Soft Computing (2020) 24:15871–15888
https://doi.org/10.1007/s00500-020-04917-5

METHODOLOGIES AND APPLICATION

A bi-objective function optimization approach for multiple sequence alignment using genetic algorithm

Biswanath Chowdhury[1] · Gautam Garai[2]

Published online: 12 April 2020

Abstract

Multiple sequence alignment (MSA) is characterized as a very high computational complex problem. Therefore, MSA problem cannot be solved by exhaustive methods. Nowadays, MSA is being solved by optimizing more than one objective simultaneously. In this paper, we propose a new genetic algorithm based alignment technique, named bi-objective sequence alignment using genetic algorithm (BSAGA). The novelty of this approach is its selection process. One part of the population is selected based on the Sum of Pair, and rest is selected based on Total Conserve Columns. We applied integer-based chromosomal coding to represent only the gap positions in an alignment. Such representation improves the search technique to reach an optimum even for longer sequences. We tested and compared the alignment score of BSAGA with other relevant alignment techniques on BAliBASE and SABmark. The BSAGA shows better performance than others do, which was further proved by the Wilcoxon sign test.

Keywords Multiple sequence alignment · Genetic algorithm · Integer coding · Selection · Wilcoxon sign test · Experimental comparison · Bi-objective function

1 Introduction

Sequence alignment (SA) is one of the most common and fundamental tasks in bioinformatics for analyzing biological macromolecules like DNAs, RNAs, and proteins. The specific residues of a sequence that play some important functional and structural roles remain conserved through natural selection. Therefore, inferring evolutionary history, functional, and/or structural properties based on residue conservation of a sequence are useful to find other related sequences (Thompson et al. 2011). The process of SA arranges two or more sequences in such a way that a maximum number of identical or similar residues are matched or aligned in a column (Mount 2004). Thus, it helps to locate the sites of common portions that share a common evolutionary history. Sometimes, the relative positions of residues within the orthologous sequences are disturbed by some insertion and deletion (indel) of stretches of residues over evolutionary time. This leads to differences in the length of the sequences. An indel event is represented by introducing one or more spaces or gaps inside an alignment. A gap indicates a possible loss or absence of a residue in a sequence with respect to other set of sequences considered in the alignment.

SA can be distinguished between pairwise sequence alignment (PSA) and multiple sequence alignment (MSA) in which the former aligns only a sequence pair (even if they are part of a larger set) while the latter aligns more than two sequences simultaneously. MSA is more advantageous than PSA as it considers multiple sequences of a family at a time and helps to reveal the structural and functional information of a sequence family within a concise format (Chuong and Kazutaka 2008). Alignment can be performed globally (Needleman and Wunsch 1970) or locally (Smith and Waterman 1981). In global alignment,

Communicated by V. Loia.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s00500-020-04917-5) contains supplementary material, which is available to authorized users.

✉ Biswanath Chowdhury
bchowdhury2410@gmail.com

[1] Department of Biophysics, Molecular Biology and Bioinformatics, University of Calcutta, Kolkata, India

[2] Computer Section, Saha Institute of Nuclear Physics, Kolkata, India

the similarity is considered over the entire length of the sequences, whereas local alignment finds the locally highest scoring region(s) of densely similar characters. Local alignment is performed when sequences are only homologous over local regions (Heringa and Taylor 1997).

An alignment is always associated with a scoring function for measuring the quality of the alignment. However, selecting an optimum scoring function is itself a very complicated task because the mathematically optimized alignment function is not the biologically optimum for MSA (Notredame 2002). The computational complexity is also very high and demands computer resources (Wang and Jiang 1994). Finding an optimum alignment for PSA is possible using dynamic programming (DP) methods (Sean 2002; Shyu et al. 2004). However, DP methods suffer from high-dimensionality issues in MSA as time requirement grows exponentially with increasing numbers of sequences (Lipman et al. 1989). The computation of an exact MSA is NP complete (Wang and Jiang 1994), and therefore, all practical MSA methods use heuristic approaches that are approximate in nature.

Two types of heuristic, namely, progressive, and iterative approaches are frequently used in MSA. In progressive alignment (Hogeweg and Hesper 1984), an MSA problem is solved indirectly by PSA. It first performs the alignment of all the possible sequence pairs and develops a guide tree (Feng and Doolittle 1987) based on the pairwise distance values. Finally, it generates an MSA in a stepwise fashion by gradually assembling all sequences progressively following the guide tree where the best alignment pair is first taken into account. The most widely used progressive method is ClustalW (Thompson et al. 1994). The main drawback of this method is that the final MSA is influenced by the progressive nature of alignment of initial sequence pairs. Therefore, altering the positions of gaps in later stages is not possible (Thompson et al. 2005). This problem is solved by iteratively modifying the alignment made by progressive methods. This process is adopted in the iterative approach. It iteratively modifies the construction of guide tree by modifying the alignment pairs in progressive alignment. Some examples of iterative methods are MAFFT (Katoh et al. 2002, 2005), MUSCLE (Edgar 2004), PRIME (Yamada et al. 2006), PRRP (Gotoh 1996), and MUMMALS/PROMALS (Pei and Grishin 2006, 2007).

Genetic algorithm (GA) (Goldberg 1989; Holland 1975) is another kind of iterative method, inspired by natural genetics. GA is stochastic in nature that follows the "survival of the fittest" mechanism. It considers a set of probable solutions of the problem under study. Each solution is represented as a *chromosome,* and the set of chromosome is called a *population*. To determine the solution quality, each chromosome is associated with a fitness function (objective function). Based on the fitness values, the *selection* operation selects the chromosomes that are fit for the generation. After that, the genetic operators like *crossover* and *mutation* are applied on the selected chromosomes to change their fitness and yield next possible better generation. The processes of selection, crossover, and mutation are iteratively continued until the solution (chromosome) is converged to the best fitness value. Many researchers have used GA to solve the sequence alignment problem. Among them, SAGA is a popular software developed by Notredame and Higgins (1996). It tries to solve the MSA problem with 22 different types of complex GA operators. Naznin et al. (2011), in one approach, proposed a GA-based method that solved the MSA problem by first dividing the sequences vertically into two or more subsequences. In their second approach (Naznin et al. 2012), the initial population was made up of randomly generated guide trees by using the distance table. The GA is used to iteratively modify the guide trees for identifying the best guide tree. MSA-GA (Gondro and Kinghorn 2007) is another GA-based method where the initial population is created using the prealignment results of Needleman–Wunsch algorithm. Narimani et al. (2012) proposed a GA-based method where one part of the population is generated randomly and the rest part of it by the clustalW. In several other approaches, GA was combined with another optimization technique like rubber band technique (RBT) (Taheri and Zomaya 2009) or ant colony optimization (ACO) (Lee et al. 2008) to optimize the sequence alignment solution. Other than the GA, another population-based meta-heuristic approach, called chemical reaction optimization (CRO) (Lam and Li 2010), was applied to change the positional values of residues and gaps in an MSA (Wadud et al. 2018).

