Auto-Poietic Algorithm for Multiple Sequence Alignment

Amouda Venkatesan and Buvaneswari Shanmugham
Centre for Bioinformatics, Pondicherry University, India

Abstract: *The concept of self-organization is applied to the operators and parameters of genetic algorithm to develop a novel Auto-poietic algorithm solving a biological problem, Multiple Sequence Alignment (MSA). The self-organizing crossover operator of the developed algorithm undergoes a swap and shuffle process to alter the genes of chromosomes in order to produce better combinations. Unlike Standard Genetic Algorithms (SGA), the mutation rate of auto-poietic algorithm is not fixed. The mutation rate varies cyclically based on the improvement of fitness value in turn, determines the termination point of algorithm. Automated assignment of various parameter values reduces the intervention and inappropriate settings of parameters from user without prior the knowledge of input. As an advantage, the proposed algorithm also circumvents the major issues in standard genetic algorithm, premature convergence and time requirements to optimize the parameters. Using Benchmark Alignment Database (BAliBASE) reference multiple sequence alignments, the efficiency of the auto-poietic algorithm is analyzed. It is evident that the performance of auto-poietic algorithm is better than SGA and produces better alignments compared to other MSA tools.*

Keywords: *Auto-poietic, crossover, genetic algorithm, mutation, multiple sequence alignment, selection.*

Received October 27, 2014; accepted November 29, 2015

1. Introduction

Genetic Algorithm (GA), a randomized search algorithm, in a large solution space act as an intelligent tool and explore the prospective regions for finding better solution in an acceptable amount of time [6].

GAs are robust and adaptive in nature, its application is not confined to the problem and can be applied to any combinatorial problem with slight modification of its elements. Since, GA is not problem specific its basic structure remains the same even if any modification in representation, fitness value, choice of parameter, operators are made. This feature of GA solves difficult problems quickly, accurately, and reliably where little is known with the knowledge of fitness function of the individuals. A major setback in computational approaches in the design and optimization of bio-process systems is the lack of reliable methods. One such biological problem, Multiple Sequence Alignment (MSA), a process of aligning more than 3 or more sequences, has been identified as a challenging NP-hard problem due to its complexity in optimization [14]. Though, many variations of Standard Genetic Algorithms (SGA) have been investigated on MSA [7], the efficiency of GA can be realized only by setting the appropriate parameter values. As various inputs require different parameter setting, they are usually obtained by a lengthy process of trial and error, and require a lot of parameters manually to find a better solution, prior to application. When the solution space of the problem becomes too large, especially in the real-world applications, inappropriate selection of parameter values leads to premature convergence and fail to find the optimum in a reasonable computational time [12, 16]. This necessitates the proposal of a novel Self-Organizing Genetic Algorithm (SOGA), which combines GA and the concept of self-organization in order to self-configure the parameters like population size, probabilities, etc. Self-organizing systems emerges global behaviour based on numerous local interactions without the intervention from the external control [15, 17].

In general, GA operation starts with an encoding of decision variables followed by the replication of chromosomes to form the population. A fitness value is assigned to each individual indicates how well, it is better or close to an optimal solution. Genetic operators affect the individuals of population and represent a possible solution to a given problem. The best individuals with a high fit are reproduced by the operational parameters of crossover and mutation for the next generation. In SGA, this procedure is repeated a number of times until the termination condition is satisfied given by the user [5].

2. Related Work

Genetic algorithms have been applied in various domain problems and the biological field is no exception [17]. Irrespective of the problem type, several variants of SOGAs were proposed to improve the performance, mostly in the form of designing new operators or hybrid algorithms combined with conventional GA. Zhang *et al*. [21] have proposed a

proposed a SOGA constructed with a dominant selection operator enhancing the action of dominant individuals and a cyclical mutation operator that periodically varies the mutation probability during execution. A nonlinear regression analysis demonstrates that the algorithm is able to avoid premature convergence with a higher convergence speed with self-organization properties. Khayat *et al*. [9] have proposed Self-Organizing Fuzzy Neural Network based on Genetic Algorithm and Particle Swarm Optimization (SOFNNGAPSO) in which the parameters of consequent parts were obtained using the error function and the parameters of premise parts in an iterative process. In the second stage, an optimization process based on GA and PSO was used to evaluate the structure of SOFNNGAPSO with the objective of adjusting the parameters of premise parts and consequent parts. Initializing the parameters of consequent parts randomly, updating them in the parameter- dependent iterative process and using GA and PSO in the optimization phase which are generally slow to reach the global optimum have made the proposed algorithm dull. Tin´os *et al*. [19] Random Immigrants Genetic Algorithm (RIGA), the new individuals created in the current chain reaction are preserved in a sub-population. Two strategies for replacement of individuals in the current population is carried

1. Some individuals with randomly generated individuals.
2. Lowest fitness individuals with random immigrants.

To Self-Organize RIGA (SORIGA), two modifications are made in the replacement function and the selection scheme.

