An Optimization Approach for Multiple Sequence Alignment using Divide-Conquer and Genetic Algorithm

Arunima Mishra[1]
Computer Science and Engineering
AKTU Lucknow, India

Sudhir Singh Soam[2]
Computer Science and Engineering
IET Lucknow, India

Bipin Kumar Tripathi[3]
Computer Science and Engineering
REC Bijnor, India

***Abstract*—Multiple Sequence Alignment (MSA) is a very effective tool in bioinformatics. It is used for the prediction of the structure and function of the protein, locating DNA regulatory elements like binding sites, and evolutionary analysis. This research paper proposed an optimization method for the improvement of multiple sequence alignment generated through progressive alignment. This optimization method consists of a fusion of two problem-solving techniques, divide-conquer and genetic algorithms in which the initial population of MSAs was generated through progressive alignment. Each multiple alignment was then divided vertically into four parts, three genetic operators were applied on each part of the MSA, recombination was done to reconstruct the full MSA. To estimate the performance of the method the results generated through the method are compared with well-known existing MSA methods named Clustal Ω, MUSCLE, PRANK, and Clustal W. Experimental results showed an 11-26% increase in sum_of_pair score (SP score) in the proposed method in comparison to the above-mentioned methods. SP score is the cumulative score of all possible pairs of alignment within the MSA.**

Keywords—Multiple sequence alignment; divide; and conquer; genetic algorithm; optimization method

I. INTRODUCTION

Sequence alignment (SA) is the most common and essential task of bioinformatics. Pairwise SA is an alignment of two biological sequences where the similarity between the two sequences has been revealed through the alignment, few examples of SA are EMBOSS [1], BLAST[2], PSI-BLAST[3], and AlignMe[4], in the case when three and more sequences are aligned, is referred as multiple sequence alignment (MSA). The objective of the MSA is to arrange the sequences in a way that exposes the evolutionary connection between the biological sequences. The key applications of MSA are the identification of a protein family-like phylogenetic analysis and finding DNA regulatory elements. MSA is a well-known problem of combinatorics and its complexity is quite high, hence to get an exact solution is not practically possible for a large number of sequences [5], that is why most of the multiple alignment methods are heuristic and provide approximate solutions. Progressive alignment and iterative alignment are the two most applied approaches for MSA. The progressive alignment method is primarily based on the PSA in which pairwise alignment is done for all the possible pairs of sequences, a distance matrix is made that shows the dissimilarities between the sequences. A guide tree is constructed through the distance matrix by any clustering algorithm. The guide tree displays the order of sequences to be aligned, most similar sequences are aligned first followed by the sequences of less similarity. Feng and Doolittle were the first who proposed a progressive alignment algorithm for MSA [6]. Many MSA methods based on the progressive alignment have been developed like CLUSTALW [7], MULTALIGN [8], CLUSTAL X[9]. The major disadvantage of this method is that the resulting MSA gets affected by the initial alignments so the position and length of gaps of aligned sequences can not be changed at a later stage.

Iterative alignment provides a solution to this problem through iteratively modifying the previously aligned sequences while keeps on adding the new sequences, few examples of iterative alignment are MAFFT [10,11], MUSCLE [12,13], and PRRP [14].

The machine learning area has been explored and several methods are applied to deal with the MSA problem. Ant colony optimization was applied by Chen et al [15] it was a partitioning approach that consists of three phases. Ant colony optimization was applied on each part and at last, all the parts were reassembled to get the solution.

Particle swarm optimization was combined with the Hidden Markov model to get the MSA by Rasmussen et al [16], they displayed improved results for protein sequences than the other HMM method for MSA like simulated annealing [17].