Nowadays, MSA is not considered to be solved with a single-objective function due to the drawbacks of selecting or formulating the optimum scoring function. MSA is now treated as a multi-objective optimization problem where each criterion is represented by a separate objective function. However, there is always a possibility in the multiple objectives that the improvement in one objective function may deteriorate the solution of one or more objective functions. Therefore, a set of non-dominated solutions are selected (Zhou et al. 2011) called Pareto optimal solutions. Besides this non-dominated set, no other solution exists which can improve any one of the objective functions without the degradation of another objective (Ehrgott 2005). A few MSA works based on the multi-objective functions were proposed where the GA acted as an optimizer. One of them was MO-SAStrE (Ortuño et al. 2013), which considered three objective functions: STRIKE score (Kemena et al. 2011), non-gaps percentage, and Total Conserved Columns (TCC). The STRIKE score helps to make the alignment accurate by using at least one single-known 3D structure, retrieved from the PDB (Berman et al. 2000). However, the main drawback of considering this objective function is the limited availability of structures. Other than that, MO-SAStrE used non-dominated sorting genetic algorithm (NSGA-II) (Deb et al. 2002) that made it computationally expensive. In another approach, called MSAGMOGA (Kaya et al. 2014), the non-dominated Pareto alignment solutions were obtained by considering three objectives: affine gap penalty minimization, similarity maximization, and support maximization. Support maximization includes a number of sequences in an alignment that increases the alignment quality. Apart from these, some other evolutionary approaches were also developed for multi-objective function optimization. In HMOABC (Rubio-Largo et al. 2016a) and H4MSA (Rubio-Largo et al. 2016b), an artificial bee colony (Karaboga 2005) and a shuffled frog-leaping optimization algorithm (Eusuff et al. 2006) were used, respectively. Both the methods considered two widely used objective functions, namely Sum of Pair (SOP) and TCC to produce a Pareto optimal set. HMOABC and H4MSA used another fast and accurate Kalign (Lassmann et al. 2009) as a local search procedure to improve the quality of solutions.

However, to make the set of non-dominated solutions in multi-objective Pareto optimal approach, one needs to determine which of the solutions are dominated and which of them are not. It is difficult to perform from a computational point of view. A simple approach to finding a non-dominated solution from a set of solutions requires $O(kn^2)$ time having total n number of solutions with k number of objective functions (DeRonne and Karypis 2013).

In this paper, we propose a GA-based approach named as bi-objective sequence alignment using genetic algorithm (BSAGA). Here, we try to optimize two most widely used objective functions, namely SOP and TCC. The proposed method is different from other multi-objective optimization techniques since it does not consider any non-dominated set of solutions and thus reduces the computational complexity. It divides the entire population of solutions into two parts. The selection from one part of the population is then performed based on SOP objective function, while another part is based on TCC objection function (discussed in Sect. 2.3.1; Fig. 2). Such selection process along with proposed crossover and mutation operations helps to obtain an optimum MSA solution for BSAGA. The BSAGA is based on an integer coding method where a set of integers represents a chromosome. Each integer value of the chromosome represents a possible gap position inside an MSA. Such integer coding technique reduces the length of the chromosome and also the computational complexity.

2 Method

The proposed method (BSAGA) finds the optimum MSA by optimizing two important criteria: SOP and TCC, without considering any non-dominated solution sets. The GA-based BSAGA method iteratively modifies the gap positions in an alignment.

2.1 Population initialization and alignment representation

For an MSA problem, a given set of sequences S_i, $\forall\ i \in \{1, 2,\ldots, k\}$ having variable lengths L_j, $\forall\ j \in \{1, 2, \ldots, m\}$, generally forms a matrix with k number of rows in which each row of the matrix represents a sequence and the number of columns defines the alignment length. During alignment, each sequence is made same in size by inserting a variable number of gaps. The *alignment length* of a sequence is the sum of the original sequence length, L_j, and the number of gaps introduced. Therefore, either a residue (nucleotide or amino acid) or a gap (–) occupies a position of each row.

The conventional GA generally represents a solution or a chromosome by a binary string. However, the binary coding in MSA increases the chromosome/string length, the computational complexity, and the memory space. In the proposed method, we have used an integer coding instead of the binary coding for chromosome representation. The integer values in a chromosome signify different possible positions of gaps in an alignment. Therefore, the quality of an MSA is improved by optimizing the number of gaps and their possible locations in the alignment.

In initialization, an initial population of size N is randomly generated. Each individual or chromosome P_i, $\forall\ i \in \{1, 2,\ldots, N\}$ in a population, is made up of integer values. Each randomly generated integer value within the lower and the upper limits of a chromosome defines a gap position in the alignment. A gap may be present at any place of an alignment. The lower and upper bounds define the lowest and highest possible positions of a gap. The lower bound is always one for every sequence in the alignment, and the upper bound is computed considering the following equation.

$$\mathrm{AL} = s * \max(l_1, l_2, \ldots, l_k) \tag{1}$$

where AL is the alignment length, s is a scaling factor used to allow the alignment length be 20% longer than the longest sequence in the alignment set, and l_i's, $\forall\ i \in (1,2,\ldots,k)$, are the sequence lengths. The solutions of common alignment problems rarely contain more than 20% gaps than the longest sequence (Gondro and Kinghorn 2007). Figure 1 is an example of the proposed chromosome codification and its corresponding decoded form. Here, the

Fig. 1 Chromosomal representation of an alignment structure (numbers within parentheses represent lengths); **a** four unaligned amino acid sequences; **b** gap positions represented by a chromosome of BSAGA; and **c** the decoded alignment is formed by placing gaps in the corresponding sequences

```
s1: GARFIELDTHELASTFATCAT    (21)

s2: GARFIELDTHEFASTCAT     (18)

s3: GARFIELDTHEVERYFASTCAT (22)

s4: THEFATCAT               (9)
```

(a)

```
20 22 18 26 21                    (5)

20 13 22 15 12 21 14 26           (8)

26 20 22 21                       (4)

13 21 3 14 20 7 2 12 6 1 4 15 26 5 22 8 18   (17)
```

(b)

```
Alignment length = 22 (largest sequence) * 1.2 = 26

GARFIELDTHELASTFA-T---CAT-   (26)

GARFIELDTHE----FAST---CAT-   (26)

GARFIELDTHEVERYFAST---CAT-   (26)

--------THE----FA-T---CAT-   (26)
```

(c)

upper limit of the gap position is set to 26 according to the Eq. 1.

