IBBOMSA: An Improved Biogeography-based Approach for Multiple Sequence Alignment

Rohit Kumar Yadav[1] and Haider Banka[2]

[1]Research Scholar, Department of Computer Science and Engineering, Indian Institute of Technology (ISM) Dhanbad, Jharkhand, India.
[2]Associate Professor, Department of Computer Science and Engineering, Indian Institute of Technology (ISM) Dhanbad, Jharkhand, India.

ABSTRACT: In bioinformatics, multiple sequence alignment (MSA) is an NP-hard problem. Hence, nature-inspired techniques can better approximate the solution. In the current study, a novel biogeography-based optimization (NBBO) is proposed to solve an MSA problem. The biogeography-based optimization (BBO) is a new paradigm for optimization. But, there exists some deficiencies in solving complicated problems such as low population diversity and slow convergence rate. NBBO is an enhanced version of BBO, in which, a new migration operation is proposed to overcome the limitations of BBO. The new migration adopts more information from other habitats, maintains population diversity, and preserves exploitation ability. In the performance analysis, the proposed and existing techniques such as VDGA, MOMSA, and GAPAM are tested on publicly available benchmark datasets (ie, Bali base). It has been observed that the proposed method shows the superiority/competitiveness with the existing techniques.

KEYWORDS: Multiple sequence alignment (MSA), biogeography-based optimization (BBO), migration operator, diversity

CITATION: Yadav and Banka. IBBOMSA: An Improved Biogeography-based Approach for Multiple Sequence Alignment. *Evolutionary Bioinformatics* 2016:12 237–246 doi: 10.4137/EBO.S40457.

TYPE: Original Research

RECEIVED: June 24, 2016. **RESUBMITTED:** August 10, 2016. **ACCEPTED FOR PUBLICATION:** August 16, 2016.

ACADEMIC EDITOR: Liuyang Wang, Associate Editor

PEER REVIEW: Three peer reviewers contributed to the peer review report. Reviewers' reports totaled 474 words, excluding any confidential comments to the academic editor.

FUNDING: Authors disclose no external funding sources.

COMPETING INTERESTS: Authors disclose no potential conflicts of interest.

CORRESPONDENCE: rohit.ism.123@gmail.com

Introduction

More than three amino acid sequences or protein sequence alignment at a time is called multiple sequence alignment (MSA). MSA is the most important tool to solve biological problems. We can solve lots of problem in biology by using MSA. MSA helps to predict the secondary and tertiary structures of RNA and proteins.[1,2] We can reconstruct phylogenetic trees using MSA, which can predict the function of an unknown amino acid by aligning its sequences with some other known functions. We can also find similarity of the sequences using MSA, which can help to define similarity in functions and structures.[3,4] In order for an MSA to be valid, entire sequences in the multiple alignments must have a common origin. The goal of MSA is to maximize the matching of protein or amino acid as far as possible.[5] Therefore, MSA is an important problem in bioinformatics to study the genetic and phylogenetic relationship. There are several methods to solve an MSA problem in the past.

The MSA problem can be solved and an optimal alignment can be achieved by using dynamic programming (DP). DP uses a scoring function that contains a large domain. In 1970, Needleman and Wunsch[6] proposed the use of DP algorithm to solve the problem of two sequence alignments. But the problem behind the use of DP is that when the number and length of sequence are increased, its complexity also increases in an exponential manner. Then, the MSA problem becomes NP-hard. Since complexity is the main constraint for the computer to solve any problem, we have to maximize the matching of protein or amino acid sequence in limited time or less complexity. This is the major reason why researchers switch to other methods.

The MSA problem can be also solved using progressive method. The progressive approach takes less complexity in terms of time and space for solving an MSA problem.[7,8] According to progressive alignment method, initially align more similar sequences and then incrementally align more divergent sequences or group of sequences in the initial alignment. The standard representative of progressive methods is CLUSTALW.[9] In the first step, according to this approach, we have to assign the weight of each pair of sequences in a partial alignment. We assign small weight for most similar sequences and big weight for most divergent sequences. After that, we take substitution matrix that defines the score between two residues of protein sequence based on similarity. Two types of gap have been introduced in the third step. The first one is residue-specific gap and the second one is locally residue gap penalties. In the fourth step, gap that has been introduced in early position receives locally reduced gap penalties to encourage the opening gap at these positions. These four steps are integrated into CLUSTALW, which is freely available. Progressive alignment method performs better for MSA package in terms of accuracy and time. Even this method has some limitation. The problem behind this method is dependency on initial alignment and choice of scoring scheme. In other words, we bound that to align more similar sequences in the initial stage. If we have not aligned more similar sequences in the

initial stage, then the solution may be trapped in local optima. An iterative method is another option for solving MSA.

An iterative method does not depend on initial alignment because it starts with initial alignment and improves the solutions per iteration until no more improvement is possible. The main objective of the iterative approach for MSA is to globally improve the quality of a sequence alignment. There are some iterative and stochastic approaches for MSA (for example, simulated annealing[10,11]). Hidden Markov Models Training (HMMT)[12] is based on a simulated annealing process. The problem behind the solution recommended by these methods may be trapped in local optima.

Evolutionary algorithms[13,14] are population-based algorithms. According to these algorithms, we generate random initial population in the first step. In the next step, we apply some operators to modify the initial population for next generation. We repeatedly use these operators until we reach the global optimum. When using Evolutionary Algorithms (EAs) for an MSA, an initial generation is generated by random manner, and then, the steps of an EA are applied to improve the similarities among the sequences. There are some evolutionary computations for MSA.[15–19] There are some other genetic algorithm (GA)-based methods for MSA, such as SAGA,[19] GA-ACO,[20] MSA-EC,[21] MSA-GA,[22] RBT-GA,[23] GAPAM,[24] VDGA,[25] and MOMSA.[26] We define methodology of some algorithm to solve an MSA problem based on GA. In SAGA, the initial generation is generated randomly. According to SAGA, 22 different operators are used to gradually improve the fitness of MSA. But the problem behind SAGA is time complexity due to repeated use of fitness function. RBT-GA is also a GA-based method, combined with the rubber band technique (RBT), to find optimal protein sequence alignments.[27] RBT[28] is an iterative algorithm for sequence alignment using a DP table. The authors[26] solved 56 problems from reference sets 1, 2, 3, 4, and 5 of the benchmark Bali base 2.0 dataset and Bali base 3.0 dataset. The drawbacks of these evolutionary methods are also local optima due to poor diversity of the solutions.