Reinforcement learning (RL) algorithms are used in solving the problem. Mircea et al [18] applied it the first time. The Q learning algorithm was applied along with the action-selection approach like softmax and epsilon-greedy for balancing the explore-exploit strategy. This strategy explores the solution space that may not provide instant high scores but may lead to a higher gain in the longer term. Exploitation is the application of information already gained by prior experiences. A good balance of exploration and exploitation helps to reach the optimum result in lesser time. Reza Jafri et al [19] used the deep Q learning method along with the actor-critic algorithm and experience replay method. They showed that their method has a speedy convergence. RLALIGN [20] is a pure RL-based algorithm for MSA.

Genetic algorithm (GA) is a type of iterative method, analogous to the theory of natural evolution. It generates many solutions at each stage every solution is attached with a fitness function that describes the goodness of that solution. The genetic operators like mutation operators and recombination operators are applied to the selected entities at each stage iteratively until the result converged to the best possible fitness score. Due to the MSA's discrete nature, GA is well suited to this problem. SAGA [21] is a famous MSA method developed by Higgins and Notredame, based on the GA. It attempts to get the MSA by the number of complex genetic operators. One more approach was proposed by Nazneen et al [22] in which the initial population was generated through randomly produced subtrees and then by shuffling of those subtrees. MSA-GA [23] is another method in which the initial population was produced through dynamic programming and then Genetic operators were applied to it to get the next generation population iteratively.

This paper suggests an approach named Genetic algorithm-based optimization with divide and conquer method (GAODC) which is a fusion approach of two problem-solving techniques namely a genetic algorithm and divide and conquer methodology. In this approach, the Sum_of_Pair score is used as a fitness function. It divides the MSAs into four parts and two operators namely insertion mutation and deletion mutation are applied to those parts. The fitness score is calculated for each part and recombination is done between the parts of two MSAs starting with the MSAs having the highest fitness score.

The most distinguishing feature of this methodology is that the recombination of MSAs is based on the fitness score of the individual parts of MSAs. Recombination between the MSAs with high fitness scores has a great chance of construction of new MSA with higher fitness scores. To evaluate the performance of GAODC, it is compared with other popular MSA methods, namely PRANK[24], CLUSTAL Ω, MAFFT, and MUSCLE. BAliBASE 3.0 [25] database is used for the evaluation of the method. Sum_of_pair score and total_column score are the two objective functions that are used as an evaluation criterion for all the MSA methods.

The rest of the paper is principally divided into eight main sections: Section II presents the basic definitions of the progressive alignment, Divide-conquer approach, and Genetic algorithm. The methodology of GAODC is explained in Section III, Section IV mentioned the fitness function and the scoring scheme. Datasets used in the method are explicated in Section V. Results generated through GAODC and other existing methods are compared in Section VI, Section VII summarizes the conclusions, and Section VIII lists out the references cited in the paper.

II. Preliminaries

This section contains the preliminaries that are used in the proposed method. Section A explains the progressive alignment technique that is applied to generate the initial population, section B and C contains the basic steps of divide and conquer and genetic algorithm respectively.

A. *Progressive Alignment*

Progressive alignment is a basic technique of multiple sequence alignment it starts with the pairwise sequencealignment of any two sequences and then the third sequence is aligned, and this process continues till all the sequences get aligned. This method does not guarantee optimal alignment, but it is a very fast method for MSA. Following are the main steps of progressive alignment:

1) Make the distance matrix for M(M-1)/2 pairs of sequences of M sequences.

2) Make a guide tree with the help of the matrix using clustering algorithms like neighbor-joining [26] and UPGMA[27], which shows the order of sequences to be aligned.

3) Add the sequences into the alignment starting from the sequences added first followed by other sequences added to the guide tree. An example of a guide tree is shown in Fig. 1 for the sequences S1, S2, S3, and S4. As depicted in the figure, S1 and S2 will be aligned first followed by S3 and S4.

B. *Divide and Conquer*

Divide and conquer is a recursive problem-solving approach. It breaks the complex problem into smaller subproblems of similar type till the subproblems are converted to simple problems that can easily be solved then each subproblem is solved and combined to obtain a complete solution. Three main steps of the divide and conquer method are divide: in which the problem is divided into subproblems of the same kind, the second step is to conquer, that solves the subproblems recursively and the final step is to combine, where all the solutions are combined to achieve the final solution of the entire problem.