2.2 Fitness function

The fitness score of a chromosome determines the quality of a solution. In BSAGA, each chromosome is associated with two fitness functions. One of them is F_{SOP} based on SOP score which indicates the quality of an alignment, while the other one is F_{TCC} based on TCC score that determines the sequence conservation. The quality of an alignment and sequence conservation are two important criteria for any sequence alignment, and therefore, we have chosen these two objectives. The benchmark databases like BAliBASE and SABmark have also used these two criteria: SOP and TCC to evaluate the performance of an alignment method. To optimize the SOP score, one needs to maximize F_{SOP} according to the following equation.

$$F_{\mathrm{SOP}} = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \delta\left(S_i, S_j\right) \tag{2}$$

where $\delta(S_i,\ S_j)$ represents the pairwise alignment score between two aligned sequences S_i and S_j and k is the total number of sequences. For similarity score, we used BLOSUM 62 matrix (Henikoff and Henikoff 1992). For the affine gap penalty score, we have evaluated the following equation.

$$W = \alpha + \beta \times (g - 1) \tag{3}$$

where W is the total gap penalty, α is the gap opening penalty, β is the gap extension penalty, and g is the total number of gaps. F_{TCC} takes into account the number of columns that are aligned completely with identical or similar amino acid residues and is defined by the following equation.

$$F_{\mathrm{TCC}} = \sum_{i=1}^{m} C_i \quad (4)$$

where C_i is the similar or identical column and m is the total similar or identical columns.

2.3 Operators

BSAGA includes three standard genetic operators, namely selection, crossover, and mutation. The crossover and mutation operations are performed to the selected chromosomes from the population according to the probabilities P_{c} and P_{m}, respectively.

2.3.1 Selection operator

This operator chooses two best individuals (say, P_a and P_b) randomly as parents from the population pool of P_i, $\forall\ i \in \{1, 2,\ldots, N\}$, to take part in the crossover and mutation operations to produce offsprings P'_a and P'_b for the next generation. The chromosome with the best fitness score is more likely to be selected. In the proposed method, each chromosome is associated with two fitness scores, F_{SOP} and F_{TCC} (given in Eqs. 2 and 4). However, selecting one chromosome having best fitness scores for both the functions is rarely possible. Therefore, in BSAGA, the entire population of size N is divided into two parts. Selection based on F_{SOP} fills one part of the population, while the selection based on F_{TCC} fills the rest of the population. Thus, the selection process continues along with crossover and mutation until an entire new population P_j, $\forall\ j \in \{1, 2,\ldots, N\}$, is created for the next generation. Figure 2 illustrates the selection operation. We have considered the tournament selection method with tournament size 2. In this method, two randomly chosen chromosomes are entered into a tournament for a competition. The fittest among the two (determined by either F_{SOP} or F_{TCC}) is selected as a parent and take part in the crossover. Tournament selection can adjust the different selection pressures without any structural changes. The others advantages of the tournament selection method than other selection processes are described in Rahman et al. (2016) and Miller and Golberg (1995).

2.3.2 Crossover operator

During the crossover, the randomly selected two parents, P_a and P_b, exchange their genetic information between each other to produce two new offsprings, P'_a and P'_b, for the next generation. In BSAGA, we have proposed a modified vertical crossover operation named as *Position Guided Crossover* (*PGC*). The occurrence of this operation is determined by the probability, P_{c}. In the proposed method, the crossover helps to exchange the gap positions between two chromosomes. At first, a position, p_g, is randomly chosen from P_a. The gap value, say v, of p_g acts as a checkpoint for the vertical division. The division is performed in such a way so that the first blocks after vertical division of both P_a and P_b contain the gap values which are less or equal to the value of v and the second block contains gap values higher than v. Figure 3 illustrates the crossover operation. In this example, a random position, p_g from P_a is chosen, which is, position 3 (as shown in Fig. 3a). The gap value, v, in position 3 is 14. Therefore, the first blocks of P_a and P_b contain the gap values which are less or equal to 14, and the second blocks of P_a and P_b contain the gap values higher than 14 as shown in Fig. 3b. Next, the first block of P_a and second block of P_b are combined to produce the offspring, P'_a, and the first block of P_b and second block of P_a are combined to create another offspring, P'_b (shown in Fig. 3c). Due to the guided vertical division, the blocks of one parent may contain an unequal number of gaps compared to the other parent. Hence, to maintain the number of

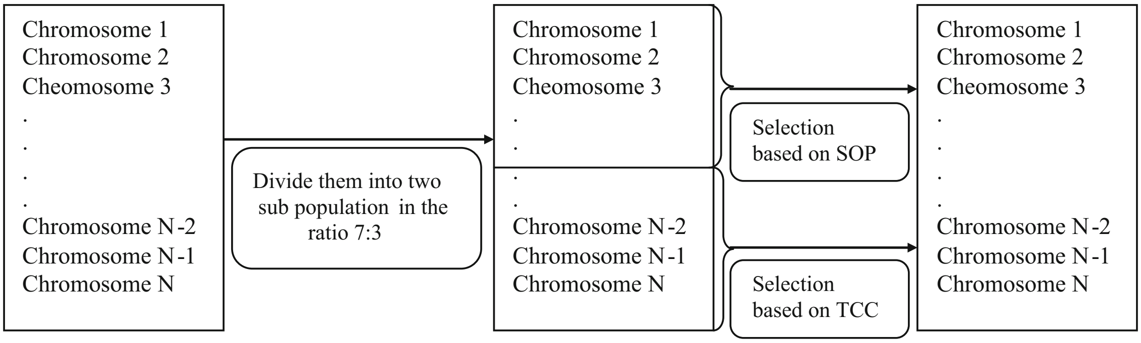


Fig. 2 Schematic representation of the selection operation based on SOP and TCC

P_a	P_b
13 5 22 15 10	19 2 4 16 20
11 23 $\boxed{14}$ 7 20 26 9 17	10 21 15 4 12 6 16 26
2 21 12 24	8 20 18 14
10 20 3 14 5 26 7 9 16 11 24 6 19 2	1 14 22 25 16 8 17 10 21 14 16 2 19 12

(a)

Block 1 (P_a)	Block 2 (P_a)	Block 1 (P_b)	Block 2 (P_b)
13 5 10	22 15	2 4	19 16 20
11 14 7 9	23 20 26 17	10 4 12 6	21 15 16 26
1 12	21 24	8 14	20 18
10 3 14 5 7 9 11 6 2	20 26 16 24 19	1 14 8 10 14 2 12	22 25 16 17 21 16 19

(b)

P'_a	P'_b
13 5 10 19 16	2 4 22 15 26
11 14 7 9 21 15 16 26	10 4 12 6 23 20 26 17
1 12 20 18	8 14 21 24
10 3 14 5 7 9 11 6 2 22 25 16 17 21	1 14 8 10 14 2 12 20 26 16 24 19 26 26

(c)

Fig. 3 Example of the PGC crossover operation; **a** out of two randomly selected parents (P_a and P_b), position 3 of sequence 2 of P_a is randomly selected for crossover operation; **b** blocks 1 and 2 are constructed based on the value of position 3; **c** two offsprings, P'_a and P'_b, are produced by interchanging the blocks of two parents

gaps in the offsprings, we either have discarded extra gaps or padded with trailing gap positions at the end of a chromosome to compensate the length variability during blocks exchange. In Fig. 3c, the number of gaps for sequence 1 was five before crossover and so we exclude gap position 20 from block 2 of P_b in producing P'_a. On the other hand, two trailing gap positions (position number 26) are included in sequence 4 of block 2 of P_a to produce P'_b.

