of the Fuzzy inference system designed for implementing the Fuzzy logic sequence matching. The Figure illustrates the input variables, output variable, number and type of member functions of each variable and number of rules in the system.

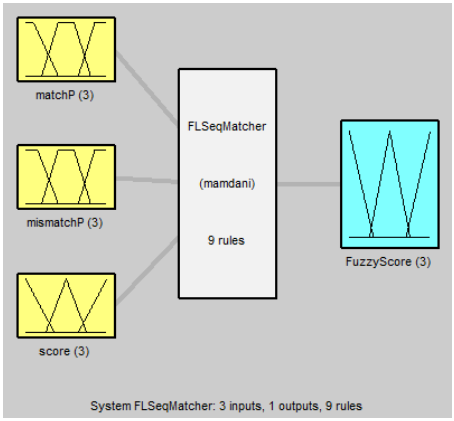

Figure 1. Overview of Fuzzy inference system FLSeqMatcher

To evaluate and compare the performance of the two approaches – Boolean and Fuzzy logic, two sets of DNA sequences have been used. Data set I is used to find out the effect of varying the number of sequences being aligned on the processing time of the two approaches; and data set II is used to determine the effect of varying the length of sequences being aligned on the processing time of two approaches. In both the data sets, three categories of genome sequences of three influenza viruses AH3N8, AH1N1 and AH5N1 have been collected (randomly) from NCBI's Influenza virus resource site [6]. The three categories of data set I have been named A (AH1N1), B (AH5N1) and C (AH3N8). The category A of sequences' length is less than or equal to 600 base pairs. The category B sequences are 700–1100 base pairs long and finally, the length of category C sequences' lies in the range of 1500-1800 base pairs. In each category of data set I sequences, numbers of sequences considered are 10, 20, 50, 70, 100, 120 and 150. The data set II categories are named as D (AH3N8), E (AH1N1) and F (AH5N1). The number of sequences in the sets of category D is 20, the category E is 40 and the category F is 70. In all the three categories of data set II, sequences of six different length ranges have been taken. These length

ranges are - less than 600 bp, 601 - 900 bp, 901 – 1200 bp, 1201 – 1800 bp, and 1801 – 2400 bp.

To compare the amount of time needed to process the two methods of alignment being discussed, the processing time has been calculated using MATLAB's 'tic' and 'toc' functions, which respectively start and stop the timer and return the value of timer in seconds. Each sequence set of both the data sets has been aligned using both methods for fifty times and the execution times for all the fifty runs have been averaged. This average execution time has been used for the comparison. The average processing time for sequence sets of data set I and data set II are tabulated in Table 2 and Table 3 respectively. The average processing time over fifty runs of both the methods on the six categories – A, B, C, D, E and F of sequences of both the data sets are graphically illustrated with the help of the line graph in Figure 2, Figure 3, Figure 4, Figure 5, Figure 6 and Figure 7 respectively.

The experiments of data set I show the effect of variation of the number of sequences on the processing time of both the discussed alignment methods. The results of first category show that if the length of sequences is less than or equal to 600 bp, the Boolean method takes less time as compared to Fuzzy method even if the number of sequences being aligned is large. The second category sequences are of length in the range 700 – 1100 bp and the alignments of these sets take almost similar time with both the methods but as the number of sequences is increased the Boolean method consumes more processing time than the Fuzzy method. For the third category of sequences of length 1500 – 1800 bp, the processing time of Boolean method is higher than the Fuzzy logic method.

