Research Article

AN ITERATIVE APPROACH FOR EVALUATING THE ALIGNMENT QUALITY OF MULTIPLE BIOLOGICAL SEQUENCES

[*]Manish Kumar and Haider Banka
Department of CSE, Indian School of Mines, Dhanbad, India
** Author for Correspondence*

ABSTRACT

Biological sequence processing is a key of information technology for molecular biology. This scientific area requires powerful computing resources for exploring large sets of biological data. This paper proposes few genetic operators to obtain better alignments of multiple molecular sequences. The goal of this study is to develop some effective and efficient genetic operators which can help in aligning different length DNA sequences in its best possible way. In this paper, a simulation study has been performed on DNA sequences with the help of genetic algorithm, to evaluate the performance of the proposed method with the Chen *et al.'s* method by calculating and comparing the fitness and match column scores. Experiments on real data sets (EMBL and GenBank) confirm that the new approach presented in this experimental study, has show an improvement in both alignment quality and resource requirements.

Keywords: *Bioinformatics, Multiple Sequence Alignment, Genetic Algorithm, Crossover Operator, Mutation Operator*

INTRODUCTION

Multiple sequence alignment is an optimization problem of bioinformatics that appears in many other diverse scientific fields (Hamidi *et al.,* 2013). A multiple sequence alignment (MSA) is a sequence alignment of three or more biological sequences, generally protein or Deoxyribonucleic acid **(**DNA) (Wei *et al.,* 2013) that is to be aligned as a set of sequences that allows identifying the regions where the sequences are similar and where they differ.

The multiple sequence alignment problems (MSA) in computational biology consists in aligning several sequences, e.g. related genes from different organisms, in order to reveal similarities and differences across the group. Either DNA can be directly compared, and the underlying alphabet ∑ consists of the set {C,G,A,T*}* for the four standard nucleotide bases cytosine, guanine, adenine and thymine; or we can compare proteins, in which case ∑ comprises the twenty amino acids.

For MSA, we try to write the sequences one above the other such that the columns with matching letters are maximized; thereby gaps (denoted here by an additional letter " - ") may be inserted into either of them in order to shift the remaining characters into better corresponding positions. Different letters in the same column can be interpreted as being caused by point mutations during the course of evolution that substituted one amino acid by another one; gaps can be seen as insertions or deletions (since the direction of change is often not known, they are also collectively referred to as indels). Presumably, the alignment with the fewest mismatches or indels constitutes the biologically most plausible explanation.

There is a host of applications of MSA within computational biology; e.g., for determining the evolutionary relationship between species, for detecting functionally active sites which tend to be preserved best across homologous sequences, and for predicting three dimensional protein structure.

In recent years, some methods have been presented for multiple sequence alignment (Ergezer *et al.,* 2004; Hunt *et al.,* 2003; Lin *et al.,* 2004; Stoye, 1998; Chellapilla and Fogel, 1999; Chellapilla *et al.,* 1999) presented a method for multiple sequence alignment using evolutionary programming techniques. Chen *et al.,* (2005), presented a method for multiple DNA sequence alignment based on genetic algorithms. Chin *et al.,* (2003) presented an efficient constrained multiple sequence alignment method with the guaranteed performance. Pietro *et al.,* (2003) presented a multiple sequence alignment method by using the antipole clustering and the linear approximate 1-median computation. Du and Lin 2004 presented parallel computation techniques for multiple sequence alignments. Edgar 2004, Edgar presented a multiple sequence alignment method with the improved accuracy and speed. Ergezer and Leblebicioglu (2004) presented a multiple sequence alignment method using the hidden Markov model. Hunt *et al.,* (2003) presented a linear programming based algorithm for multiple sequence alignment. Isokawa *et al.,* (1996) presented a method for dealing with multiple sequence alignment by employing a genetic algorithm (GA). Lin (2000) used genetic algorithms to deal with the multiple sequence alignment problem, where every randomly generated multiple sequence alignment was transformed into an individual chromosome, and a best chromosome was found by the GA for multiple DNA sequence alignment. Liu *et al.,* (2004) presented a method for aligning multiple sequences by employing genetic algorithms. Luo *et al.,* (2005) presented a method for parallel multiple sequence alignment with dynamic scheduling. Needleman and Wunsch (1970) used dynamic programming techniques for multiple sequence alignment. Notredame and Higgins (1996) presented a method for sequence alignment by employing genetic algorithms (SAGA). Stoye (1998) presented a method of multiple sequence alignment with a divide-and-conquer technique. Thompson *et al.,* (1994), presented a method for improving the sensitivity of progressive multiple sequence alignment. Waterman and Zhang (2003), presented Eulerian path methods for multiple sequence alignment. Zhang and Wong (1997), presented a genetic algorithm for dealing with multiple molecular sequence alignment. Zhang and Wong (1997) presented a method for multiple molecular sequence alignment based on genetic algorithms and dynamic programming (DP) techniques.