2.3.3 Mutation

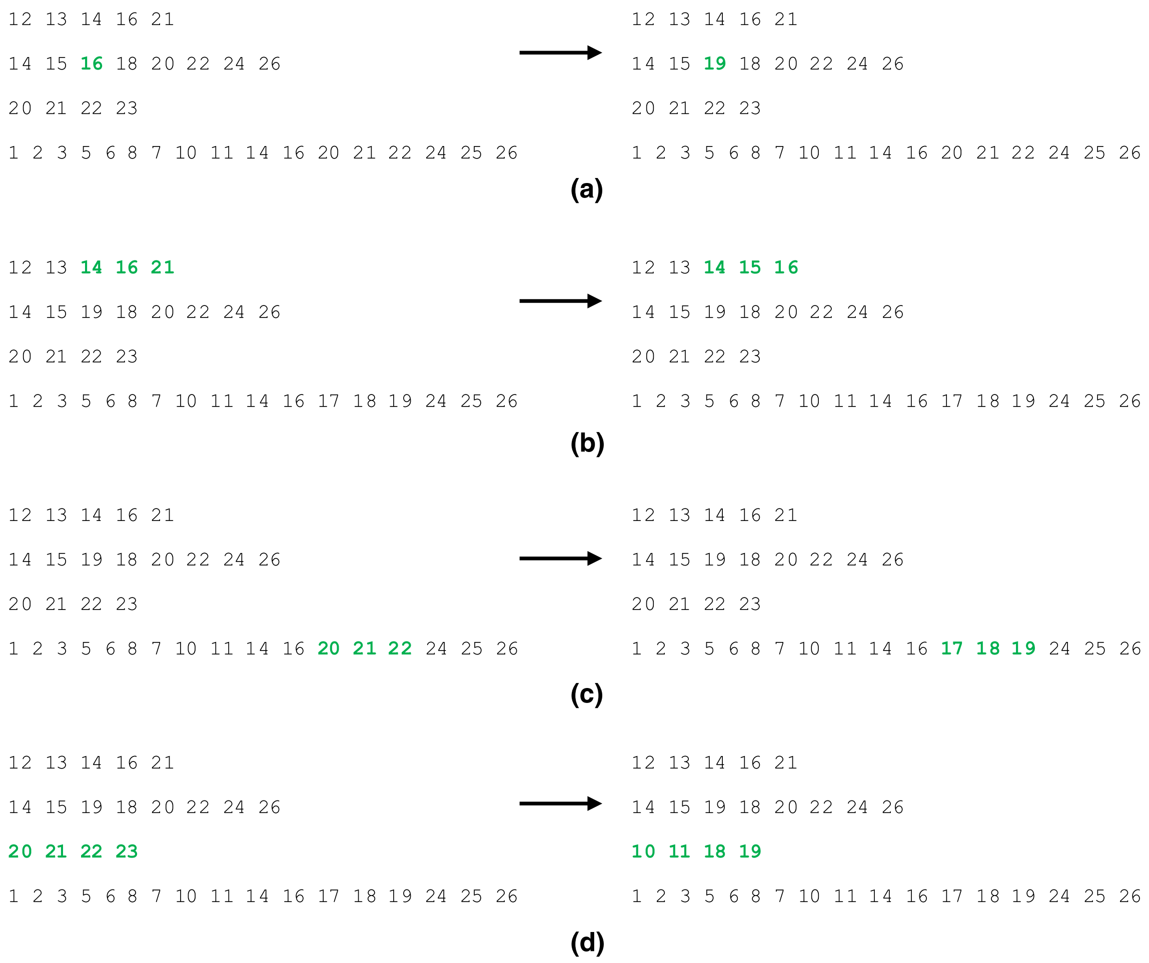
Mutation operation helps to maintain the diversity in the population. In BSAGA, we have considered four types of mutation operations, namely *Single-gap mutation*, *Merge-gap mutation*, *Block-gap mutation*, and *Split-gap mutation*. After crossover, an offspring follows the above mutation operations in order. However, the occurrence of each of the operations is determined by the mutation probability, P_{m}. A mutation operation may either change a position of a single gap or a block of gaps to another position in an alignment. For that, we have used *Single-gap* and *Block-gap mutations*, respectively. Also biologically more meaningful alignment contains a stretch of gaps in one place more often than discretely distributed gaps. Based on the criterion, we have introduced *Merge-gap mutation*. Sometimes, due to *Merge-gap mutation* there is a possibility to have a very long stretch of gaps in only one particular place in an alignment. To reduce it, we have used *Split-gap mutation* to divide the long length of gaps into two. These four mutation operations are not performed in any specific gap of any specific sequence in an alignment. The operations have been used randomly to make the mutation process free from any biasness. Figure 4 illustrates all four mutation operations. For better understanding, we have shown the gap values in a chromosome in ascending order in the figure.

2.3.3.1 Single-gap mutation In this operation, a single row of a chromosome (say, row 2 from top) that represents the gap positions of a sequence is randomly selected. From that row, one of the positions is randomly selected. Finally, the value of the selected position is changed to another random value to allow a different the gap position in the sequence. The operation is illustrated in Fig. 4a. In this figure, the gap position (marked as green) is selected randomly and changed to another random gap position that lies within upper limit of gap position as described in Sect. 2.1.

2.3.3.2 Merge-gap mutation In this method, a single row is selected randomly (say top most row). After that, it randomly chooses more than one positions at a stretch that represent discrete gap values. Finally, it changes them to continuous gap values so that the gaps are merged in a sequence. Figure 4b represents this operation using the gap positions highlighted in green.

2.3.3.3 Block-gap mutation In this mutation, a single row is randomly (say, lower most row) selected. It then randomly chooses one block of gap positions that contains a continuous gap values. Finally, it shifts the entire block of gap values to a different random position. The selected gap block must have more than one continuous gap positions. This operation with the green-colored gap values is shown in Fig. 4c.