Wu *et al*. [20] have presented an algorithm combining the genetic algorithm and self-organizing neural network to solve MSA. This approach demonstrated improved performance in long DNA and RNA data sets exhibiting small similarity. Self-organizing neural network as local optimization like classification is embedded into genetic algorithm to keep away from local optima. Kubota *et al*. [10] have suggested a Virus-Evolutionary Genetic Algorithm (VEGA) based on virus theory of evolution composed of a host population of candidate solutions and a virus population of substrings of host individuals.

A reverse transcription operator overwrites a virus string on a host's string and a transduction operator generating a new virus from a host string. The virus infection operators enable increase of effective schemata with reverse transcription and transduction.

The reverse transcription plays roles of crossover and selection simultaneously, since the reverse transcription generates new individuals overwriting on host individuals according to the virus infection rate.

Further, transduction generates new virus individuals and evolves the virus population. Therefore, co-evolution of the virus population and the host population enables quick solution of the optimization problem.

3. The Proposed Algorithm: Auto-Poietic MSA

The developed algorithm is focused to make use of the influence of crossover operator to improve the population diversity to enhance the global exploration. The algorithm starts with the generation of chromosomes (possible solutions) consisting of gap positions. The integer form of chromosome with gap positions is changed to binary form (presence and absence of gap) for mutation process. The length of the chromosomes varies for each input depending on the length of sequences [18]. The chromosome representation employed and the process of population initialization adopted as shown in the previous work [11]. Elite, the chromosome with best fitness value are selected and saved for each genetic operation. During crossover operation, each gene in a chromosome is evaluated individually to identify the best one among the chromosomes of current population.

A new strategy, self-organizing dual operation of designed crossover operator is based on gene level distribution. The Auto-poietic Crossover Operator (ACO) selects the best genes of all chromosomes followed by a dual process, swapping and shuffling.

The best genes of all chromosomes are swapped in different combinations. Thereby, it increases the population size from initial population paving way to the generation of best individuals which could not be arrived at chromosome level using normal crossover.

The chromosome representation is converted to binary form by a newly developed mutation operator Self-Organizing Binary Shuffler (SOBS) to perform mutation for a range of rates [11]. The mutation rate is increased cyclically at uniform intervals based on the betterment of the resulting population which enhances the capacity of GA to refine its solution is an added advantage. Based on the improvement, the total number of generation thereby, the termination is also self-organized [13].

3.1. Fitness Function

In order to select the elite (best chromosome of the current population) after every genetic operation, the fitness of every chromosome is evaluated using a function called Column Score (CS). Column Score=Exact Match/ Alignment length, where, exact match=1, when aligned residues in the entire column are same [13, 14].

3.2. Pseudo Code of Auto-Poietic Algorithm for MSA (AP-MSA)

1. [Start] Generate the random population of *n* chromosomes.