Table 2. Average Processing Time (in seconds) for category A, B and C DNA sequences of data set I

Number of Sequences	Category A: AH1N1 (Length<=600 bp)		Category B: AH5N1 (Length: 700 -1100 bp)		Category C: AH3N8 (Length: 1500-1800 bp)	
	Boolean Algebra Method	Fuzzy Logic Method	Boolean Algebra Method	Fuzzy Logic Method	Boolean Algebra Method	Fuzzy Logic Method
10	0.71	1.13	8.48	8.32	30.19	28.77
20	2.97	7.17	18.97	16.85	66.3	63.52
50	22.97	27.68	64.98	59.24	195.93	172.63
70	26.54	38.75	100.11	89.15	293.68	239.42
100	46.18	70.41	167.25	144.68	467.51	367.04
120	77.19	98.7	212.9	185.27	596.12	448.59
150	99.09	139.84	305.45	261.88	834.52	592.71

Table 3. Average Processing Time (in seconds) for category D, E and F DNA sequences of data set II

Length of Sequences (in bp)	Category D: AH3N8 (No. of Sequences = 20)		Category E: AH1N1 (No. of Sequences = 40)		Category F: AH5N1 (No. of Sequences = 70)	
	Boolean Algebra Method	Fuzzy Logic Method	Boolean Algebra Method	Fuzzy Logic Method	Boolean Algebra Method	Fuzzy Logic Method
<=600	3.44	9.45	7.56	10.44	17.63	35.29
601 – 900	17.31	16.9	26.45	25.59	73.82	69.14
901 – 1200	21.87	21.92	52.31	47.57	117.28	105.54
1201 – 1500	43.47	40.07	101.54	90.35	182.27	151.24
1501 – 1800	65.54	63.52	151.15	128.06	275.8	216.96
1801 – 2400	114.7	106.88	305.4	265.04	503.17	399.91

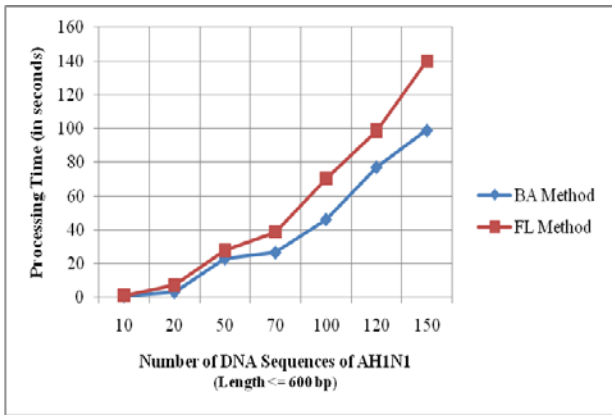

Figure 2. Line graph for average processing time of Boolean and Fuzzy methods on category A: AH1N1 DNA sequences

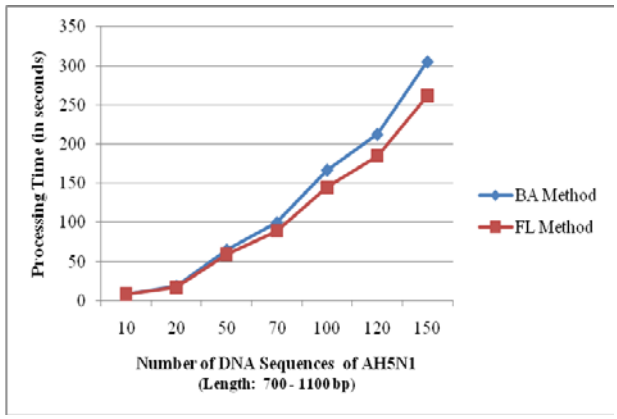

Figure 3. Line graph for average processing time of Boolean and Fuzzy methods on category B: AH5N1 DNA sequences

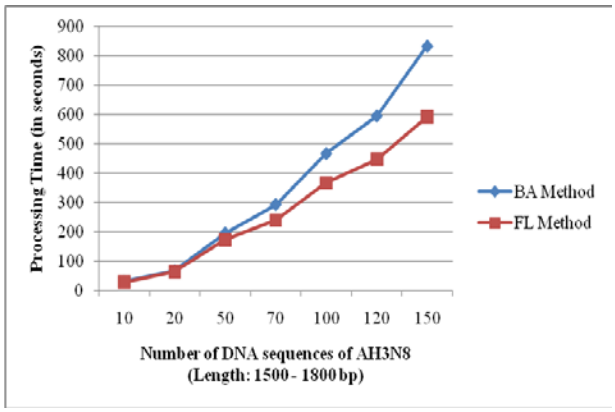

Figure 4. Line graph for average processing time of Boolean and Fuzzy methods on category C: AH3N8 DNA sequences