Motivation and contributions. In the domain of biology, MSA is the most crucial to solve numerous standard problems such as structure prediction and phylogenetic property. According to the open literature, the MSA is still an open-challenging problem. Hence, we motivate to solve an MSA problem using the improved version of biogeography-based optimization (BBO). However, this paper achieves the following contributions.

a. We first proposed a method to improve migration operator in BBO and then used it in MSA for maintaining diversity of the solutions.
b. The results obtained in experimental analysis are better in terms of time factor. In addition, we provide a comparison table, which claims that our method is better than the existing competitive solutions in terms of matching score.

Biogeography-Based Optimization

BBO[29] was designed by emigration and immigration of species from one habitat to another. In the BBO algorithm, candidate solutions are called habitats (or islands). Each feature in a solution represented by a habitat is called a suitability index variable (SIV), while the goodness of a habitat is measured by the habitat suitability index (HSI). Habitats with a high HSI can support more species, whereas low HSI habitats support only a few species. Poor habitats can improve their HSI by accepting new features from more attractive habitats in the evolution process.

In BBO, there are two main operators: migration and mutation. The migration operator is a probabilistic operator that can randomly modify SIVs based on the immigration rate λ_i and emigration rate μ_i. Both λ_i and μ_i are functions of the number of species in the ith habitat (Hi). In the original BBO algorithm, for mathematical convenience, μ_i and λ_i are assumed to be linear with the same maximum values, which means that the immigration rate λ_i and emigration rate μ_i are linear functions of the number of species. The linear migration model for the ith habitat (Hi) can be calculated as

$$\lambda_i = I * (1 - n_i)/n$$
$$\mu_i = E * n_i / n \quad (2.1)$$

where E is the maximum possible emigration rate, I is the maximum possible immigration rate, n_i is the number of species in the ith habitat, and n is the maximum number of species. The complete process of BBO is given in Algorithm 1.

Algorithm 1. Main procedure of BBO

```
1  Begin
2  Initialize the population Pop with N habitats randomly
3  Evaluate the fitness (HSI) for each Habitat in Pop
4  while (criteria of termination not satisfied)
5  Map the HSI to the number of species count S for each habitat
6  Calculate the immigration rate and emigration rate according
   to migration model
7  Modify habitats with the migration operator (algorithm 2)
8  Mutate habitats with mutation operator (algorithm 3)
9  End While
10 End
```

In BBO, the migration operator is a probabilistic operator that is used to randomly adjust each habitat Hi by sharing features among them. The probability that Hi is modified is proportional to its immigration rate λ_i, while the probability that the source of the modification comes from Hj is proportional to the emigration rate μ_j. The migration equation is expressed as

$$Hi(\text{SIV}) = Hj(\text{SIV}) \quad (2.2)$$

where Hi(SIV) denotes the feature (SIV) of the ith habitat Hi.

As Simon stated, the migration operator merely migrates SIVs from one solution to another and does not involve reproduction of "children".[29] The migration operator algorithm process is shown in Algorithm 2.

Algorithm 2. Migration operator	
1	***Begin***
2	***For*** *i = 1 to N*
3	***If*** *rand (0,1)* $< \lambda_i$
4	*Hi is selected*
5	***End If***
6	***For*** *j = 1 to N*
7	***If*** *rand (0,1)* $< \mu j$
8	*Hi(SIV) = Hj (SIV)*
9	***End If***
10	***End For***
11	***End For***
12	***End***

Cataclysmic events can cause a species count to differ from its equilibrium value, thereby suddenly changing a habitat's HSI. We model this sudden operation in BBO as mutation. The SIVs of the *i*th habitat *Hi* can be randomly modified by the mutation operator according to the habitat's priori probability P_i. The mutation probability m_i of the *i*th habitat *Hi* is expressed as

$$m_i = m_{\max} * (1 - P_i / P_{\max}) \tag{2.3}$$

where $m_{\max}$ is a user-defined parameter and $P_{\max} = \max(P_i)$, $i = 1, 2. N$. In the BBO mutation operator, an SIV in each habitat is randomly replaced by a new feature, randomly and probabilistically generated in the entire solution space, which tends to increase population diversity. The process of mutation operator is given in Algorithm 3.

Algorithm 3. Mutation operator	
1	***Begin***
2	***For*** *i = 1 to N*
3	*Use* μ *to compute the probability Pi*
4	***If*** *rand (0, 1)* $< P_i$
5	*Hi is selected*
6	*Hi(SIV) = Random Value generated within the search space*
7	***End if***
9	***End for***
10	***End***

Proposed Method

Habitat representation. In BBO, each solution is represented as habitat.

$$X_i = \left(X_i^1, \ldots, X_i^d, \ldots, X_i^n \right) \quad \forall 1 \le i \ge N \tag{3.1.1}$$

where *N* is the number of habitats.

In the initialization state, first put the gap in our given MSA randomly. The initial solution is given in Figure 1.

Binary encoding scheme: In the encoding scheme, put 1 in the position of gap and put 0 in the position of protein sequences. Figure 2 shows an encoding of initial solution.

After that, we are taking decimal value of this binary encoded value from bottom to top of each column. Hence, habitat representation of this solution is $X_1 = (1, 0, 0, 8, 2, 4)$ and the number of columns in the MSA is equal to the number of features in the habitat. Now in this manner, we can generate 100 number of solutions putting gap in MSA. Hence, we can find 100 habitats in initialization.