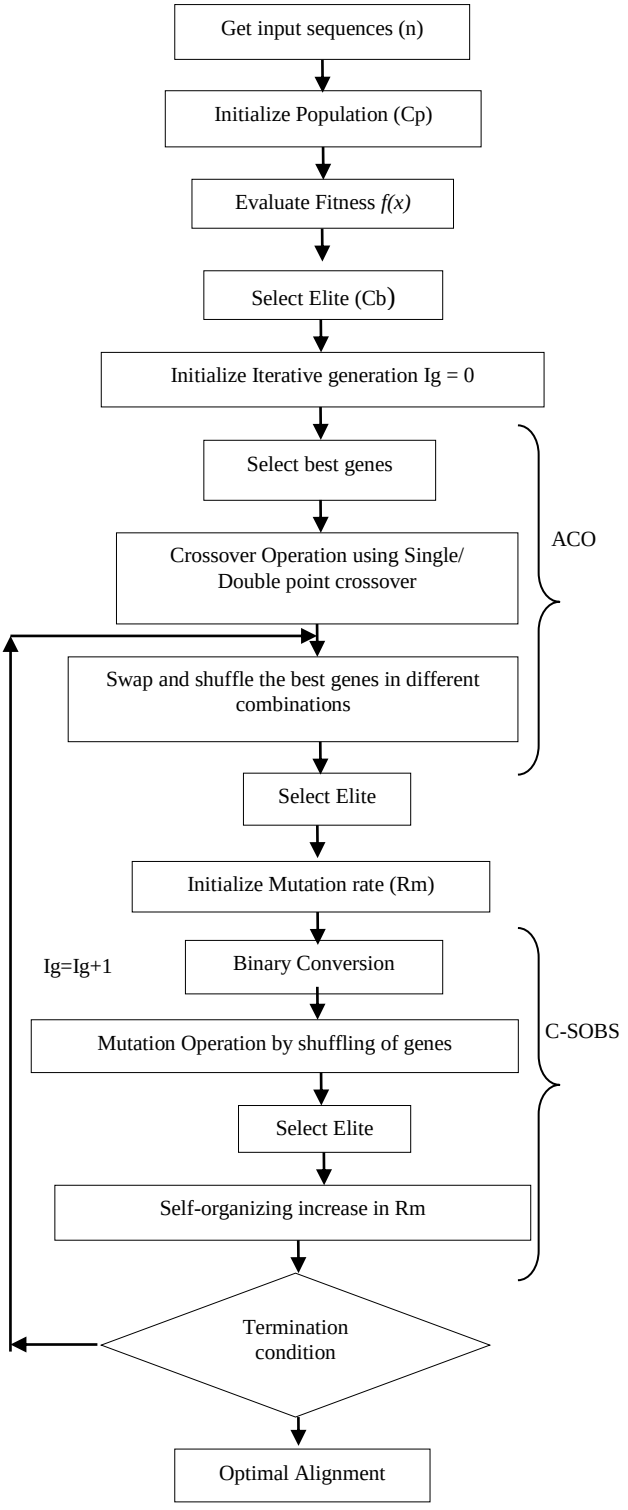

Figure 1. Flowchart of Auto-Poietic Algorithm for MSA (AP-MSA).

2. [Fitness] Evaluate the fitness *f(x)* of each chromosome *x* in the population.
3. [Selection] Select and save elite (chromosome with best fitness value) of current population.
4. [New Population] Create a new population using (i to iv) repeatedly until the process is complete
   a. [ACO] Select best genes in the current population. Swap and shuffle the best genes in different combinations of all chromosomes using ACO.
   b. [Selection] Select and save elite in the current population.
   c. [C-SOBS] Convert the chromosome representation to a binary form and perform cyclic mutation for a range of rates using **SOBS.**
   d. [Selection] Select and save elite of the current population.
5. [Test] Check the termination condition. If satisfied, stop, and return the best solution
6. [Loop] Go to step 4, as shown in Figure 1.

3.3. Auto-Poietic Crossover Operator

Two types of operations are inbuilt in the newly designed crossover operator. ACO with dual operation perform a self-organizing swap and shuffle process is proposed to design a new operator. ACO operator chooses the best individuals by calculating the gap penalty for each gene, where, lesser the penalty better the genes. The affine gap penalty model is used to evaluate the fitness evaluation of gene of the chromosomes.

3.3.1. Self-Organizing Swap and Shuffle Operation

- To illustrate the self-organizing crossover operation a chromosome consists of three genes (G1, G2, and G3) is considered as shown in Figure 2. Initially, gap penalty of each gene is calculated to identify the best genes in a population. The Best Genes (BGs) are named as BG1, BG2, and BG3 and saved separately
- *Swap*: in this process, the swap operation replaces the gene G1 with BG1 of all chromosomes of entire population, whereas the other two genes G2 and G3 remain unaltered. In the same way, the swap process is carried for G2 with BG2 and G3 with BG3 individually, thereby creating three different set of population.
- *Shuffle*: the second part of the crossover is to shuffle the best genes collected in the above process. In this operation, the genes G1and G2 are shuffled with best genes BG1 and BG2, as a result the chromosome [BG1||BG2||G3] is generated for the whole population. Similarly, the genes (G2, G3) and (G1, G3) is shuffled with (BG2, BG3) and (BG1, BG3) respectively to get [G1||BG2||BG3] and [BG1||G2||BG3] combination in the population.

As a result of dual crossover operations (swap and shuffle), the best genes are differently combined to create the population set. As an instance, for an input of three chromosomes, the resultant population size is six times more than the initial population. The best individuals produced by the above operation become the current population. Keeping the best genes around to crossover for swap operation favours the better shuffling which may not be possible in the normal crossover. Finally, the fitness score of the

chromosomes are calculated to select the elite. It is evident, that the dual crossover creates a wide range of search space, in turn facilitates the exploration of optimal solution.

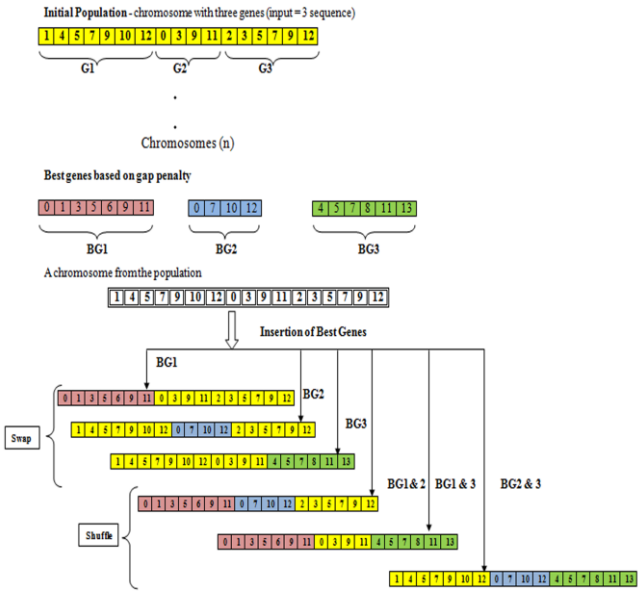

Figure 2. An illustration of auto-poietic crossover.