C. *Genetic Algorithm*

Genetic algorithms (GA) are analogous to the process of evolution where the best individuals are chosen to produce next-generation offspring. The main steps of GA are as follows:

1) Generation of the initial population.

2) Calculate the fitness function for each entity within the population.

3) Choose some individual as parents.

4) Apply genetic operators on them.

5) Produce the next generation with the help of some recombination of the previous generation.

6) Repeat steps 2 to 5 until the stop criterion.

7) End.

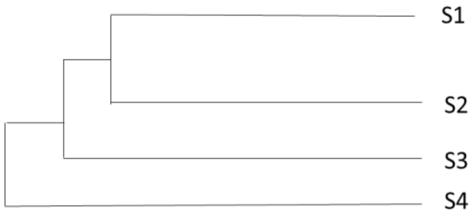


Fig. 1. The Guide Tree Displays the Order of the Sequences to be Aligned in Progressive Alignment.

III. Method

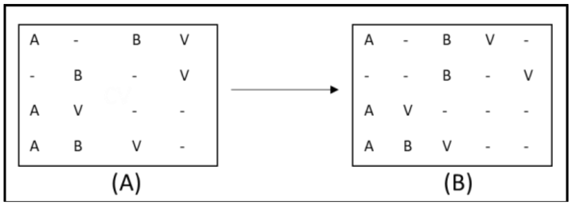
Genetic algorithm-based optimization with divide and conquer (GAODC) consists of the following main steps: population initialization, division, mutation, and recombination.

After generating the initial population with the help of progressive alignment and random insertion of gaps, the division of each individual into four upright parts is performed, Fig. 3 shows the vertical division of the MSA. Mutation operators are applied on each part of the MSA and two types of recombination (One Point and Two Point recombination) are performed to generate a next-generation population. Mutation operations are done on each part of the MSA instead of the whole MSA and one point and two-point recombination are achieved based on the fitness score. Application of mutation operators on the vertical parts of MSA instead of full MSA and the fitness score-based recombination are the main features of the method. This vertical division is done after the initial population generated, two mutation operators, namely, insertion mutation operator and deletion mutation operators are applied on each segment of the MSA to achieve a better alignment score.

A. Gap Insertion Mutation

a gap insertion mutation operator is introduced that picks an MSA from the population and creates a gap randomly at each row, the changes are retained if fitness improves. Fig. 2 illustrates the example of gap insertion mutation for the following four sequences:

S1: ABV, S2: BV, S3: AV, and S4:ABV



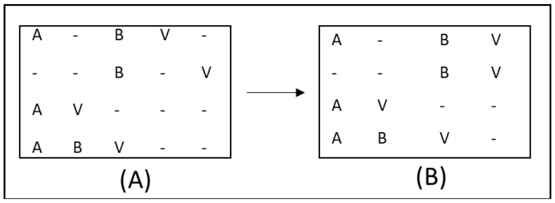
A	-	B	V
-	B	-	V
A	V	-	-
A	B	V	-

A	-	B	V	-
-	-	B	-	V
A	V	-	-	-
A	B	V	-	-

Fig. 2. Depicts Insertion Mutation Applied on Alignment A and the Resulting Alignment B.

B. Gap Removal Mutation

A gap elimination mutation operator is designed that picks an MSA and selects the positions randomly to eliminate the gap. If the selected positions are not a gap then move forward till a gap is found and delete the gap. If no gap is found, then go back and delete the first gap found. Retain the changes if the fitness score is increased. Fig. 3 illustrates the example of deletion mutation operation on MSA (A) for the sequences S1, S2, S3, and S4.



A	-	B	V	-
-	-	B	-	V
A	V	-	-	-
A	B	V	-	-

A	-	B	V
-	-	B	V
A	V	-	-
A	B	V	-

Fig. 3. Depicts Deletion Mutation Applied to Alignment A and the Resulting Alignment B.