In this study, genetic algorithms Dongardive and Abraham (2012) has been considered for experimental analysis. The main advantage of using GA for MSA problem is that there is no need to provide a particular algorithm to solve a given problem. It only needs a fitness function to evaluate the quality of different solutions. Here, in this study different genetic operators has been developed and the role of each operators has been observed, as what they contribute in overall alignment of DNA sequences. Based on these genetic operators experimental studies were performed with standard datasets to have a comparative study with Chen *et al.`s* methods (Chen *et al.,* 2005) over fitness score and match column.

The rest of the paper is organised as follow. The next section introduces the concepts underlying the research work with detailed discussion on the proposed approach, The experiments setup required in order to validate and observe the results and the detailed results over different datasets are discussed in the section that follows. Finally, the concluding section presents the final consideration.

MATERIALS AND METHODS

This section detailed about the proposed approach which is based on various parameters and are described below.

Representation and Initial Generation

In the proposed approach, the population is initially randomly generated at first. Then the largest sequence in size is determined. Based on the largest sequence size, the initially generated population is filled with gap sign (-) until they reach the size of the biggest sequence plus a random number of gaps between 0 and 25% of the size of the largest loaded sequence. These gaps are randomly placed into the sequences. After the population's has initialized, all the solutions are combined and mutated, so as to produce new individuals with a defined number of generations (iterations), which is 100 for this experimental study.

Fitness

The sum-of-pairs score (SPS): is calculated so that the score increases with the number of sequences correctly aligned which is used to determine the extent to which the programs succeed in aligning. The SPS score is defined as below:

Considering a test alignment of size *NxM*, and a reference alignment of size NxMr, where *N* is the number of sequences, and *M,Mr* are the number of columns in the test and reference alignment accordingly and Ai1 Ai1, …, AiN, is the *ith* column in the alignment, for each pair of residues Aij and Aik we define pijk = 1 if residues Aij and Aik are aligned with each other in the reference alignment, otherwise pijk = 0. The score Si for the ith column will be the sum of pijk for all pairs of symbols in this column:

$$S= \sum_{j=1,j\neq 1}^{N} \sum_{k=1}^{N} P_{ijk}$$

Similarly *Sri* is the score Si for the ith column in the reference alignment.
The SPS score for the test alignment is:

$$\text{SPS} = \sum_{I=1}^{M} S_i / \sum_{i=1}^{Mr} S_{ri}$$

The Column Score (CS)

The Column score (CS): Considering a test alignment of size *NxM*, and a reference alignment of size NxMr, where *N* is the number of sequences, and *M*, *Mr* are the number of columns in the test and reference alignment accordingly: the score Ci = 1 if all the residues in the column are aligned in the reference alignment, otherwise Ci = 0
The CS score for the test alignment is then:

$$\text{CS}=\sum_{i=1}^{M} C_i \text{ / M}$$

Since the two scoring systems have been implemented successfully in the program BaliBASE called Baliscore which takes as input a test alignment and a reference alignment in MSF format, in this thesis we will use the Baliscore to estimate the quality of a test alignment in our experiment.