3.4. Cyclic Self-Organizing Binary Shuffler (C-SOBS)

In SGA, a fixed optimal rate or optional rates are given which may not be suitable for all input. It is too hard for the user to select appropriate rate without prior knowledge of the problem. To eliminate these problems, a new mutation operator with a different approach is discussed. Instead of a fixed rate, the operator performs mutation for a range of rates cyclically [1, 19] till the termination conditions are satisfied.

In default, shuffling process for mutation leads to the problems like:

1. Increase in the number of gaps for a particular sequence
2. Occurrence of repeated gap positions in a sequence

In order to avoid this, the proposed mutation operator converts the chromosome representation from integer to binary digits (1, 0) represents the presence and absence of gaps. The working principle of C-SOBS is as follows:

1. Convert the chromosome representation to a binary form.
2. Initialize minimum optimal mutation rate and the corresponding mutation point is selected.
3. Genes before mutation point are considered for mutation.
4. The genes within each complete point and if any gene occurs between the last complete point and mutation point are shuffled separately as shown in Figure 3.

Change the chromosome representation back to gap positions

1. Generate MSA corresponding to the chromosome.
2. Calculate fitness score.
3. Select elite.

If elite is replaced, the generation continues with same rate else increases cyclically until an optimal upper limit is reached. The algorithm terminates on reaching the optimal upper limit when no further increase in column scores [13].

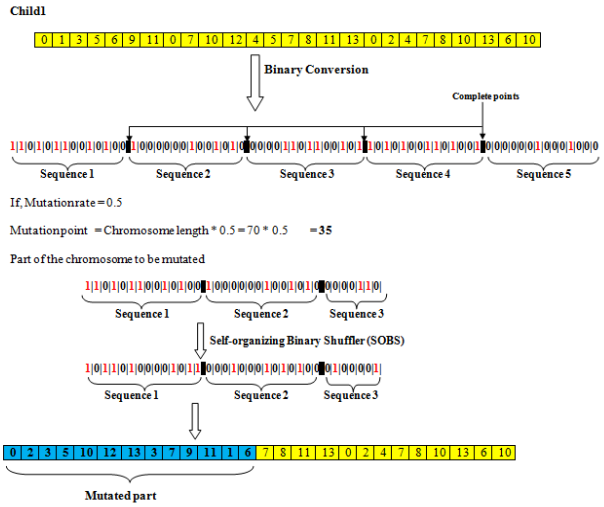

Figure 3. An illustration of self-organizing binary shuffler.

4. Results and Discussion

To validate the proposed algorithm, it is tested using the datasets of standard Benchmark Alignment Database (BAliBASE), OXBench Benchmark suite (OXBench), and a Simple Modular Architecture Research Tool (SMART) [2] were taken and some of them were reported. The number of input sequences ranges from 4 to 16 and its total length vary from 1000 to 7000 are considered. The important parameters like, number of generation and resultant score of the final alignment are calculated and tabulated.

4.1. Comparative Analysis of AP-MSA and SGA

The number of generation is fixed in case of SGA whereas in AP-MSA, it is self-organized based on the betterment of alignment resulting in each generation.

The comparative result shows that, on average, Auto-Poietic algorithm for MSA (AP-MSA) produces better alignments than SGA in less number of generations and time. For RV11_BB11001 dataset with 4 sequences, SGA generated the final alignment scoring 0.016 in 50 generations, whereas, AP-MSA produced a high scoring optimal alignment in much less generations. The results are tabulated Tables 1, 2, and 3 and a graphical representation is shown in Figures 4, 5, and 6.

Table 1. Comparative analysis of performance of AP-MSA and SGA for BAliBASE datasets.

S.No.	Dataset	SGA		AP-MSA	
		No. of Gen	Score	No. of Gen	Score
1	RV11_BB11015	50	0.0127	43	0.0152
2	RV11_BB11021	50	0.016	44	0.02
3	RV11_BB11025	50	0.0135	44	0.0188
4	RV11_BB11032	50	0.0201	42	0.0006
5	RV11_BBS11001	50	0.0129	43	0.0640
6	RV11_BBS11029	50	0.0172	44	0.0207
7	RV12_BB12006	50	0.0195	42	0.0206
8	RV20_BBS20001	50	0.0069	42	0.0036
9	RV11_BBS11022	50	0.0186	45	0.0322
10	RV12_BBS12010	50	0.0038	41	0.0034
11	RV12_BBS12021	50	0.0224	43	0.0174
12	RV12_BBS12034	50	0.0040	42	0.0225
13	RV11_BBS11002	50	0.0011	42	0.0115
14	RV11_BBS11009	50	0.0305	41	0.0280
15	RV11_BB11022	50	0.0183	45	0.0067
16	RV11_BBS11006	50	0.0127	42	0.0034
17	RV11_BB11001	50	0.016	40	0.0397
18	RV11_BBS11008	50	0.0135	44	0.0229