2.3.3.4 Split-gap mutation This operation selects a row randomly (say, third row from top) and then selects a block of continuous gap positions at random. It then divides the block into two blocks of approximately same size. After dividing, the values of two sub-blocks are changed



12 13 14 16 21 → 12 13 14 16 21

14 15 **16** 18 20 22 24 26 → 14 15 **19** 18 20 22 24 26

20 21 22 23 → 20 21 22 23

1 2 3 5 6 8 7 10 11 14 16 20 21 22 24 25 26 → 1 2 3 5 6 8 7 10 11 14 16 20 21 22 24 25 26

(a)

12 13 **14 16 21** → 12 13 **14 15 16**

14 15 19 18 20 22 24 26 → 14 15 19 18 20 22 24 26

20 21 22 23 → 20 21 22 23

1 2 3 5 6 8 7 10 11 14 16 17 18 19 24 25 26 → 1 2 3 5 6 8 7 10 11 14 16 17 18 19 24 25 26

(b)

12 13 14 16 21 → 12 13 14 16 21

14 15 19 18 20 22 24 26 → 14 15 19 18 20 22 24 26

20 21 22 23 → 20 21 22 23

1 2 3 5 6 8 7 10 11 14 16 **20 21 22** 24 25 26 → 1 2 3 5 6 8 7 10 11 14 16 **17 18 19** 24 25 26

(c)

12 13 14 16 21 → 12 13 14 16 21

14 15 19 18 20 22 24 26 → 14 15 19 18 20 22 24 26

20 21 22 23 → **10 11 18 19**

1 2 3 5 6 8 7 10 11 14 16 17 18 19 24 25 26 → 1 2 3 5 6 8 7 10 11 14 16 17 18 19 24 25 26

(d)

Fig. 4 Different mutation operations performed by BSAGA; **a** single-gap mutation operation; **b** merge-gap mutation operation; **c** block-gap mutation operation; and **d** split-gap mutation operation (color figure online)

randomly in such a manner that two of them contain two different series of continuous gap positions. Figure 4d illustrates this operation with green highlighted gap positions.

2.4 Termination

We have assigned termination criterion, which is based on the convergence to the optimum score of the elite or best so far chromosome. In the proposed method, as the selection considers bi-objective criteria, we thus get two elite chromosomes, one each best with respect to one of the two objective functions. If the fitness scores of both of the best solutions remain unchanged for 100 consecutive generations, we allow terminating the process to reduce computational time and memory requirement.

The proposed BSAGA is represented algorithmically in the following way.

1. Read the sequences for alignment.
2. Initialize the population size N, crossover probability P_c, mutation probability P_m, and generation, $G = 1$.
3. Generate an initial population P_i, $i \in \{1, 2, \ldots, N\}$ of N individuals or chromosomes. Each chromosome represents the gap positions in an alignment, and its length is defined by the number of gaps in an alignment based on the Eq. 1.
4. Evaluate the potential of each individual P_i, $i \in \{1, 2, \ldots, N\}$ based on fitness functions, F_{SOP} and F_{TCC} as described in fitness function.
5. Divide the total population N into two parts.
6. Using tournament selection with tournament size 2, select individuals from one part based on F_{SOP} and from the next part based on F_{TCC}.
7. Perform crossover with probability, P_c, between the selected chromosomes and mutate then with probability, P_m.
8. Each pair of selected parental chromosomes (P_a and P_b) thus creates two new offsprings P'_a and P'_b of next generation.
9. Repeat steps 6–8 until a new pool of individuals P_j, $j \, \varepsilon \, \{1, 2, \ldots, N\}$ is formed and $G = G + 1$.
10. Terminate the process if the termination criterion is satisfied (discussed in Sect. 2.4), otherwise, go to step 4.

3 Results and discussion

To evaluate the performance of BSAGA, we considered a number of test datasets from the benchmark alignment database or BAliBASE (Thompson et al. 1999, 2005), and sequence alignment benchmark or SABmark (Van Walle et al. 2005). The results of BSAGA were compared with other well-known approaches. However, in the comparison, we did not choose those approaches that used the three-dimensional structure of protein to improve the alignment score. This is because the structures for many sequences are still not available, and therefore, for those sequences these approaches fail to align optimally. Finally, we considered Wilcoxon signed-rank test (Corder 2009) to compare the result of proposed approach statistically with others.

3.1 Test datasets

We require some benchmark sequences and the "gold standard" reference alignments of those sequences for assessing the performance of our proposed method quantitatively relative to the gold standard reference alignments.

3.1.1 BAliBASE (http://www.lbgi.fr/balibase/)

It is a popular and most widely applied benchmark database. It contains an application, called BAliscore, which measures the SOP and the TCC scores of the test alignment in comparison with the reference alignments. The scores vary between 0.0 and 1.0. If the test alignment does not match at all with the reference alignment, the score is 0.0. If the test alignment is identical with the reference alignment, then the score is 1.0. If it matches with some parts of the reference alignment, then the score lies between 0.0 and 1.0. We considered BAliBASE version 2.0 and version 3.0 as most of the alignment methods used these versions. Version 3 measures Q-score which is same as SOP score used in version 2.

The version 2 contains eight reference sets. *Reference set 1* contains a number of equidistant sequences with various levels of conservation. The orphan or highly divergent sequences are considered in *Reference 2*. *Reference 3* consists of divergent subfamilies or subgroups where the identity is less than 25% between two subgroups. *Reference 4* contains amino/carboxy (N/C) terminal extensions, and *Reference 5* contains internal insertions and deletions. *References 6* to *8* contain repeats, circular permutations, and transmembrane proteins, respectively.

The version 3 is divided into six different groups or families: *RV11*, *RV12*, *RV20*, *RV30*, *RV40*, and *RV50*. *RV11* contains 38 sets of equidistant sequences and have less than 20% identities. *RV12* is formed by 44 sets of sequences and includes families with at least four equidistant sequences having identities between 20 and 40%. *RV20* consists of 41 sets of sequences and considers families including sequences with more than 40% identities. *RV30* contains 30 sets of sequences from different subfamilies that share less than 25% identity between different subfamilies. *RV40* is formed by 49 sets of sequences having large terminal insertions and share more than 20% identity. *RV50* consists of 16 sets of sequences that contain a large amount of internal insertions and share more than 20% identity.

3.1.2 SABmark (http://bioinformatics.vub.ac.be/databases/databases.html)

It contains sets of multiple alignment problems derived from the structural classification of proteins (SCOP) classification. These sets are divided into two groups: (a) Superfamily (315 sets) that shares at most 50% identity and (b) Twilight (108 sets) that shares between 0 and 25% identity.