The key feature of the algorithm is the division step which takes place before the mutation operators are applied and the mutation is done on the part of the MSAs instead of the full MSA. The crux of this algorithm is that this strategy increases the randomness and the recombination between the parts of the parents based on the fitness score, enhances the chance of getting a new MSA with a higher score. Following are the main steps of the algorithm-

1) Population initialization: The initial population is produced with the help of the progressive alignment technique which consists of the following steps:

A. Compute the distance between each possible pair of sequences using the formula:

$$d\ (Q_i\ , Q_j) = 1-\ \{\ M\ (Q_i,Q_j)/\ \min\ (l_i,l_j)\ \} \qquad (1)$$

Here M (Q_i, Q_j)- Number of matches between the ith sequence and jth sequence.

li and lj – Sequence length of the ith sequence and jth sequence, respectively:

B. A guide tree is constructed to get the order of sequences to be aligned.

C. Sequences are aligned in the order directed by the guide tree. Most identical sequences are aligned first followed by the distant sequences. Three types of alignment are possible here

   a. Sequence to sequence alignment
   b. Group of aligned sequences to a sequence
   c. Group to group.

After generating the MSA through progressive alignment, random gap insertions are done to generate the initial population of size n.

2) To generate the next generation alignments these steps are used: Division: Divide the individual MSA into four upright parts for example, from MSAs 1 to n, it will be - a1, b1, c1, d1 to an, bn, cn, dn. Deletion mutation operators and insertion mutation operators are applied on each part of the MSA individually and changes are saved with the highest fitness score of each part. Now we have four parts of each parent, fitness score of (a+b), and (a+b+c) parts are calculated for all the MSAs. Fig. 4 illustrates the example of the division process for four sequences S1: ABVKWSPNVS, S2: BVKWSNS, S3: AVKSPV, and S4: ABVKSYS. Now two types of recombination are done to produce the next generation alignments, one-point recombination, and two-point recombination. An illustration of one-point recombination for the above example is shown in Fig. 6, 7 and 8 whereas Fig. 5 depicts the two-point recombination for the above-mentioned example.

One-point recombination: It contains the following steps:

1) (a) part of one parent with MaxScore(a) will be combined with the (b+c+d) part of the other parent with MaxScore (b+c+d).

2) (a+b) part of one parent with MaxScore (a+b) will be combined with the (c+d) part of the other parent with MaxScore(c+d).

3) (a+b+c) part of one parent with MaxScore(a+b+c) will be combined with the d part of another parent with MaxScore(d).

4) Continue Steps 1, 2 and 3 with other parents with the next maximum scores.

5) Evaluate the fitness function for all new MSAs.

Two-point recombination: Two-point recombination contains the following steps:

1) Part a and c of one parent having MaxScore(a+c) are recombined with the b and d part of the other parent with MaxScore(b+d).

2) Continue step 1 with other parents with the next maximum scores.

3) Evaluate the fitness score for all new MSAs.

Elitism: This is a popular approach of the genetic algorithm where the individual with the highest fitness value of that generation, is passed as it is to the next generation so that we may not lose the best solution at any stage.

New generation: The creation of a new generation is formed with the selection of the best distinct half of the collective parents, and children who are generated through the mutation and crossovers. The key feature of the method is that selection of parents is not random but it is based on the fitness score of the part to be recombined with the part of the other parent and this increases the possibility of getting a higher score after recombination and safeguard a good balance between exploitation and exploration. The new generation formation (shown in Fig. 9) is considered as the parent population of the next generation and therefore the process of evolution continues.

Termination condition: The best score and its corresponding MSA have recorded in each generation if there is no improvement in the solution till 100 MSA then the execution of the algorithm will be ended.