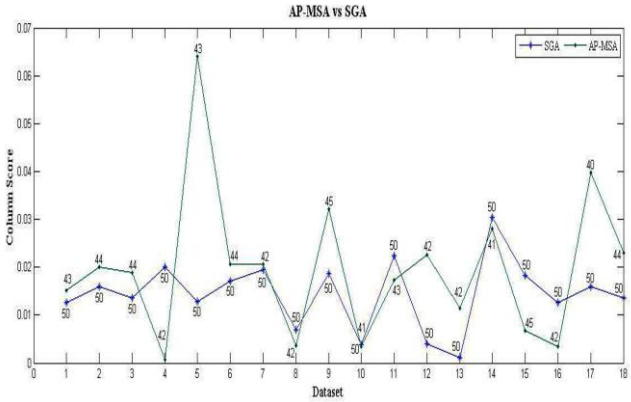

Figure 4. Comparative analysis of performance of AP-MSA and SGA for BAliBASE datasets.

Table 2. Comparative analysis of performance of AP-MSA and SGA for OXBench datasets.

S.No.	Dataset	SGA		AP-MSA	
		No. of Gen	Score	No. of Gen	Score
1	4t3	50	0.0085	42	0.0170
2	10s10	50	0.0017	44	0.0051
3	12s55	50	0.0072	41	0.0120
4	12S83	50	0.0146	40	0.0170
5	22s30	50	0.0209	45	0.0287
6	43s6	50	0.0078	43	0.0089
7	57t7	50	0.0027	44	0.0054
8	82t5	50	0.0042	41	0.0106
9	469	50	0.0985	46	0.1034

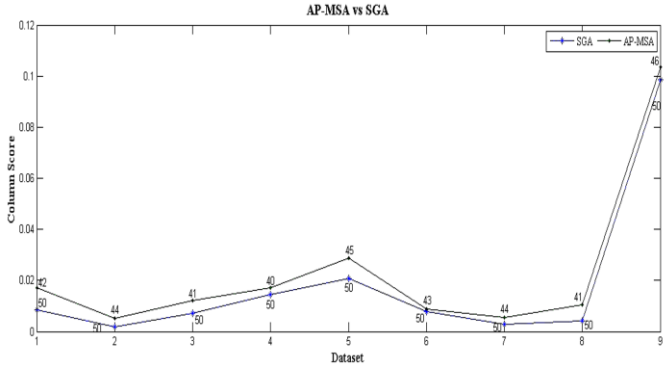

Figure 5. Comparative analysis of performance of AP-MSA and SGA for OXBench datasets.

Table 3. Comparative analysis of performance of AP-MSA and SGA for SMART datasets.

S.No.	Dataset	SGA		AP-MSA	
		No. of Gen	Score	No. of Gen	Score
1	Crf	50	0.0189	44	0.0316
2	Chsh	50	0.0038	42	0.0154
3	Fimac	50	0.0075	45	0.0112
4	Net pep	50	0.0111	43	0.0333
5	Calcitonin	50	0.0324	41	0.0519
6	Clb	50	0.0026	46	0.0065
7	AXH	50	0.0191	44	0.0276
8	NMU	50	0.1777	40	0.2111
9	SORB	50	0.0421	42	0.0526
10	Thy	50	0.0751	43	0.1052

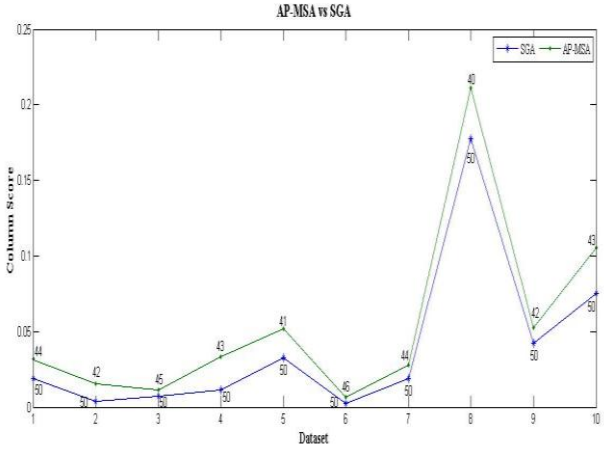

Figure 6. Comparative analysis of performance of AP-MSA and SGA for SMART datasets.

4.2. Performance Analysis on Consecutive Execution of AP-MSA

To analyze the changes in the resulting score and self-organizing convergence point, the algorithm is executed for ten times using model dataset 469 with three sequences from OXBench benchmark alignment suite [2] and the scores, number of generations are tabulated Table 4. A graphical representation indicating changes in the column score against execution is given in Figure 7. This analysis shows that the alignment produced and number of generations varies for various executions. It is evident from the result that best score was generated more number of times, thus, proving the efficiency of the algorithm.

Table 4. Impact of execution on score (AP-MSA).