3.2 Setup parameters

We have explored different parameter values for testing the performance of BSAGA algorithm to achieve the best score. To determine the optimum values of different parameters, ten BAliBASE (version 2) datasets were randomly selected (five from Reference 2 and five from Reference 3). Tests were performed on them to identify the values of various parameters for achieving the best performance of BSAGA. The initial population was generated randomly for BSAGA. We did not follow any other heuristic approach to produce the initial population like other MSA techniques. This helps the BSAGA independent of other alignment techniques. Also, like HMOABC (Rubio-Largo et al. 2016a) and H4MSA (Rubio-Largo et al. 2016b), we have not used Kalign or any other alignment technique to improve the result.

To generate a population of solutions based on the SOP and the TCC scores (as discussed in Sect. 2.3.1), we have considered eleven different combinations of selection based on SOP–TCC. The combinations were: 0–100, 10–90, 20–80, 30–70, 40–60, 50–50, 60–40, 70–30, 80–20, 90–10, and 100–0. In a combination of 10–90 in a population means, 10 percent of selection was made based on the SOP value and 90% was based on the TCC value. For each of the combinations, the range of P_c was [0.0–1.0] with a step of 0.1. For each value of P_c, the value of P_m was varied in the range [0.1–1.0] with a step of 0.1. Mutation operation plays a significant role in MSA; therefore, we discarded the value 0.0 for P_m. The population size (N) was set to 100 as used by other GA-based alignment methods (Notredame and Higgins 1996; Naznin et al. 2011, 2012). From Fig. 5, we observed that for four combinations of SOP–TCC: 70–30, 80–20, 90–10, and 100–0 (represented by the *x*-axis), BSAGA performed better than other combinations of SOP–TCC. However, all four combinations produced similar scores. For our experiment, we selected the combination of 70–30.

After that, we tried to determine the best combination of P_c and P_m values for the selected SOP–TCC. The performance of BSAGA for different P_c and P_m values was presented by heatmaps in the supplementary figure (Fig. S1). Heatmaps show that, for the default value of SOP–TCC, BSAGA performed better for the value of P_m in the range [0.1–0.5]. The effect of P_c is not very distinguishable. However, the P_c value in the range [0.5–1.0] showed better results for most cases. Therefore, we considered P_c= [0.5–1.0] and P_m= [0.1–0.5] for the BSAGA.

For N, we chose the value 100. However, we run the experiment for different values of N such as 100, 150, and 200 with already selected values of P_c, P_m, and SOP–TCC. The BSAGA with N = 150 and 200 does not show much improvement compared to N = 100 as shown in Fig. 6. Moreover, increasing the value of N leads to increase the computation time.

To set the termination condition of BSAGA after score convergence (discussed in Sect. 2.4), we have considered three values of consecutive iteration numbers: 50, 100, and 150 in the experiment and the results are depicted in Fig. 7. Figure 7 shows that the scores are higher for the numbers 100 and 150 compared to 50. However, the performance of BSAGA was almost similar for 100 and 150 for all ten datasets. Therefore, to reduce the computational time, we have selected 100 as the final value. The final values of the parameters for BSAGA are tabulated in Table 1. After setting up the parameter values, we allowed BSAGA to run ten times with different initial population on each dataset of BAliBASE and SABmark. Best among the ten runs was selected as the final score.

3.3 Result comparison of BSAGA with other approaches for BAliBASE 2.0

To judge the quality of alignment solutions produced by BSAGA, we have compared its results with the results obtained from different well-known and relevant methods. The scores of these methods along with BSAGA for BAliBASE 2.0 are tabulated in Table 2. The values of different methods are collected from the literatures (Naznin et al. 2011, 2012). All the methods reported only Sum of Pair (SPS) score or SOP scores of the BAliscore for comparisons. Therefore, we have considered only the SPS scores for BSAGA. Bold-faced data signify the best score among others for each dataset. A blank cell indicates the unavailability of the result.

From Table 2, it was observed that the GA-based methods performed better than ClustalW. However, it is very difficult to confirm any particular approach which is superior to others due to unequal numbers of dataset