Selection Strategies Description

In the proposed scheme, two different type of selection scheme mainly roulette wheel selection and elitist model selection have been used. Here, 40% of the new generation is directly filled with the fittest individuals from the previous generation and the reaming 60% are filled by modifying the parents through roulette wheel selecting scheme using crossover and mutation operator. The use of specific genetic operators for modifying the parents is depends upon its probability which is 0.6% and 0.01% for crossover and mutation respectively for this experiment.

Crossover

Cross over is a process of taking more than one parent chromosomes and producing a child solution from them. It is performed by selecting two parents with higher fitness values as shown in fig.1 and then randomly selecting a single crossover point, based on the length of the parents. This crossover operation between the parents would give two child chromosomes, which may be considered into the coming generation based on their fitness values. As an experimental scheme, only those child chromosomes which have better fitness scores than their parents are considered for this experimental study.

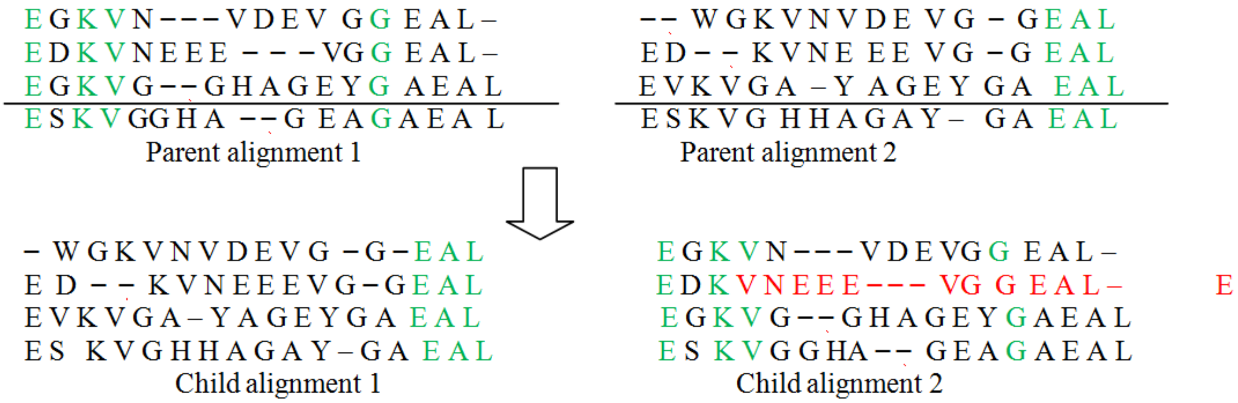

Figure 1: Proposed crossover operator

Mutation

After crossover, the strings are moved for mutation. Mutation prevents the algorithm to be trapped in a local minimum. It distributes the genetic information randomly among other individuals and helps to recover the lost genetic materials. Mutation operation involves randomly flipping of few bits in chromosomes. For example, the string 00100100 might be mutated in its second position to yield 01100100. Mutation operation can happen with very small probability at each bit position in a string.
The main aim of mutation operator is to slightly alter the parent to introduce new' genetic information. The proposed scheme of mutation operator works as follow. First, a shorter segment from the parent is

chosen at random, which is limited to *5≤ l≤80*. Then the chosen segment is divided into two groups from a random chosen position. In each group, the column consisting of only gap character were removed and the Myers-Miller algorithm is used to re-align these two groups to a segment of alignment. Finally, the new segment is connected to two terminal segments of the parent to complete the offspring. Now, if the newborn child is different compared to previously generated children then, it will be put into the new generation, otherwise, it will simply be discarded. As, the length of the short segment is limited to *5≤ l≤80*, the computational time for the mutation is bound by a constant, not dependent on the length of sequences of the problem.

Termination Condition

The termination conditions used for the experiment are as follows:

In the experimental study, the results were tested on maximum 100 iterations (generations), and hence made the experiment to be terminated after reaching 100 iterations, as there is negligible amount of improvement in the alignment quality.