No. of Execution	No. of Generations	Column Score
1	42	0.0985
2	40	0.0960
3	43	0.1034
4	41	0.1009
5	45	0.0985
6	41	0.1034
7	39	0.1009
8	44	0.1034
9	42	0.1009
10	43	0.1034

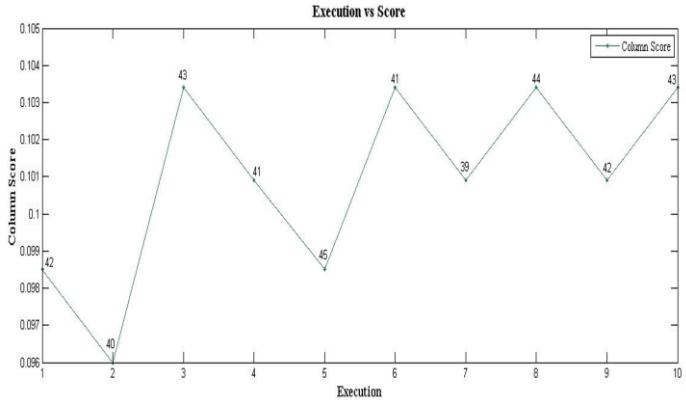

Figure 7. Impact of execution on score (AP-MSA).

4.3. Procedural Analysis of AP-MSA

To explain the self-organizing procedure of AP-MSA, a model dataset 469 with three sequences from OXBench benchmark [2] are used as input. In AP-MSA, the crossover rate (Rc) is random depending on the alignment length and the mutation rate (Rm) is allowed to change from 1% to 80% based on the improvement of alignment score which has an impact on termination of execution thereby number of generations. The execution starts with Rm=1% and generation (1) produced an alignment with Column score 0.0788. In next generation (2) by the concept of self-organization implemented in the procedure Rm increased to 3% resulting a score of 0.0812. Generation (3) retains the same rate Rm=3% resulting a score of 0.0812 and no further increase in the score. Hence by the concept of self-organization Rm is increased to 5%. Self-organizing increase in Rm continues till the upper limit (80%) is reached. Mutation rate reaches the upper limit 81%, when generation=46 and the score of output alignment is 0.1034 as shown in Table 5. A Graph explaining the above procedure with generation against score is given in Figure 8. It is evident from the result data, for 469 dataset, seven improvements occurred for execution/ procedure to obtain the final score of 0.1034.

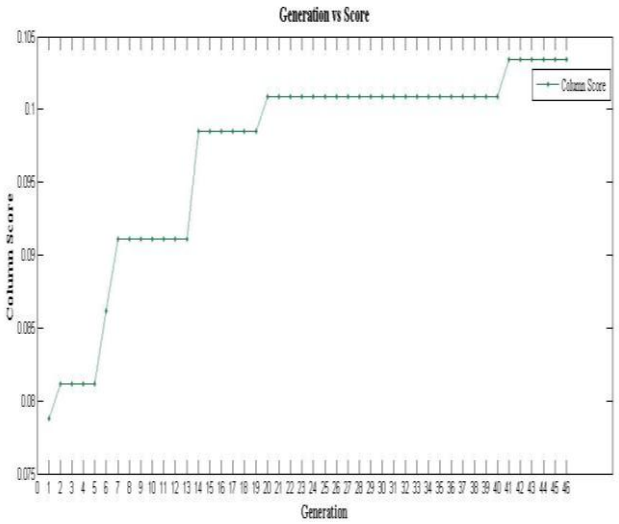

Figure 8. Impact of self-organizing procedure (AP-MSA).

Table 5. Impact of Self-organizing procedure (AP-MSA).

Generation	Rate of Cyclic Mutation Operator (in Percentage)	EM of elite	Column score
1	1	32	0.0788
2	3	33	0.0812
3	3	33	0.0812
4	5	33	0.0812
5	7	33	0.0812
6	9	35	0.0862
7	9	37	0.0911
8	9	37	0.0911
9	11	37	0.0911
10	13	37	0.0911
11	15	37	0.0911
12	17	37	0.0911
13	19	37	0.0911
14	21	40	0.0985
15	21	40	0.0985
16	23	40	0.0985
17	25	40	0.0985
18	27	40	0.0985
19	29	40	0.0985
20	31	41	0.1009
21	31	41	0.1009
22	33	41	0.1009
23	35	41	0.1009
24	37	41	0.1009
25	39	41	0.1009
26	41	41	0.1009
27	43	41	0.1009
28	45	41	0.1009
29	47	41	0.1009
30	49	41	0.1009
31	51	41	0.1009
32	53	41	0.1009
33	55	41	0.1009
34	57	41	0.1009
35	59	41	0.1009
36	61	41	0.1009
37	63	41	0.1009
38	65	41	0.1009
39	67	41	0.1009
40	69	41	0.1009
41	71	42	0.1034
42	71	42	0.1034
43	73	42	0.1034
44	75	42	0.1034
45	77	42	0.1034
46	79	42	0.1034