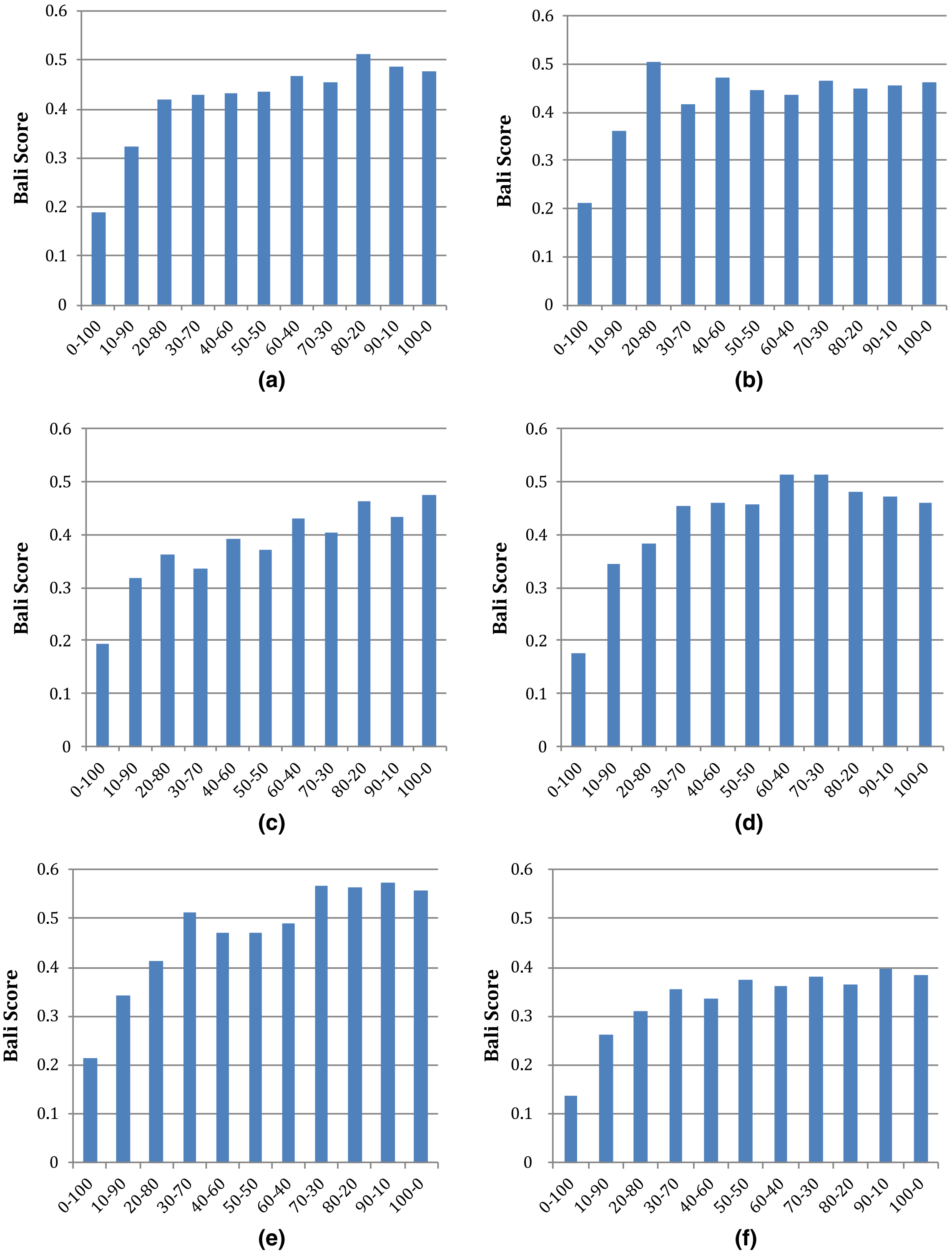


Fig. 5 Performance comparison of BSAGA for different ratios of selection based on SOP and TCC scores. Each figure from **a** to **j** stands for each dataset of ten randomly selected datasets

considered by different methods. Therefore, we have performed pairwise comparison (Sect. 3.3.1) where the BSAGA is compared with another method considering one at a time from the group of methods (listed in Table 2). Along with that, we have also carried out statistical testing. For that, we have chosen Wilcoxon signed-rank test to understand the difference between paired scores.

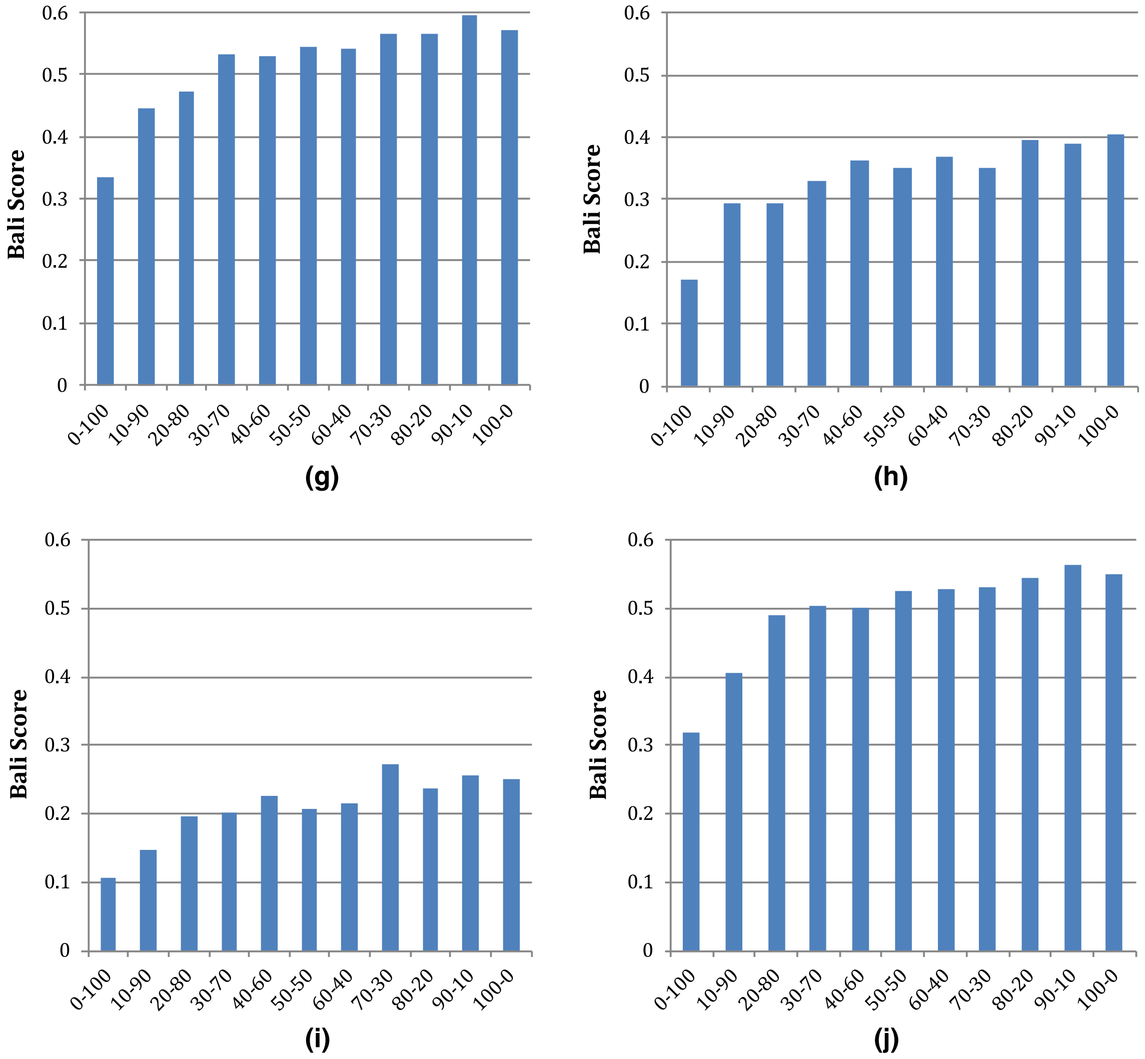


Fig. 5 continued

Wilcoxon signed-rank test is the nonparametric alternative to paired *t* test. It does not require normally distributed data. It uses ranked or ordinal data. Wilcoxon signed-rank test has the null hypothesis that assumes there is no significant difference in the values of two samples. Therefore, the alternative hypothesis is that there is a significant difference in the values of the two samples. Hence, null hypothesis should be rejected to prove the significant difference between two samples. For the statistical testing, we have used the 5% level of significance.