RESULTS AND DISCUSSION

The proposed approach is implemented with C programming language. All tests have been fulfilled on a PC with an Intel i7 core 2.53 GHz processor and 4GB RAM. The experiments for each datasets are processed with the parameters which are most commonly used by the normal users. The population size was established to 1000 individuals and the maximum number of generations was 100 with a crossover probability of 0.6%, mutation rate of 0.01% for the experiment. In order to evaluate the proposed approach, the experiment is carried out with a set of DNA sequences of different length taken from EMBL and GenBank dataset.

Table 1: Representation of the Dataset used for the experimental study

Sequence set number	Number of sequences	Average length (max, min)	Database
S1[1]	20	122 (122, 122)	EMBL
S2[1]	10	211 (212, 211)	EMBL
S3[21]	5	1092 (1142, 1006)	EMBL
S4[21]	5	1093 (1098, 1088)	EMBL
S5[21]	6	1456 (1463, 1430)	EMBL
S6[11]	8	1680 (1680, 1680)	EMBL
S7[11]	6	2685 (2692, 2677)	GenBank

Table 2: A comparison of the scores and match columns of Chen et al.'s method with the proposed method

Sequence Set Number	Chen *et al.'s*		The proposed method	
	Score	Match column	Score	Match column
S1	45957	105	53624	118
S2	17768	193	25142	225
S3	11755	703	12652	754
S4	17912	908	19833	929
S5	34784	1139	35627	1257
S6	84652	1419	86954	1524
S7	77148	2596	69584	2265

In every experiment, alignments were performed both with the proposed method as well as with the Chen *et al.`s* method. Performance, in terms of both fitness score and match column, are recorded in table 2. The accuracy of an alignment of the purposed method is measured with respect to Fitness score and

Match Column (MC). MC is the number of correctly aligned columns to the number of columns in the reference alignment and Fitness score is the number of correctly aligned residue pairs to the number of residue pairs in the reference alignment. Table 1 represents the datasets used in the experimental analysis. Table 2 indicates that for the GenBank dataset, the proposed method didn't give optimal result for both fitness score and match column, as the outcome of these measuring factors is less than the compared method. Fig 2 and 3 represents the bar graph comparison of Fitness score and the Match column between the proposed and Chen *et al.*`s method.

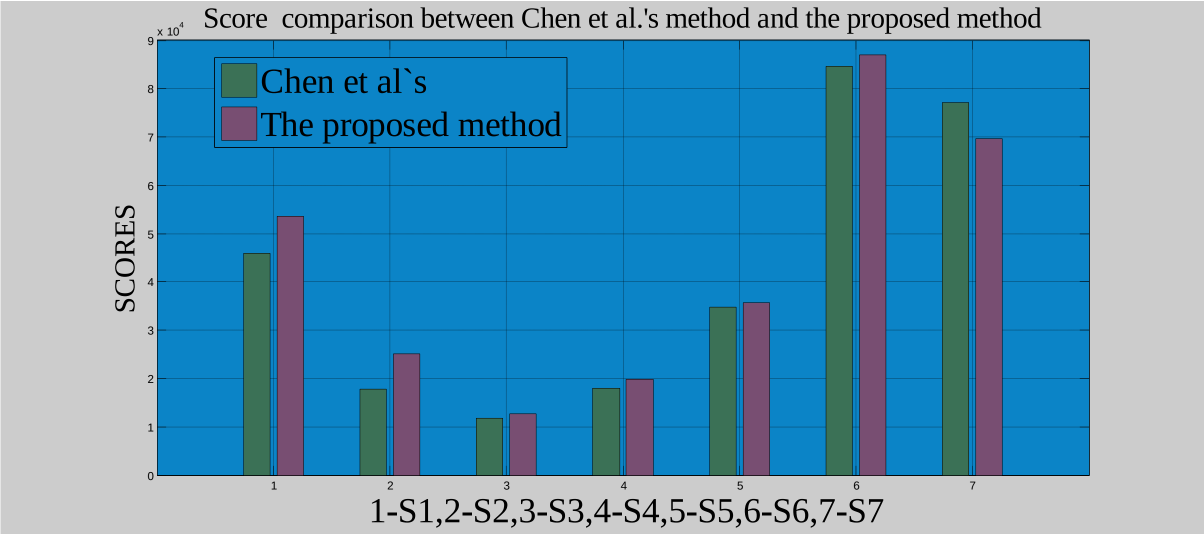

Figure 2: Bar graph comparison result of scores between proposed and Chen *et al.*`s method