4.4. Comparison of AP-MSA and Existing MSA Tools

BAliBASE, OXBench and SMART [2] sequences are aligned using other MSA tools namely, Web-based computer program (CLUSTAL) for multiple sequence alignment (CLUSTALW) [18], Multiple Alignment using Fast Fourier Transform (MAFFT) [8], MUltiple Sequence Comparison by Log-Expectation (MUSCLE) [4], Multiple sequence alignment with hierarchical clustering (MULTALIN) [3], Tree-based Consistency Objective Function for Alignment Evaluation (T-COFFEE) [13], DNA or protein ALIGNment program (DIALIGN) [10] with default parameter settings and tabulated in Table 6. The RV11 dataset used consists of sequences with less than 25% identity. The alignment results of Auto-Poietic algorithm for MSA (AP-MSA) and other MSA tools were compared using column score to prove the performance efficiency of developed algorithm. The

results areTable 6. Comparison of AP-MSA with other MSA Tools in Figure 9. The column score of the alignment generated by MULTALIN for RV11_BB11009, RV11_BB11037 and RV11_BB11002 datasets is zero due to the absence of exact match. Similarly, the alignment programs, MAFFT, and DIALIGN, generated alignment without any exact match for datasets RV11_BB11002, and RV11_BB11032, respectively.

Whereas, AP-MSA produced alignments with better column score than the widely used MSA tools considered in the comparative analysis. The ability of AP-MSA to generate optimal or close to optimal alignments for all reference datasets shows its performance efficiency.

Table 6. Comparison of AP-MSA with other MSA tools.

S. No.	DATASET	CLUSTALW	MAFFT	MUSCLE	MULTALIN	T-COFFEE	DIALIGN	AP-MSA
1	RV11_BB11002	0.0046	0	0.0016	0	0.0056	0.0098	0.0038
2	RV11_BB11009	0.0203	0.0188	0.0490	0	0.0177	0.0801	0.0322
3	RV11_BB11015	0.1166	0.0043	0.0849	0.1118	0.1394	0.0604	0.0152
4	RV11_BB11032	0.0119	0.0058	0.0134	0.0024	0.0063	0	0.0006
5	RV11_BB11037	0.0234	0.0124	0.0321	0	0.0350	0.0038	0.0004
6	RV11_BBS11013	0.0459	0.0210	0.0696	0.0282	0.0534	0.0337	0.0146
7	RV11_BBS11022	0.0801	0.0402	0.1058	0.0809	0.0884	0.0218	0.0241

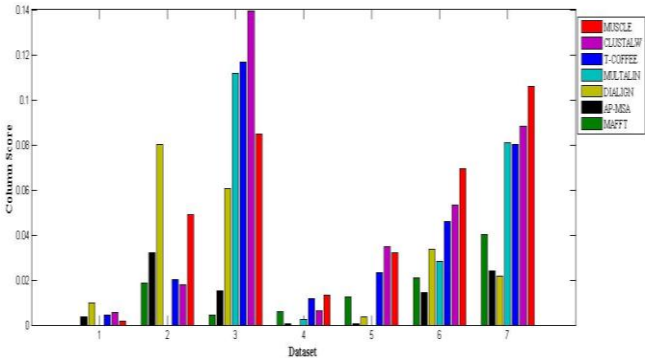

Figure 9. Comparison of AP-MSA with other MSA tools.

5. Conclusions

There are several variants in genetic algorithm modified at genetic operator level and procedural level that ultimately trying to improve the performance of conventional genetic algorithm. This work is an attempt to devise a novel self-organizing GA to explore MSA problem and to enhance the performance of SGA. In contrast to SGA, the parameters are designed to adapt optimal values during execution. This procedure helps non-domain users to avoid execution with fixed parameter values as it may not be appropriate in all circumstances. As an advantage, the developed auto-poietic crossover operator encourages the gene to enhance exploration in order to improve the working efficiency and to analyze the influence of crossover on population diversity. Saving a copy of best chromosome after each genetic operation with elitism selection guarantees the avoidance of disruption of best chromosome is avoided. SOBS increases the mutation rate cyclically based on the improvement in the resultant score, thereby, allowing the process to explore all rates within the range. Incorporating the designed operators, AP-MSA is developed and implemented to solve MSA and it is found to perform better to avoid premature convergence. The comparative analysis of performance with SGA and other widely used MSA tools shows the efficiency and consistency of AP-MSA in producing optimal alignment.

Acknowledgements

We acknowledge the support from the Centre for Bioinformatics, Pondicherry University and UGC [F. No: 42-862/2013dt.25 March 2013].

References

[1] Bao-Juan H., Jian Z., and De-Hong Y., "A Novel and Accelerated Genetic Algorithm," *WSEAS Transactions on Systems and Control*, vol. 3, no. 4, pp. 269-278, 2008.

[2] Carroll H., Beckstead W., O'Connor T., Ebbert M., Clement M., Snell Q., and McClellan D., "DNA Reference Alignment Benchmarks Based on Tertiary Structure of Encoded Proteins," *Bioinformatics*, vol. 23, no. 19, pp. 2648-2649, 2007.