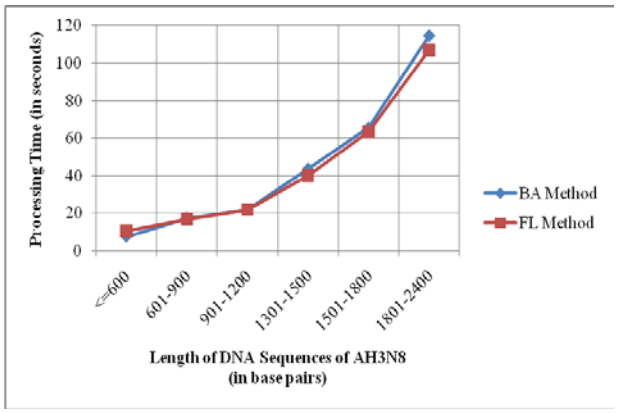

Figure 5. Line graph for average processing time of Boolean and Fuzzy methods on category D: AH3N8 DNA sequences

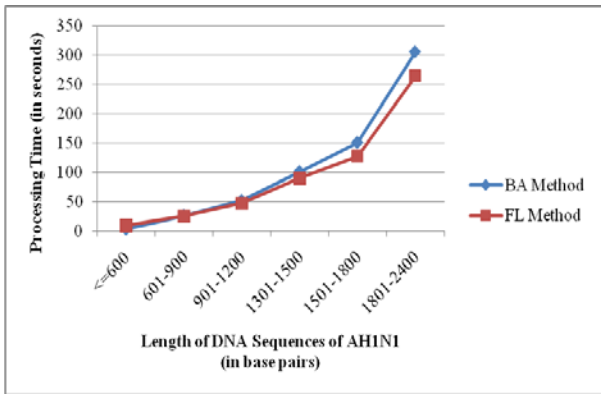

Figure 6. Line graph for average processing time of Boolean and Fuzzy methods on category E: AH1N1 DNA sequences

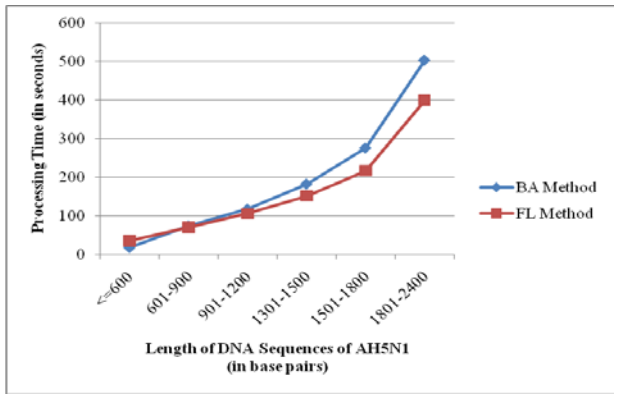

Figure 7. Line graph for average processing time of Boolean and Fuzzy methods on category F: AH5N1 DNA sequences

The results of experiments on data set II show the effect of varying the length of sequences on the processing time of alignment performed by both the methods. All the results show that if the length of sequences is less than or equal to 600 bp, the Boolean method takes less time than the Fuzzy method. The results of first category of sequences show that the time taken to align multiple sequences is almost same as the number of involved sequences is 20 only. The second and third category of sequences that contain 40 and 70 sequences respectively shows the increase in difference in execution time of the two methods with the increase in length of sequences. Boolean method takes more time than Fuzzy method as the length of sequences being aligned is increased.

The comparative analysis of the processing time of the two methods indicates that the Boolean method is very inefficient if the number and length of involved sequences is large. If the length of involved sequences is less than or equal to 600 bp then the Boolean method outperforms the Fuzzy method.