Fitness function. The sum of pair is used to measure fitness of MSAs. Here, each column in an alignment is scored by summing the product of the scores of each pair of symbols. The score of the entire alignment is then summed over all column scores by using (3.2.1) and (3.2.2).

$$W = \sum_{i=1}^{P} W_i, \text{ where } W_i = \sum_{l=1}^{N-1} \sum_{k=l+1}^{N} \text{Cost}(A_l, A_k) \tag{3.2.1}$$

Here, *W* is the cost of MSAs. *P* is the length (columns) of the alignment, W_i is the cost of the *i*th column of length *P*, *N* is the number of sequences, Cost(A_l, A_k) is the alignment score between two aligned sequences A_l and A_k. When $A_l \neq$ "_" and $A_k \neq$ "_", then Cost(A_l, A_k) is determined from the percentage of acceptable mutations matrix. Also when $A_l =$ "_" and $A_k =$ "_" then Cost$(A_l, A_k) = 0$. Finally, the cost function Cost(A_l, A_k) includes the sum of the substitution costs of the insertion/deletions when $A_l =$ "_" and $A_k \neq$ "_" or $A_l \neq$ "_" and $A_k =$ "_" using a model with affine gap penalties as shown in (Eq. 3.2.2).

$$Z = Q + Ar. \tag{3.2.2}$$

Here, *Z* is the gap penalty, *Q* is the cost of opening a gap, *r* is the cost of extending the gap, and *A* is the number of the gap. In this paper, gap penalties (gap opening penalty is −5 and the gap extension penalty is −0.40).

New solution generation. In this process, two types of operators are used, one is migration and the other is mutation. To improve the solution, low HSI solution accepts the species from the high HSI solution. The entire process is called as migration.

Migration. Migration is used to diversify the solution space or to explore the solution search space, whereas mutation intensifies the solution search space. In each iteration, we are applying migration and mutation operators to the habitats. In the migration process, we share the feature of high HSI habitat

C	G	A	–	G	T
A	T	G	T	C	–
T	G	T	T	–	T
–	C	C	A	T	C

Figure 1. Initial solution.

to low HSI to improve the solution quality. This operator is very effective, and the resultant habitat is much more different from the actual habitat. We chose two habitats according to immigration and emigration rates. Afterward, one index was chosen randomly in emigration habitat, and this SIV/element goes to the same position of immigration habitat. This process is presented in Figure 3.

Mutation. This operator is not much more effective, and the difference between actual habitat and resultant habitat is very less. This operator is not frequent and intensifies the solution of search space. In this operator, one habitat is chosen based on mutation probability. Afterward, one index is chosen randomly of this habitat, and put one new SIV/element between 0 and 2^N (where N is the total number of sequences in MSA) in place of this element. The graphical representation of this process is shown in Figure 4.

Algorithm 4. Main procedure of IBBOMSA	
1	***Begin***
2	*Initialize the population with N habitats randomly*
3	*Evaluate the fitness (HSI) for each Habitat in initial population*
4	***While** (termination criteria are not satisfied)*
5	*Map the HSI to the number of species count S for each habitat*
6	*Calculate the immigration rate and emigration rate using a migration model*
8	*Modify habitats with the improved migration operator (**algorithm 2**)*
9	*Mutate habitats (**algorithm 3**)*
11	***End While***
12	***End***

	0	0	0	1	0	0
	0	0	0	0	0	1
	0	0	0	0	1	0
	1	0	0	0	0	0
Habitat →	1	0	0	8	2	4

Figure 2. Encoding scheme.

Algorithm 5. Improved migration operator	
1	***Begin***
2	***For** I = 1 to N*
3	***If** rand (0,1) < λ_i*
4	*Hi is selected*
5	***End If***
6	***For** j = 1 to N*
7	*Generate two different integers p1 and p2 in {1, N}*
8	***If** rand (0,1) < μj*
9	*Hj is selected*
10	$H_i(SIV) = H_j(SIV) - F * (H_{p1}(SIV) + H_{p2}(SIV))$
11	***End If***
12	***End For***
13	***End For***
14	***End***

Algorithm 6. Mutation operator	
1	***Begin***
2	***For** I = 1 to N*
3	*Use μ to compute the probability P_i*
4	***If** rand (0,1) < P_i*
5	*Hi is selected*
6	*Hi(SIV) = Random Value generated within the search space*
7	***End if***
9	***End for***
10	***End***

Test Dataset

We have tested a large number of datasets from Bali base benchmark database to check the quality of our approach. Bali base version 1.0[30] contains 142 reference alignments, which keeps more than 1000sequences. Bali base version 2.0[31] is an extended version of Bali base version 1.0. Bali base version 2.0 contains 167 reference alignments, which keeps more than 2100sequences. Bali base version 2.0 contains eight reference sets. Each reference set keeps different types of

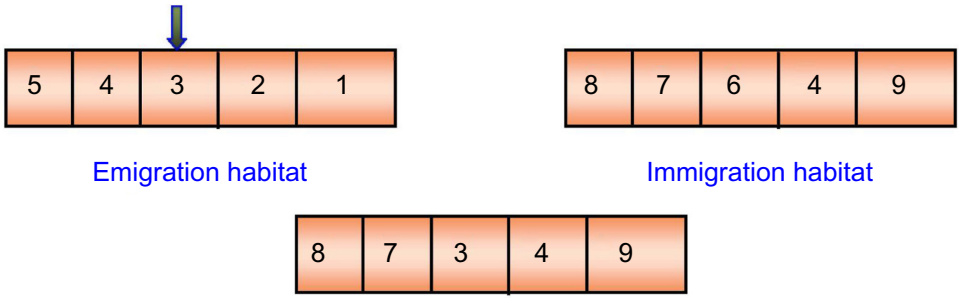

Figure 3. Graphical representation of migration process.

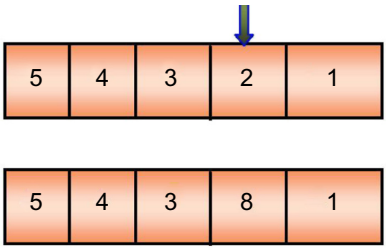

Figure 4. Graphical representation of mutation process.

sequences. Reference set 1 contains a small number of equidistance sequences. Reference set 2 contains totally different or unrelated sequence. Reference set 3 contains a pair of divergent subfamilies. Reference set 4 contains long terminal extension sequence. Reference set 5 contains large internal insertions and deletions. Finally, reference sets 6–8 contain test case problems where the sequences are repeated and the domains are inverted. Bali score is a score that measures the quality of algorithm. Bali score compares between manual alignment sequence (which is available on Bali base version 2.0) and alignment (which comes from some existence method). Range of Bali score is 0–1. If the manual alignment file and our output file are the same, then the score is 1. If the manual alignment file and our output file are totally different, then the score is 0. It gives the value between 0 and 1 according to similarity between Bali base manually alignment file and our output file.