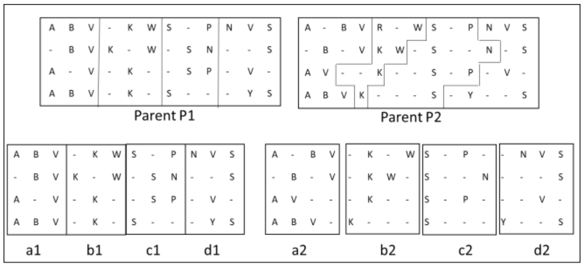


Fig. 4. Depicts the Division Process of MSAs in Four Parts Namely a,b, c, and d.

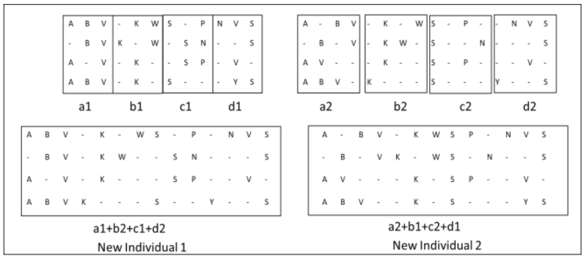


Fig. 5. Two-point Crossover after Applying Insertion and Deletion Mutation on a, b, c, and d Parts of the MSAs.

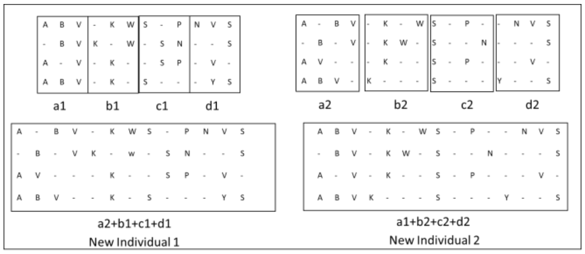


Fig. 6. One-point Crossover on (a) Part of One Parent with MaxScore (a) Combined with the (b+c+d) Part of the other Parent with MaxScore (b+c+d), after Applying Insertion and Deletion Mutation on a, b, c, and d Parts of the MSAs.

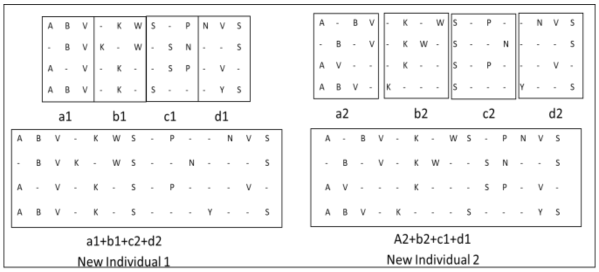


Fig. 7. One-point Crossover on (a+b) Part of One Parent with MaxScore (a+b) Combined with the (c+d) Part of the other Parent with MaxScore (c+d), after Applying Insertion and Deletion Mutation on a, b, c, and d Parts of the MSAs.

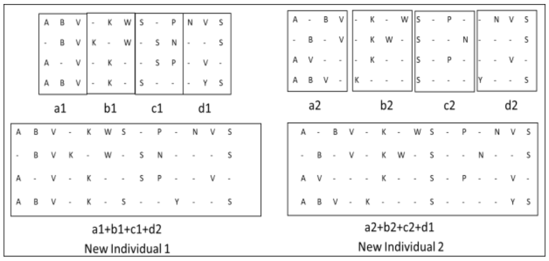


Fig. 8. One-point Crossover on (a+b+c) Part of One Parent with MaxScore(a+b+c) Combined with the (d) Part of the other Parent with MaxScore (d), after Applying Insertion and Deletion Mutation on a, b, c, and d Parts of the MSAs.