6. Conclusion

Two methods for sequence matching: Boolean algebra and fuzzy logic have been discussed, implemented and tested on real data sets. In the first method, the sequences were encoded into binary form and then using logical operators, the match value was determined. In the second method, number of matches, mismatches and the score for two sequences were determined, and then fuzzified so as to apply fuzzy logic to calculate the fuzzy similarity value. In both the methods, the match/ similarity value guides in the progressive alignment of multiple sequences, which was done using dynamic programming. The effect of varying lengths and numbers of sequences being aligned on processing time, averaged for fifty runs of the programs, of both the methods indicate that the Boolean method is efficient if the length of sequences is less than or equal to 600 base pairs, otherwise the Fuzzy logic method is better than the Boolean method. The comparative analysis of the two methods indicates that the Boolean method is very inefficient if the number and length of involved sequences is large. If the length of involved sequences is less than or equal to 600 bp then the Fuzzy method is inefficient in terms of processing time as compared to Boolean method.

References

[1] David W. Mount, “Bioinformatics: Sequence and Genome Analysis”, Cold Spring Harbor Laboratory Press.

[2] Farhana Naznin, Ruhul Sarker, and Daryl Essam , “Iterative Progressive Alignment Method (IPAM) for Multiple Sequence Alignment”, Computers & Industrial Engineering, 2009. pp. 536-541.

[3] Feng Yue and Jijun Tang, “A Divide-and-Conquer Implementation of Three Sequence Alignment and Ancestor Inference”, 2007 IEEE International Conference on Bioinformatics and Biomedicine, pp. 143-150.

[4] L. Cai, D. Juedes, E. Liakhovitch, “Evolutionary computation techniques for multiple sequence alignment”, *Proceedings of the 2000 Congress on Evolutionary Computation*, 2000, pp. 829-835.

[5] MATLAB Fuzzy Logic Toolbox. Website: http://www.mathworks.com/help/toolbox/fuzzy.

[6] NCBI Influenza Virus Data base. Website: http://www.ncbi.nlm.nih.gov/genomes/FLU/Database/nph-select.cgi.

[7] Pin-Teng Chang, Lung-Ting Hung, Kuo-Ping Lin, Chih-sheng Lin, Kuo-Chen Hung, “Protein Sequence Alignment Based on Fuzzy Arithmetic and Genetic Algorithm”, 2006 IEEE International Conference on Fuzzy Systems, pp. 1362-1367.

[8] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *J. Molecular Biology*, vol. 48, pp. 443-453, 1970.

[9] S.S. Bandyopadhyay, S.S. Paul, A. Konar, “Improved Algorithms for DNA Sequence Alignment and Revision of Scoring Matrix”, Proceedings of International Conference on Intelligent Sensing and Information Processing, 2005, pp. 485-490.

[10] Sara Nasser, Gregory L. Vert, Monica Nicolescu1 and Alison Murray, “Multiple Sequence Alignment using Fuzzy Logic”, Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, pp. 304-311.

[11] Swagatam Das & Debangshu Dey, “A new algorithm for local alignment in DNA sequencing”, *Proc. of IEEE Conference, INDICON 2004*, pp. 410-413.

[12] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequence," *J. Molecular Biology*, vol. 147, pp. 195-197, 1981.

[13] V. Anitha and B. Poorna, “DNA Sequence Matching using Boolean Algebra”, 2010 International Conference on Advances in Computer Engineering, pp. 212-216.

[14] Website: “A Short tutorial on Fuzzy Logic”, http://www.cs.bilkent.edu.tr/~bulbul/depth/fuzzy.pdf.

[15] Website: http://en.wikipedia.org/wiki/Boolean_algebra_(logic).

[16] Website: http://en.wikipedia.org/wiki/XNOR_gate.

[17] Website:http://en.wikipedia.org/wiki/Fuzzy_logic.

[18] Website:http://en.wikipedia.org/wiki/Needleman-Wunsch_algorithm.

[19] Y. Pan, Y. Chen, Juan Chen, Wei Liu, Ling Chen “Partitioned optimization algorithms for multiple sequence alignment”, Proceedings of the 20th International Conference on Advanced Information Networking and Applications, 2006, pp. 5.