Experimental Analyses

In this section, first, we compare IBBOMSA with the recently proposed MSA algorithms based on evolutionary algorithms, including VDGA,[23] GAPAM,[22] and MOMSA[24] to prove its dominance. After that, we also compare the performance

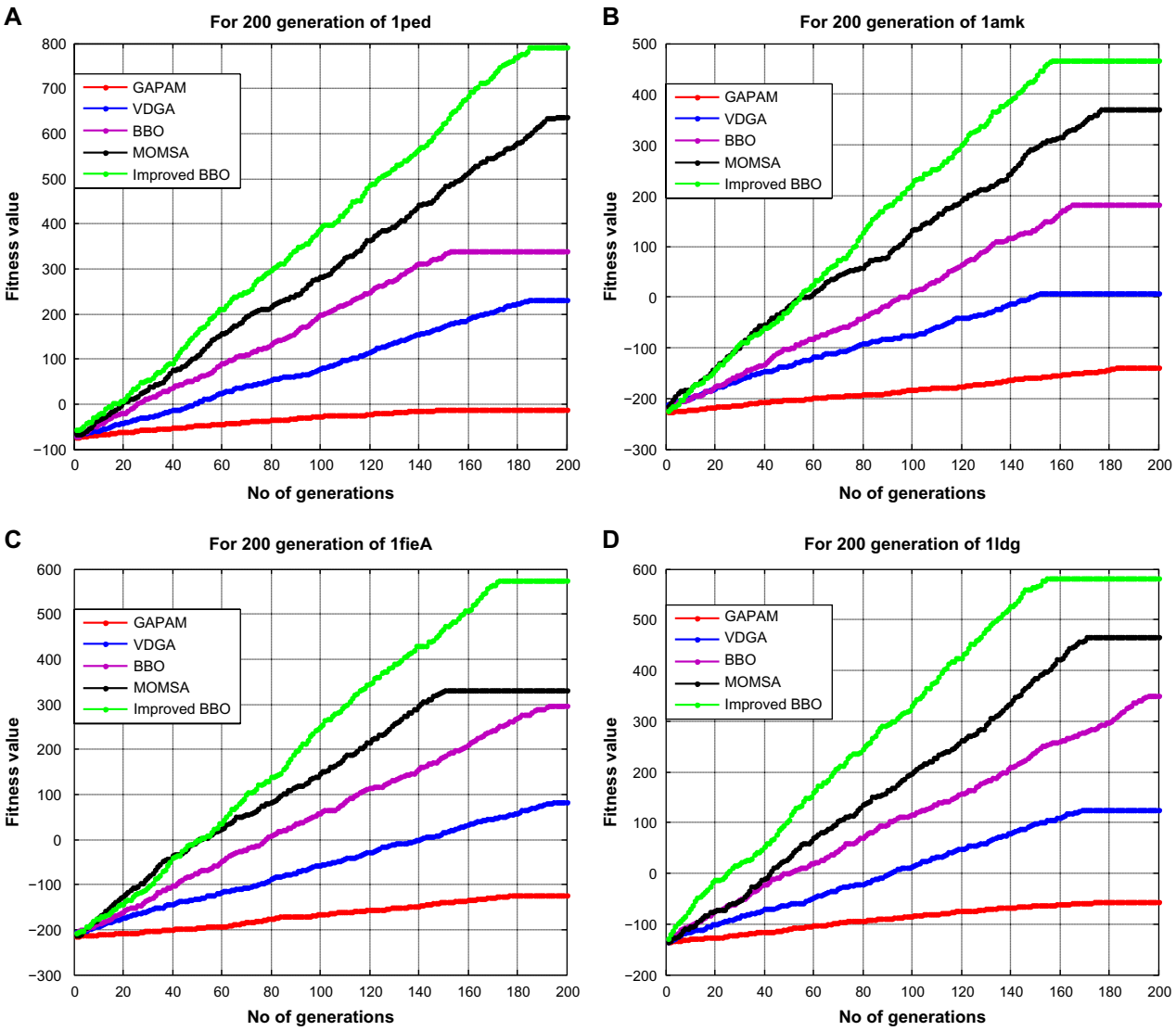

Figure 5. Performance of improved BBO and some existing methods per generation with respect to reference set 1. (**A**) Performance of proposed method and other existing methods with respect to 1ped Data. (**B**) Performance of proposed method and other existing methods with respect to 1amk Data. (**C**) Performance of proposed method and other existing methods with respect to 1fieA Data. (**D**) Performance of proposed method and other existing methods with respect to 1ldg Data.

of IBBOMSA with many well-liked aligners. In this paper, IBBOMSA is coded in C language and implemented in the personal computer in Linux platform.

Effect of improved operator in BBO. The BBO algorithm was invented for immigration and emigration of species between habitats in multidimensional search space. Each habitat represents a solution. In traditional BBO, migration features of good solution appear in poor solution as a new feature while still remaining in good solution. Since this feature may exist in several number of solutions, this may increase the exploitation capability and decrease the diversity of search space. An improved migration with in updated feature appears in poor solution, where updated features come from our proposed migration operator. We used one scaling function for maintaining the exploration (diversity) and exploitation capability. But we have to use this scaling function in a proper way to maintain diversity and exploitation capability. If $F = 0$, it is similar to traditional BBO. Hence if $F = 0$, diversity of search space is decreasing and exploitation capability is decreasing. If $F = 1$, diversity of search space is increasing and exploitation capability is increasing. For maintaining these two things, we have taken $F = 0.5$. To analyze the effect of this proposed operator on the algorithms performance, we have designed five set of experiments. In this set, GAPAM, VDGA, BBO, MOMSA, and improved BBO were run. We measure the fitness of each habitat according to fitness function, which is given in "Fitness function" section. We have used eight BAliBASE datasets for these experiments (4 from each of reference sets 1 and 2, which are illustrated in Figs. 5 and 6, respectively).