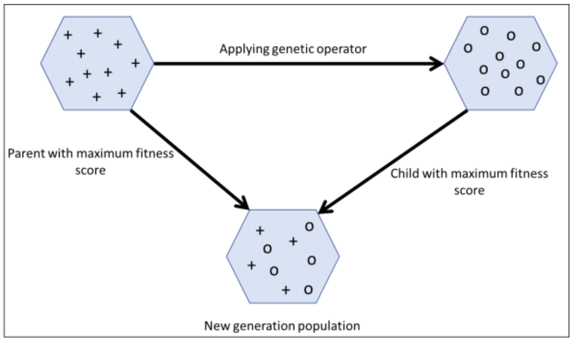

Applying genetic operator
Parent with maximum fitness score
Child with maximum fitness score
New generation population

Fig. 9. Depiction of Construction of New Generation Population.

IV. Fitness Function and Scoring Scheme

The fitness function in a genetic algorithm comprises all the parameters of the particular problem and estimates the solution that how much it is close to the actual solution. In the MSA problem, the most reliable scoring scheme Sum _of _Pair score (SP score) is taken as the fitness function, it is represented by the equation 2:

$$S = \sum_{j=1}^{n-1} \sum_{k=j+1}^{n} S(j,k) \text{ +Gap- Penalty} \qquad (2)$$

Here S(j,k), is the sum of the pair score of jth and kth sequences, and ' n' is the total number of sequences. The sum_of_pair score for all n(n-1) /2 pairs of the biological sequences is computed and added. For the match/mismatch score the BLOSUM 62 matrix is used.

The gap penalty is a fine incurred for inserting a gap in the process of MSA. The affine gap penalty is applied to compute the fine, represented by equation 3:

$$Gp = A + B\,(t-1) \qquad (3)$$

Where A is the Gap Opening penalty, B is the gap extension penalty and t is the number of consecutive gaps in a row.

To calculate the quality of MSA methods one more parameter total_column_score (TC score) is being used. TC score evaluates the ability of the MSA methods to align all the residues appropriately in each column. Mathematically it is defined in equation 4:

$$S = \sum_{i=1}^{d} \begin{cases} 1 \text{ if } Ti = Ri \\ 0 \text{ 0therwise} \end{cases} \qquad (4)$$

Here d is the length of the MSA, Ti and Ri are the ith columns of test MSA and reference MSA respectively if the column of test MSA matched completely it will return '1' else it will return '0'. The summation of all the values for each column divided by the number of total columns gives the value of TCscore as shown in equation 5.

$$TCscore = S/d \qquad (5)$$

V. Database

To evaluate the performance of the proposed method the dataset BAliBASE V3.0 (http://www.lbgi.fr/balibase/) was chosen. It is a commonly used benchmark database of protein sequences. It contains the set of protein sequences and their corresponding reference MSAs. It comprises an application BAliscore that calculates the SPscore and TCscore of test MSAs with the comparison of reference MSAs, its scores vary in the range of 0 to 1. If the test MSA is identical to the reference MSA the value of BAliscore is '1' and if the test MSA not at all matches the reference MSA BAliscore is '0'.

BAliBASE v3 contains six different groups of protein sequences namely RV11, RV12, RV20, RV30, RV40, and RV50. RV11 has 38 sets of very divergent protein sequences that are equidistant and have <20% identity. RV12 was constructed by 44 sets of sequences of 20%- 40% similarity. RV20 formed with 41 sets of sequences having few orphan sequences while all other sequences are having <40% sequences. RV 30 consists of 30 sets of sequences from different families having <25% of identity among the families. RV 40 is a set of 49 sets of protein sequences with a large number of insertions whereas RV 50 is formed by 16 sets of sequences having a large number of internal insertions with < 20% identity.

VI. Results and Discussion

To evaluate the results of GAODC, its results for benchmark database BAliBASE 3.0 are compared with four other MSA methods namely Clustal Ω, MUSCLE, Clustal W, and PRANK. Parameter values for GOADC are given in Table I. Two criteria SPscore and TCscore are used to evaluate the quality of the MSAs. The results are shown in Table II.

Fig. 10 depicts the sp scores of five methods across the six datasets for which the analysis is conducted. Fig. 11 illustrates the TC Score across the same datasets used to calculate SP Score for all five methods.