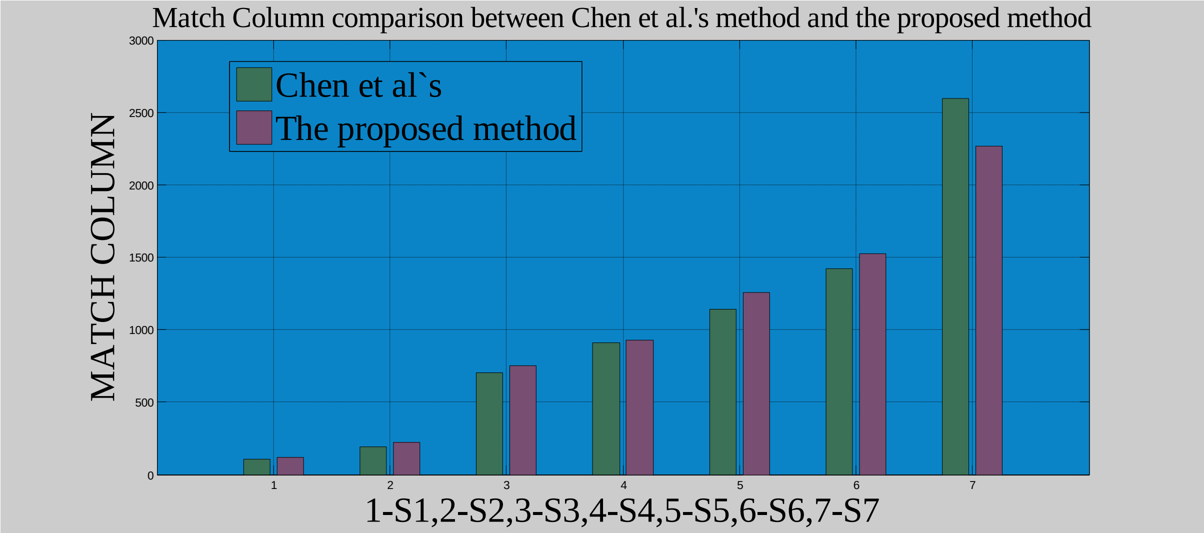

Figure 3: Bar graph comparison result of match column between proposed and Chen *et al.*`s method

Discussion

The iterative techniques used in this paper for solving multiple sequence alignment are Genetic Algorithm. Various methods for crossover, mutation and selection schemes are proposed for optimal alignment of multiple DNA sequences. In this report, an efficient approach for alignment problem of DNA sequences has been presented. It can be analyzed by the experimental results, that how different parameters of genetic algorithm can help in achieving high alignment quality. The comparative study with different method advocates that the presented approach can be utilized in achieving high alignment quality for multiple DNA sequences. Only, for GeneBank dataset the presented approach does not show optimal result. Simulation result shows that the novel presented approach has superior performance compared to other existing algorithms.

REFERENCES

Chen SM, Lin CH and Chen SJ (2005). Multiple DNA sequence alignment based on genetic algorithms and divide-and-conquer techniques. *International Journal of Applied Science and Engineering* **3** 89-100.

Chellapilla K and Fogel GB (1999). Multiple sequence alignment using evolutionary programming, *Proceedings of the Congress on Evolutionary Computation*, Washington, D.C. 445-452.

Chin YLF, Ho NL, Lam TW, Wong WHP and Chan MY (2003). Efficient constrained multiple sequence alignment with performance guarantee. *Proceedings of the IEEE Bioinformatics Conference* 337-346.