Experimental results and analysis. Comparison of IBBOMSA with MOMSA, VDGA, and GAPAM. In order to examine the performance of our proposed method, IBBOMSA, we compare with well-known existence methods such as VDGA,[23] GAPAM,[22] and MOMSA,[24] which are

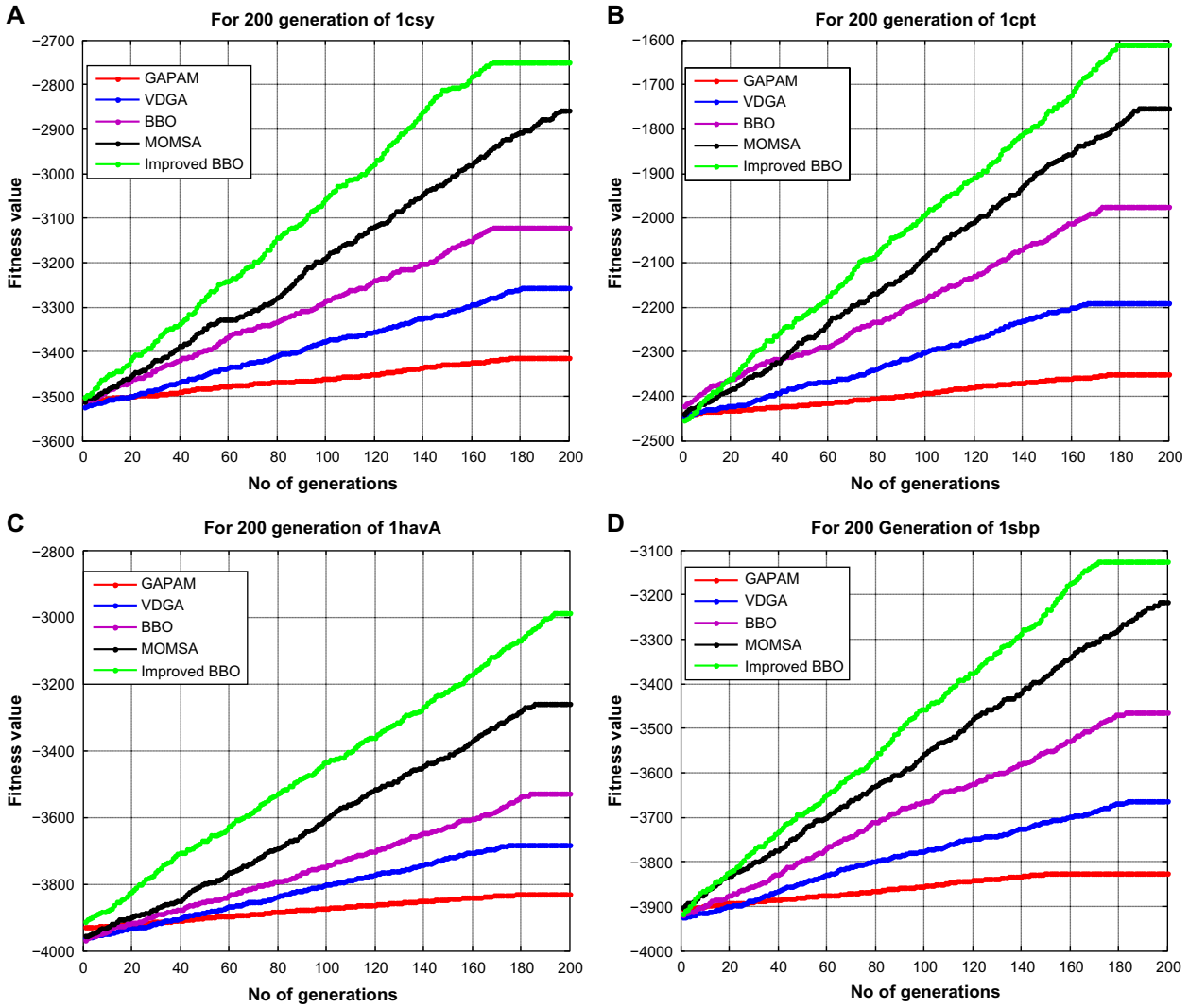

Figure 6. Performance of improved BBO and some existing methods per generation with respect to reference set 2. (**A**) Performance of proposed method and other existing methods with respect to 1csy Data. (**B**) Performance of proposed method and other existing methods with respect to 1cpt Data. (**C**) Performance of proposed method and other existing methods with respect to 1havA Data. (**D**) Performance of proposed method and other existing methods with respect to 1sbp Data.

Table 1. Result of IBBOMSA, MOMSA-W, VDGA, and GAPAM on Bali base reference set 1.

NAME	SEQ NUMBER	SEQ LENGTH	GAPAM[22]	VDGA[23]	MOMSA[24]	IBBOMSA
1idy	50	58	0.5650	0.5730	0.2154	**0.5745**
1tvxA	4	69	0.3160	0.2670	0.0526	**0.4234**
1uky	4	220	0.4020	0.4490	0.5148	**0.5879**
kinase	5	276	0.4870	0.5450	**0.8496**	0.7834
1ped	3	374	0.4980	0.4820	0.7389	**0.8269**
2myr	4	474	0.3170	0.3590	0.4372	**0.4678**
1ycc	4	116	0.8450	0.7550	**0.9345**	0.8269
3cyr	4	109	**0.9110**	0.8210	0.8154	0.8934
1ad2	4	213	0.9560	0.9410	**0.9562**	0.9279
1ldg	4	675	0.9630	0.9060	**0.9886**	0.8256
1fieA	4	442	0.9630	0.9300	0.9820	**0.9852**
1sesA	5	63	0.9820	0.9620	0.9583	**0.9929**
1krn	5	82	0.9600	0.9600	**1.0000**	0.9286
2fxb	5	63	0.9700	0.9780	0.9357	**0.9798**
1amk	5	258	**0.9980**	0.9840	0.9947	0.9456
1ar5A	4	203	**0.9740**	0.9380	0.9604	0.9238
1 gpb	5	828	0.9830	0.9840	0.9862	**0.9889**
1taq	5	928	0.9450	**0.9590**	0.9477	0.9125
Avg. score	–	–	0.7797	0.7662	0.7926	0.8219

Table 2. Result of IBBOMSA, MOMSA-W, VDGA, and GAPAM on Bali base reference set 2.