TABLE I. PARAMETERS USED IN GAODC

Name	Value
Population Size	100
Substitution Matrix	BLOSUM 62
Gap Penalty	Gap opening -3, Gap extension-2

TABLE II. RESULTS OF SP SCORE AND TC SCORE OF EACH METHOD ACROSS ALL SIX DATASETS

Methods		GAODC	CLUSTALΩ	MUSCLE	CLUSTAL W	PRANK
RV11	SP	0.548	0.452	0.442	0.5	0.354
	TC	0.25	0.247	0.228	0.229	0.168
RV12	SP	0.878	0.826	0.827	0.865	0.737
	TC	0.763	0.683	0.679	0.717	0.548
RV20	SP	0.855	0.772	0.766	0.852	0.7
	TC	0.15	0.31	0.25	0.22	0.17
RV30	SP	0.82	0.68	0.65	0.73	0.51
	TC	0.33	0.37	0.25	0.28	0.18
RV40	SP	0.845	0.756	0.726	0.789	0.6
	TC	0.402	0.428	0.338	0.398	0.244
RV50	SP	0.747	0.681	0.724	0.742	0.55
	TC	0.481	0.417	0.344	0.312	0.245

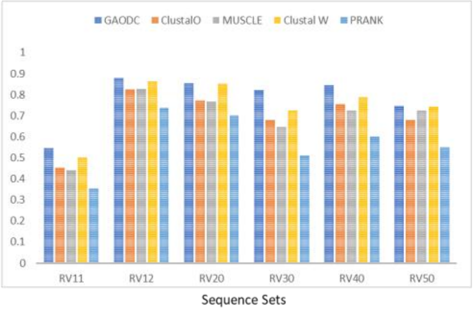


Fig. 10. Comparative Results of SPscore of each Method Across all Six Datasets.

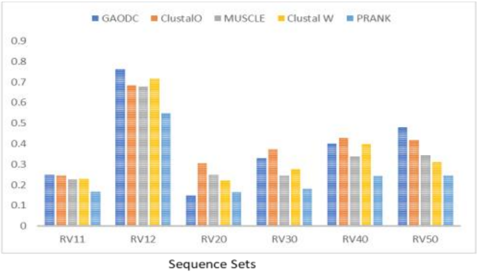


Fig. 11. Comparative Results of TCscore of each Method Across all Six Datasets.

Results show that the overall performance of the method is better than other methods. SP score of GAODC is approximately 13% higher than all other methods across six databases. The TC score of the GAODC method is highest among all other methods for four datasets including RV11, RV12, RV40, and RV50. In the case of datasets RV 20 and RV 30, which contain orphan sequences and groups of sequences having different families respectively, the TCscore of GAODC is marginally less by approximately 1.7% than the other methods.

High SP Score and TC Score suggest that GAODC generate better quality MSAs as compared to other methods used in this analysis.

VII. Conclusion

This paper proposes a method for the MSA of biological sequences that is a combination of two problem-solving techniques divide-conquer and genetic algorithm. As part of this method, the recombination method is applied where the MSAs are recombined based on the SP score of the parts of each MSA thus increasing the possibility of getting the most optimum MSA. Results show that our method outperformed the other widely used MSA techniques on SPscore criteria.

References

[1] Rice P, Longden I, Bleasby A,” EMBOSS: the European molecular biology open software suite”, Trends Genet. 2000; 16:276–7.

[2] Johnson M, Zaretskaya I, Raytselis Y, Merezhuk Y, McGinnis S, Madden TL, “NCBI BLAST: a better web interface”, Nucleic Acids Res. 2008;36: W5–9.

[3] Altschul SF1, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ, “Gapped BLAST and PSI-BLAST: a new generation of protein database search programs”, Nucleic Acids Res. 1997 Sep 1;25(17):3389-402.