[3] Corpet F., "Multiple Sequence Alignment with Hierarchical Clustering," *Nucleic Acids Research*, vol. 16, no. 22, pp. 10881-10890, 1988.

[4] Djennas S., Bendimerad F., and Meriah S., "Genetic Algorithm-Based Synthesis of Three-Dimensional Microstrip Arrays," *The International Arab Journal of Information Technology*, vol. 2, no. 3, pp. 182-190, 2005.

[5] Edgar R., "MUSCLE: Multiple Sequence Alignment with High Accuracy and High Throughput," *Nucleic Acids Research*, vol. 32, no. 5, pp. 1792-1797, 2004.

[6] Fatumo S., Akinyemi I., and Adebiyi E., "Aligning Multiple Sequence with Genetic Algorithm," *International Journal of Computer Theory and Engineering*, vol. 1, no. 2, pp. 179-182, 2009.

[7] Gondro C. and Kinghorn B., "A simple Genetic Algorithm for Multiple Sequence Alignment," *Genetics and Molecular Research*, vol. 6, no. 4, pp. 964-982, 2007.

[8] Katoh K., Misawa K., Kuma K., and Miyata T., "MAFFT: A Novel Method for Rapid Multiple Sequence Alignment Based on Fast Fourier Transform," *Nucleic Acids Research*, vol. 30, no. 14, pp. 3059-3066, 2002.

[9] Khayat O., Ebadzadeh M., Shahdoosti H., Rajaei R., and Khajehnasiri I., "A Novel Hybrid Algorithm for Creating Self-Organizing Fuzzy Neural Networks," *Neurocomputing*, vol. 73, no. 1, pp. 517-524, 2009.

[10] Kubota N., Fukuda T., and Shimojima K., "Virus Evolutionary Genetic Algorithm for ASelf-Organizing Manufacturing System," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 1015-1026, 1996.

[11] Mansour N., Awad M., and El-Fakih K., "Incremental Genetic Algorithm," *The International Arab Journal of Information Technology*, vol. 3, no. 1, pp. 42-47, 2006.

[12] Morgenstern B., Dress A., and Werner T., "Multiple DNA and Protein Sequence Based on Segment-to-Segment Comparison," *in Proceedings of the National Academy of Sciences*, vol. 93, no. 22, pp. 12098-12103, 1996.

[13] Nizam A., Shanmugham B., and Subburaya K., "Self-Organizing Genetic Algorithm for Multiple Sequence Alignment," *Global Journal of Computer Science and Technology*, vol. 11, no. 7, pp. 7-14, 2011.

[14] Nizam A. and Shanmugham B., "Self-Organizing Multi-mutation Algorithm for Multiple Sequence Alignment," *in Proceedings of 15th International Conference on Advanced Computing Technologies*, Rajampet, 2013.

[15] Notredame C., Higgins D., and Heringa J., "T-Coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment," *Journal of Molecular Biology*, vol. 302, no. 1, pp. 205-217, 2000.

[16] Notredame C., "Recent Progresses in Multiple Sequence Alignment: A Survey," *Pharmacogenomics*, vol. 3, no. 1, pp. 131-144, 2002.

[17] Seeley T., "When Is Self-Organization Used in Biological Systems?," *The Biological Bulletin*, vol. 202, no. 3, pp. 314-318, 2002.

[18] Thompson J., Higgins D., and Gibson T., "CLUSTAL W: Improving the Sensitivity of Progressive Multiple Sequence Alignment through Sequence Weighting Position Specific Gap Penalties and Weight Matrix Choice," *Nucleic Acids Research*, vol. 22, no. 22, pp. 4673-4680, 1994.

[19] Tin´os R. and Yang S., "Self-Organizing Random Immigrants Genetic Algorithm for Dynamic Optimization Problems," *Genetic Programming and Evolvable Machines*, vol. 8, no. 3, pp. 255-286, 2007.

[20] Wu S., Lee M., Lee Y., and Gatton T., "Multiple Sequence Alignment Using GA and NN," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 1, no. 1, pp. 21-30, 2008.

[21] Zhang J., Zhuang J., Du H., and Wang S., "Self-Organizing Genetic Algorithm Based Tuning of PID Controllers," *Information Sciences*, vol. 179, no. 7, pp. 1007-1018, 2009.

Amouda Venkatesan is currently working as an Assistant Professor in the Centre for Excellence in Bioinformatics, Pondicherry University, India. She received the master degree in Software System from BITS Pilani and Doctorate degree in Computer Science and Engineering from Pondicherry University. She has more than 12 years of teaching experience. Her principle area of interests is Genetic algorithm and published more than 25 research publications.

Buvaneswari Shanmugham is pursuing Ph.D. Bioinformatics in the Centre for Excellence in Bioinformatics, Pondicherry University, India. She received her M.Sc. degree in Bioinformatics in the year 2010. She has published six research papers in International journals and conferences. Her area of research includes Genetic algorithm and Comparative genomics.