NAME	SEQ NUMBER	SEQ LENGTH	GAPAM[22]	VDGA[23]	MOMSA[24]	IBBOMSA
1aboA	15	80	0.7960	0.6910	0.8398	**0.8425**
1idy	19	60	0.9890	**0.9920**	0.9743	0.9270
1csy	19	99	0.7640	**0.8850**	0.8536	0.8576
1r69	20	76	0.9650	0.8340	0.9450	**0.9789**
1tvxA	16	69	0.9200	0.9740	0.9365	**0.9819**
1tgxA	19	71	0.8780	0.8780	0.9522	**0.9628**
1ubi	15	60	0.7670	0.7780	**0.9211**	0.8967
1wit	20	106	0.8510	0.8150	**0.9203**	0.9119
2trx	18	94	0.9860	0.9860	**0.9863**	0.9468
1sbp	16	262	0.7650	0.7720	0.8808	**0.9226**
1havA	26	242	0.8790	0.8460	0.8969	**0.8997**
1uky	23	225	0.8080	0.8910	0.9404	**0.9525**
2hsdA	20	255	0.7960	0.8290	0.9192	**0.9249**
2pia	16	294	0.8280	0.8500	**0.9733**	0.9345
3grs	15	237	0.7460	0.7510	0.8492	**0.8719**
kinase	18	287	0.7990	0.8880	0.9397	**0.9452**
1ajsA	18	389	0.8990	0.9050	0.9015	**0.9110**
1cpt	15	434	0.8750	0.8120	0.8862	**0.8943**
1lvl	23	473	0.7810	0.8190	**0.9462**	0.9268
1pamA	18	511	0.8600	0.8630	0.9581	**0.9719**
1ped	18	388	0.9120	0.9470	0.9717	**0.9779**
2myr	17	482	0.8220	0.8300	0.9659	**0.9618**
4enl	17	440	0.8960	0.8890	0.9151	**0.9201**
Avg. score	–	–	0.8513	0.8576	0.9249	0.9270

Table 3. Result of IBBOMSA, MOMSA-W, VDGA, and GAPAM on Bali base reference set 3.

NAME	SEQ NUMBER	SEQ LENGTH	GAPAM[22]	VDGA[23]	MOMSA[24]	IBBOMSA
1idy	27	60	0.6010	0.5990	0.4600	**0.6025**
1r69	23	78	0.7090	0.7330	0.8784	**0.8879**
1ubi	22	97	0.3860	0.4140	0.6606	**0.7107**
1wit	19	102	0.7580	0.8730	**0.8895**	0.7935
1uky	24	220	0.4680	0.4810	0.6393	**0.6634**
kinase	23	287	0.8280	0.8900	**0.8912**	0.8345
1ajsA	28	396	0.3110	0.4530	0.5422	**0.5754**
1pamA	19	511	0.8350	0.7880	**0.9236**	0.8689
1ped	21	388	0.8130	0.8930	0.9131	**0.9240**
2myr	21	482	0.5130	0.6510	0.7278	**0.7464**
4enl	19	427	0.8000	0.8660	0.8158	**0.8698**
Avg score.	–	–	0.6383	0.6946	0.7583	0.7706

Table 4. Result of IBBOMSA, MOMSA-W, VDGA, and GAPAM on Bali base reference set 4.

NAME	SEQ NUMBER	SEQ LENGTH	GAPAM[22]	VDGA[23]	MOMSA[24]	IBBOMSA
1dynA	6	848	0.0330	0.0330	0.8000	**0.8978**
kinase2	18	468	0.3840	0.5420	**1.0000**	0.8426
Avg. score	–	–	0.2085	0.2875	0.9000	0.8702

Table 5. Result of IBBOMSA, MOMSA-W, VDGA, and GAPAM on Bali base reference set 5.

NAME	SEQ NUMBER	SEQ LENGTH	GAPAM[22]	VDGA[22]	MOMSA[24]	IBBOMSA
2cba	8	328	0.8520	0.8350	**0.9875**	0.8687
s51	15	301	0.8350	0.7430	**0.9814**	**0.9829**
Avg. score	–	–	0.8435	0.7890	0.9844	0.9258

the best methods for MSA in recent time. We have taken a selected dataset from MOMSA for comparing our proposed method to other methods in an appropriate manner. The authors chose 56 test cases in Bali base 2.0, which contains 18 test cases from reference set 1, 23 test cases from reference set 2, 11 test cases from reference set 3, and 2 test cases from reference sets 4 and 5, respectively. Calculation of fitness function of MSA is given in "Fitness function" section, and the fitness value of the corresponding MSA is calculated. IBBOMSA is performed 10 times, and the best of their results are recorded. Tables 1–5 show the results of IBBOMSA, MOMSA, VDGA, and GAPAM on Bali base reference sets 1, 2, 3, 4, and 5, respectively.

Comparison of IBBOMSA with MOMSA. MOMSA was recently developed for MSA, which is based on multiobjective optimization. MOMSA method has the ability to develop more than one solution at a time. The authors of MOMSA have described related results with many of the alignment algorithms. The proposed method, IBBOMSA, also has the ability to develop more than one solution at a time. For assessment of both algorithms, we have taken all the datasets of BAliBASE version 2.0 and 3.0. Tables 6 and 7 show average SP and TC scores obtained by these two algorithms based on every group of test cases of BAliBASE versions 2.0 and 3.0. The values of SP and TC scores obtained by MOMSA are reported in Ref. 24. From Table 2, we can say that the proposed IBBOMSA performed better than in most of the cases in both terms, SP and TC scores, in BAliBASE version 2.0. From Table 7, we can also say that the proposed IBBOMSA outperforms in terms of SP and TC scores in BAliBASE version 3.0.