[4] Marcus Stamm, René Staritzbichler, Kamil Khafizov, and Lucy R. Forrest, “AlignMe—a membrane protein sequence alignment web server”, Nucleic Acids Res. 2014 Jul 1; 42(Web Server issue): W246–W251.

[5] F. Sievers, A. Wilm, D. Dineen, et al., “Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega”, Molecular Systems Biology, vol. 7, article 539, 2011.

[6] Feng DF, Doolittle RF, “Progressive sequence alignment as a prerequisite to correct phylogenetic tree",1987, J Mol Evol. 25 (4): 351–360.

[7] Thompson JD, Higgins DG, Gibson,”. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties, and weight matrix choice”, Nucleic Acids Res.1994;22:4673–80.

[8] Corpet F., “Multiple sequence alignment with hierarchical clustering”. Nucl Acids Res. 1988.

[9] Francois Jeanmougin et al “Multiple sequence alignment with Clustal X “ Trends in biochemical sciences, 1998.

[10] Katoh K, Standley DM,” MAFFT multiple sequence alignment software version 7: improvements in performance and usability”, Mol Biol Evol. 2013; 30:772–80.

[11] Katoh K, Misawa K, Kuma K, Miyata T,” MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform”, Nucleic Acids Res., 2002;30:3059–66.

[12] Edgar RC,” MUSCLE: multiple sequence alignment with high accuracy and high throughput”, Nucleic Acids Res. 2004;32:1792–7.

[13] Edgar RC,” MUSCLE: a multiple sequence alignment method with reduced time and space complexity”, BMC Bioinformatics. 2004; 5:113.

[14] Gotoh O,” Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments”, J Mol Biol 1996, 264: 823–838.

[15] Chen Y, Pan Y, Chen L, Chen J (2006) Partitioned optimization algorithms for multiple sequence alignment. In: Proceedings of the 20th international conference on advanced information networking and applications, pp 618–622.

[16] Rasmussen TK, Krink T (2003) Improved hidden Markov model training for multiple sequence alignment by a particle swarm optimization-evolutionary algorithm hybrid. BioSystems 72:5– 17.

[17] Jin Kim, Sakti Pramanik, Moon Jung Chung, “Multiple sequence alignment using simulated annealing”, Bioinformatics, Volume 10, Issue 4, July 1994.

[18] M.I. Bocicor, I.G. Mircea, and G. Czibula. “A novel reinforcement learning-based approach to multiple sequence alignment, Information Sciences, 2014.

[19] Reza Jafari, Mohammad Masoud Javidi · Marjan Kuchaki Rafsanjani, “Using deep reinforcement learning approach for solving the multiple sequence alignment problem”, 2019, Springer Nature Switzerland.

[20] RK Ramakrishnan, J.Singh and M. Blanchette, “RLALIGN: A Reinforcement Learning Approach for Multiple Sequence Alignment”, IEEE 18th International Conference on Bioinformatics and Bioengineering,2018.

[21] Notredame C, Higgins DG (1996) SAGA: sequence alignment by genetic algorithm. Nucl Acids Res 24:1515–1524.

[22] Naznin F, Sarker R, Essam D (2011) Vertical decomposition with genetic algorithm for multiple sequence alignment. BMC Bioinformatics 12:353.

[23] Gondro C, Kinghorn BP (2007) A simple genetic algorithm for multiple sequence alignment. Genet Mol Res 6:964–982.

[24] Ari Loytynoja, “ Phylogeny-aware alignment with PRANK”, multiple Sequence Alignment Methods pp 155-170.

[25] Thompson JD, Koehl P, Ripp R, Poch O, “ BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark”, Proteins. 2005 ;61(1):127-36.

[26] Saitu N, Nei M, “ The neighbor-joining method: a new method for reconstructing phylogenetic trees”, Mol Biol Evol. 1987 Jul;4(4):406-25.

[27] I.GronauandS.Moran, “Optimal implementations of UPGMA and other common clustering algorithms”, Information Processing Letters, vol.104, no.6, pp.205–210,2007.