Di Pietro C, Di Pietro V, Emmanuele G, Ferro A, Maugeri T, Pigola G, Pulvirenti A, Purrello M, Ragusa M, Scalia M, Shasha D, Travali S and Zimmitti V (2003). Anticlustal: Multiple sequence alignment by antipole clustering and linear approximate 1-median computation. *Proceedings of the IEEE Bioinformatics Conference* 326-336.

Dongardive J and Abraham S (2012). Finding consensus by sequence evolution: An application of Differential evolution. *World Congress on Information and Communication Technologies* 248-53.

Du Z and Lin F (2004). Parallel computation for multiple sequence alignments. *Proceedings of the Joint Conference of the Fourth International Conference on Information, Communications and Signal Processing and the Fourth Pacific Rim Conference on Multimedia* **1** 300-303.

Edgar RC (2004). MUSCLE: Multiple sequence alignment with improved accuracy and speed. *Proceedings of the 2004 IEEE Computational Systems Bioinformatics Conference* 728-729.

Ergezer H and Leblebicioglu K (2004). Multiple sequence alignment using hidden Markov model. *Proceedings of the IEEE 12th Signal Processing and Communications Applications Conference* 434-437.

Hamidi S, Naghibzadeh M and Sadri J (2013). Protein multiple sequence alignment based on secondary structure similarity. *International Conference on Advances in Computing, Communications and informatics* 1224-29.

Hunt FY, Kearsley AJ and O'Gallagher A (2003). A linear programming based algorithm for multiple sequence alignments. *Proceedings of the 2003 IEEE Bioinformatics Conference* 532-533.

Isokawa M, Wayama M and Shimizu T (1996). Multiple sequence alignment using a genetic algorithm. *Proceedings of the Seventh Workshop on Genome Informatics* **7** 176-177.

Lin CH and Chen SM (2004). A new method for multiple DNA sequence alignment based on genetic simulated annealing algorithms. *Proceedings of the International Conference on Information Management, Miauli, Taiwan, R.O.C.* 307-314.

Lin CM (2000). Using genetic algorithms to solve multiple sequence alignments. *Proceedings of the Genetic and Evolutionary Computation Conference, Las Vegas, Nevada* 883-890.

Liu LF, Huo HW and Wang BS (2004). Aligning multiple sequences by genetic algorithm. *Proceedings of the International Conference on Communications, Circuits and Systems* **2** 994-998.

Luo J, Ahmad I, Ahmed M and Paul R (2005). Parallel multiple sequence alignment with dynamic scheduling. *Proceedings of the International Conference on Information Technology: Coding and Computing* **1** 8-13.

Needleman SB and Wunsch CD (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology* **48** 443-453.

Notredame C and Higgins DG (1996). SAGA: Sequence alignment by genetic algorithm. *Nucleic Acids Research* **24** 1515-1524.

Stoye J (1998). Multiple sequence alignment with the divide-and-conquer method. *Gene* **211** 45-56.

Thompson JD, Higgins DG and Gibson TJ (1994). CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position speci_c gap penalties and weight matrix choice. *Nucleic Acids Research* **22** 4673-4680.

Waterman MS and Zhang Y (2003). Eulerian path methods for multiple sequence alignment. *Proceedings of the IEEE Bioinformatic Conference* **29**.

Wei CC, Yu JC, Chien CC, Der TL and Jan MH (2013). Optimizing a Map Reduce module of pre-processing high-throughput DNA sequencing data. *IEEE International Conference on Big Data* 6-9.

Zhang C and Wong AKC (1997). A genetic algorithm for multiple molecular sequence alignment. *Computer Applications in the Biosciences* **13** 565-581.

Zhang C and Wong AKC (1997). Toward efficient multiple molecular sequence alignment: A system of genetic algorithm and dynamic programming. *IEEE Transactions on Systems, Man, and Cybernetics- part B: Cybernetics* **27** 918-932.