Comparison of IBBOMSA with the state-of-the-art alignment algorithms. In order to prove the accuracy of our proposed IBBOMSA algorithm, we compare the proposed method with some of the widely used alignment algorithms such as MSAProbs,[30] Probalign,[31] MAFFT,[32] Procons,[33] Clustal Omega,[34] T-Coffee,[35] Kalign,[36] MUSCLE,[37] FSA,[38] DIALIGN,[39] PRANK,[40] and CLUSTALW.[9] Table 4 shows the average TC scores of these algorithms on six subsets of BAliBASE 3.0. The data used in Table 8 are drawn from Ref. 24, except the data about IBBOMSA. The proposed IBBOMSA is the fourth best aligner in terms of accuracy. The top aligners are MSAProbs, which reach the highest SP and TC scores on almost all

Table 6. Alignment score comparison between MOMSA and IBBOMSA on the BAliBASE version 2.0.

ALGORITHMS	MOMSA-W (SP)	MOMSA-W (TC)	IBBOMSA (SP)	IBBOMSA (TC)
Ref1 (82)	0.844	0.771	**0.892**	**0.774**
Ref2 (23)	0.925	0.557	**0.947**	**0.637**
Ref3 (12)	0.766	**0.488**	**0.802**	0.442
Ref4 (12)	0.871	0.617	**0.876**	**0.653**
Ref5 (12)	0.936	0.802	**0.948**	**0.812**
Total (141) (mean & SD)	0.861 ± 0.181	**0.893** ± 0.079	**0.702** ± 0.305	0.663 ± 0.290

Table 7. Alignment score comparison between MOMSA and IBBOMSA on the BAliBASE version 3.0

ALGORITHMS	MOMSA-W (SP)	MOMSA-W (TC)	IBBOMSA (SP)	IBBOMSA (TC)
BB11 (38)	0.496	0.379	**0.543**	**0.396**
BB12 (44)	0.848	0.814	**0.869**	**0.879**
BB2 (41)	0.784	**0.362**	**0.798**	0.342
BB3 (30)	0.694	0.371	**0.793**	**0.396**
BB4 (49)	**0.763**	**0.534**	0.742	0.523
BB5 (16)	0.683	**0.418**	**0.692**	**0.498**
Total (218) (mean & SD)	0.722 ± 0.183	0.500 ± 0.309	**0.739** ± 0.2925	**0.505** ± 0.436

Table 8. Average TC score of several algorithms on BAliBASE version 3.0.

ALIGNMENT ALGORITHMS	AVERAGE SCORE (218)	BB11 (38)	BB12 (44)	BB2 (41)	BB3 (30)	BB4 (49)	BB5 (16)	TOTAL TIME(S)
MSAProbs	**0.607**	0.441	0.865	**0.464**	**0.607**	0.622	**0.608**	12382
Probalign	0.589	**0.453**	0.862	0.439	0.566	0.603	0.549	10095.2
MAFFT (auto)	0.588	0.439	0.831	0.45	0.581	0.605	0.591	1475.4
IBBOMSA	0.571	0.411	**0.874**	0.418	0.592	0.635	0.498	2472.6
Procons	0.558	0.417	0.855	0.406	0.544	0.532	0.573	13086.3
Clustal omeg	0.554	0.358	0.789	0.45	0.575	0.579	0.533	539.91
T-Coffee	0.551	0.41	0.848	0.402	0.491	0.545	0.587	81041.5
Kalign	0.501	0.365	0.79	0.36	0.476	0.504	0.435	**21.88**
MOMSA-W	0.500	0.379	0.814	0.362	0.371	0.534	0.418	110289
MUSCLE	0.475	0.318	0.804	0.35	0.409	0.45	0.46	789.57
MAFFT (default)	0.458	0.318	0.749	0.316	0.425	0.48	0.496	68.24
FSA	0.419	0.258	0.818	0.187	0.259	0.474	0.398	53648.1
Dialign	0.415	0.27	0.696	0.292	0.312	0.441	0.425	3977.44
PRANK	0.376	0.265	0.68	0.257	0.321	0.36	0.356	128355
CLUSTALW	0.374	0.223	0.712	0.22	0.272	0.396	0.308	766.47

the subsets of BAliBASE version 3.0. The fastest method is Kalign2, and the slowest one is PRANK. IBBOMSA is the seventh best aligner in terms of time. It proves that the effort in improving the accuracy and running time for the proposed IBBOMSA method is successful.

Conclusions

In this paper, we have proposed an improved BBO algorithm for solving MSA. We design a new migration operator to maintain exploration and exploitation. However, we have to use scaling function carefully. We compared the new algorithm with the existing BBO algorithm. It shows that the new algorithm is superior to the existing BBO or at least competitive. To test our present approach, we considered a good number of benchmark datasets from Bali base 2.0, so as to cover all the test sets of MOMSA. Therefore, the corresponding Bali score of this solution was used to compare with other methods, as they used Bali score as their measure of the quality/accuracy of the

MSA. The experimental results proved that the proposed BBO performed better for most of the test cases. Since the solution of the proposed method was not best for some test cases, but it is close to the best. The proposed method performed better than the others because of its improved migration operator to help maintain diversity of search space. After the experimental analysis, we can say that the proposed method can effectively solve an MSA problem.

Author Contributions

Conceived and designed the experiments: RKY. Analyzed the data: RKY. Wrote the first draft of the manuscript: RKY. Contributed to the writing of the manuscript: RKY, HB. Agree with manuscript results and conclusions: RKY, HB. Jointly developed the structure and arguments for the paper: RKY, HB. Made critical revisions and approved final version: RKY, HB. Both authors reviewed and approved of the final manuscript.

REFERENCES

1. Gusfield D. *Algorithms on Strings, Trees and Sequences Computer Science*. Cambridge: Cambridge University Press; 1997.
2. Feng D, Johnson M, Doolittle R. Aligning amino acid sequences: comparison of commonly used methods. *J Mol Evol*. 1985;21:112–25.
3. Bonizzoni P, Della Vedova G. The complexity of multiple sequence alignment with SP-score that is a metric. *Theor Comp Sci*. 2001;259:63–79.
4. Carrillo H, Lipman D. The multiple sequence alignment problem in biology. *SIAM J Appl Math*. 1988;48:1073–82.
5. Dayhoff MO, Schwartz RM. A model of evolutionary change in proteins. In: Dayhoff MO, ed. *Atlas of Protein Sequence and Structure*. Washington, DC: National Association for Biomedical Research; 1978:345–52.
6. Needleman SB, Wunsch CD. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*. 1970;48:443–53.
7. Taylor WR. A flexible method to align large numbers of biological sequences. *J Mol Evol*. 1988;28:161–9.
8. Feng DF, Doolittle RF. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J Mol Evol*. 1987;25:351–60.
9. Thompson JD, Higgins DG, Gibson TJ. 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.
10. Kim J, Pramanik S, Chung MJ. Multiple sequence alignment using simulated annealing. *Comput Appl Biosci*. 1994;10:419–26.
11. Lukashin AV, Engelbrecht J, Brunak S. Multiple alignment using simulated annealing: branch point definition in human mRNA splicing. *Nucleic Acids Res*. 1992;20:2511–16.
12. Eddy SR. Multiple alignment using hidden Markov models. *Proc Int Conf Intell Syst Mol Biol*. 1995;3:114–20.
13. Yadav RK, Banka H. Genetic Algorithm with Improved Mutation Operator for Multiple Sequence Alignment. *Springer AISC*. 2015;340:515–524.
14. Yadav RK, Banka H. Genetic Algorithm using Guide Tree in Mutation Operator for solving Multiple Sequence Alignment. *Springer AISC*. 2016;395:145–157.
15. Cai L, Juedes D, Liakhovitch E. Evolutionary computation techniques for multiple sequence alignment. IEEE In: *Evolutionary Computation, Proceedings of the* Congress on 2000;2:829–835.
16. Chellapilla K, Fogel GB. Multiple sequence alignment using evolutionary programming. IEEE In: *Evolutionary Computation, CEC. Proceedings of the* Congress on 1999;1.
17. Horng JT, Lin CM, Liu BJ, Kao CY. Using genetic algorithms to solve multiple sequence alignments. In *Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation,* Morgan Kaufmann Publishers Inc. July 10, 2000;883–890.
18. Ishikawa M, Toya T, Totoki Y, Konagaya A. Parallel iterative aligner with genetic algorithm. *Genome Inform*. 1993;4:84–93.
19. Notredame C, Higgins DG. Saga: sequence alignment by genetic algorithm. *Nucleic Acids Res*. 1996;24:1515–24.
20. Lee ZJ, Su SF, Chuang CC, Liu KH. Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. *Appl Soft Comput*. 2008;8:55–78.
21. Shyu C, Sheneman L, Foster JA. Multiple sequence alignment with evolutionary computation. *Genet Program Evol Mach*. 2004;5:121–44.
22. Gondro C, Kinghorn B. A simple genetic algorithm for multiple sequence alignment. *Genet Mol Res*. 2007;6:964–82.
23. Taheri J, Zomaya AY. RBT-GA: a novel metaheuristic for solving the multiple sequence alignment problem. *BMC Genomics*. 2009;10:1–11.
24. Naznin F, Sarker R, Essam D. Progressive alignment method using genetic algorithm for multiple sequence alignment. *IEEE Trans Evol Comput*. 2012;16:615–31.
25. Naznin F, Sarker R, Essam D. Vertical decomposition with genetic algorithm for multiple sequence alignment. *BMC Bioinformatics*. 2011;12:353.
26. Zhu H, He Z, Jia Y. A novel approach to multiple sequence alignment using multi-objective evolutionary algorithm based on decomposition. *IEEE J Biomed Health Inform*. 2015:20(2);1–11.
27. Taheri J, Zomaya AY, Zhou BB. *RBT-L: A Location Based Approach for Solving the Multiple Sequence Alignment Problem*. Darlington, NSW: School of Information Technologies, University of Sydney; 2008.
28. Taheri J, Zomaya AY. RBT-L: a location based approach for solving the multiple sequence alignment problem. *Int J Bioinform Res Appl*. 2010;6:37–57.
29. Simon D. Biogeography-based optimization. *IEEE Trans Evol Comput*. 2008;12:702–13.
30. Thompson JD, Plewniak F, Poch O. Balibase: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics*. 1999;15:87–8.
31. Bahr A, Thompson JD, Thierry JC, Poch O. Balibase (benchmark alignment database): enhancements for repeats, transmembrane sequences and circular permutations. *Nucleic Acids Res*. 2001;29:323–6.
32. Liu Y, Schmidt B, Maskell DL. MSAProbs: multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities. *Bioinformatics*. 2010;26:1958–64.
33. Roshan U, Livesay DR. Probalign: multiple sequence alignment using partition function posterior probabilities. *Bioinformatics*. 2006;22:2715–21.
34. Katoh K, Standley DM. MAFFT: multiple sequence alignment software version 7: improvements in performance and usability. *Mol Biol Evol*. 2013;30:772–80.
35. Do CB, Mahabhashyam MS, Brudno M, Batzoglou S. ProbCons: probabilistic consistency-based multiple sequence alignment. *Genome Res*. 2005;15:330–40.
36. Sievers F, Wilm A, Dineen D, et al. Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Mol Syst Biol*. 2011;7:539.
37. Notredame C, Higgins DG, Heringa J. T-Coffee: a novel method for fast and accurate multiple sequence alignment. *J Mol Biol*. 2000;302:205–17.
38. Lassmann T, Frings O, Sonnhammer ELL. Kalign2: high-performance multiple alignment of protein and nucleotide sequences allowing external features. *Nucleic Acids Res*. 2009;37:858–65.
39. Edgar RC. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res*. 2004;32:1792–7.
40. Bradley RK, Roberts A, Smoot M, et al. Fast statistical alignment. *PLoS Comput Biol*. 2009;5:e1000392.
41. Morgenstern B, Frech K, Dress A, Werner T. DIALIGN: finding local similarities by multiple sequence alignment. *Bioinformatics*. 1998;14:290–4.
42. Loytynoja A, Goldman N. webPRANK: a phylogeny-aware multiple sequence aligner with interactive alignment browser. *BMC Bioinformatics*. 2010;11:579–84.