3.3.1 Pairwise comparison of BSAGA with other approaches

To compare the result of BSAGA with others, we have performed pairwise comparison. For that, we have selected only those datasets that were considered by other respective approaches and the comparative results of BSAGA with them along with the statistical results are tabulated in supplementary tables (Tables S1–S7). Bold-faced data signify the best score. The summary of the statistical results is provided in Table 3. For the statistical comparison with VDGA, we have considered VDGA Decomp_3 as the authors of the method found it is better than the other two decomposition values of VDGA (Naznin et al. 2011). From tables S1 to S7, it is observed that BSAGA produced better scores for most of the datasets irrespective of the reference number of the datasets. By considering the statistical *P* values from Table 3, we can say that BSAGA strongly rejected null hypothesis and performed significantly better than MSA-GA, MSA-GA MSA-GA w/prealign, SAGA, RBT-GA, and CLUSTALW. However, comparing with GAPAM, null hypothesis was rejected on a marginal basis. If we compare the performance of BSAGA with VDGA_Decomp_3, then the observed the results were not significantly different, and therefore, the null

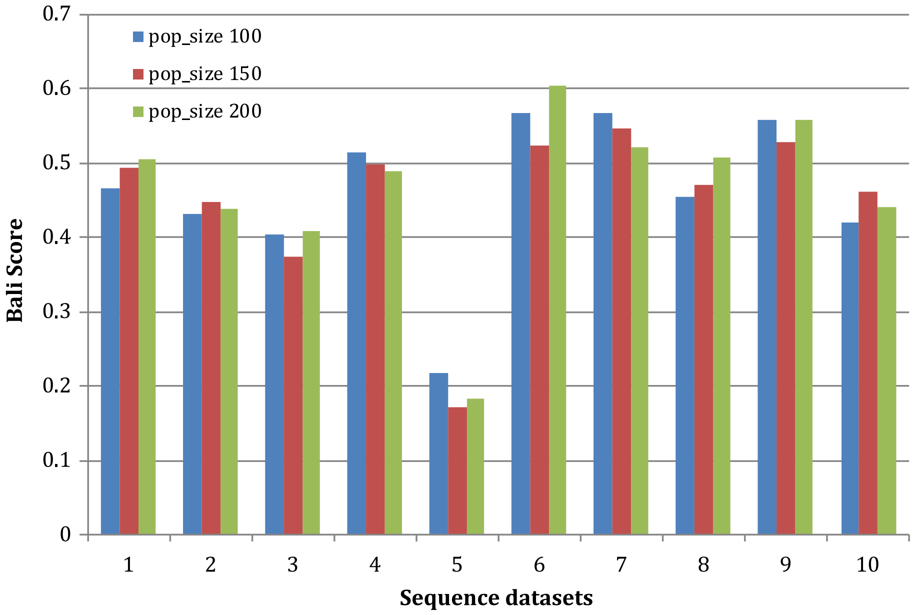


Fig. 6 Performance comparison of BSAGA with different population sizes

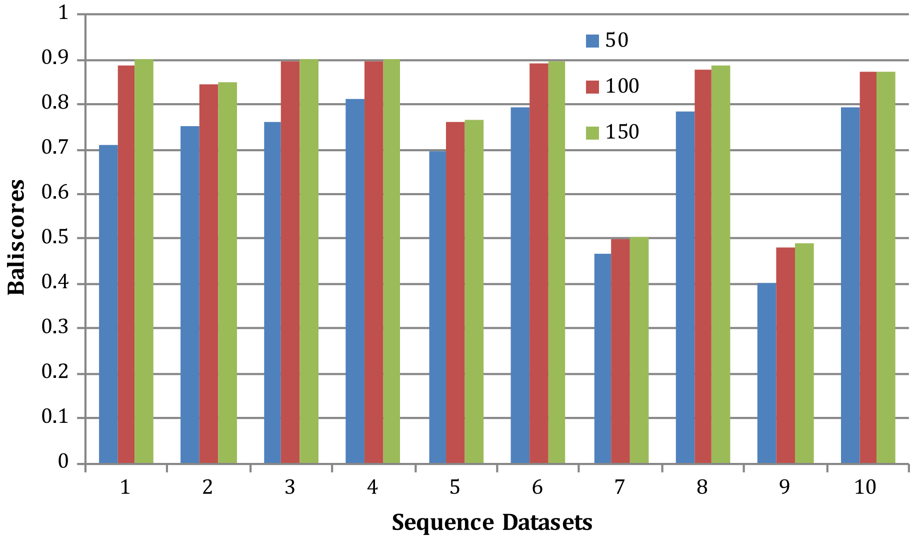


Fig. 7 Performance comparison of BSAGA for different consecutive iteration numbers as termination criterion

Table 1 Parameter values used in BSAGA

Population size (*N*)	100
SOP–TCC selection ratio	70–30
Crossover probability (p_c)	[0.5–1.0]
Mutation probability (p_m)	[0.1–0.5]
Substitution matrix	BLOSUM62
Gap penalty	Gap opening = − 10; gap extension = − 0.6

hypothesis was accepted. However, BSAGA yielded better results for 37 cases and VDGA_Decomp_3 performed better for 22 cases and for 3 datasets, both the methods produced the same results. In addition to that, VDGA applied another heuristic approach to find the MSA solution using guide tree optimization approach, which is computationally complicated, whereas BSAGA optimized an MSA only by changing the gap positions in an alignment without considering any other heuristic approach.

Table 2 Comparative alignment results of different approaches along with BSAGA on BAliBASE 2.0

References	Name of dataset	CLUST-ALW/X	MSA-GA	MSA-GA w/prealign	SAGA	RBT-GA	GAPAM	VDGA_ Decomp_2	VDGA_Decomp_3	VDGA_Decomp_4	BSAGA
Ref. 1	1idy	0.500	0.427	0.438	0.342	–	0.565	0.550	0.651	0.654	**0.700**
	1tvxA	0.042	0.295	0.209	0.278	–	0.316	0.316	0.316	0.310	**0.440**
	1uky	0.392	0.443	0.405	**0.672**	–	0.402	0.416	0.459	0.464	0.410
	Kinase	0.479	0.295	0.488	**0.862**	–	0.487	0.531	0.545	0.548	0.602
	1ped	0.592	0.501	0.687	**0.746**	–	0.498	0.443	0.482	0.451	0.688
	2myr	0.296	0.212	0.302	0.285	–	0.317	0.347	**0.359**	0.282	0.325
	1ycc	0.643	0.650	0.653	0.837	–	0.845	0.752	0.839	0.685	**0.847**
	3cyr	0.767	0.772	0.789	**0.908**	–	0.911	0.797	0.898	0.797	0.807
	1ad2	0.773	0.821	0.845	0.917	–	0.956	0.959	0.950	0.941	**0.970**
	1ldg	0.880	0.895	0.922	**0.989**	–	0.963	0.914	0.946	0.903	0.914
	1fieA	0.932	0.843	0.942	0.947	–	**0.963**	0.926	0.960	0.927	0.913
	1sesA	0.913	0.620	0.913	0.954	–	**0.982**	0.917	0.962	0.923	0.976
	1krn	0.895	0.908	0.895	**0.993**	–	0.960	0.942	0.960	0.892	0.972
	2fxb	0.985	0.941	0.985	0.951	–	0.970	0.978	0.978	0.978	**0.978**
	1amk	0.945	0.965	0.959	**0.997**	–	0.998	0.982	0.984	0.982	0.962
	1ar5A	0.946	0.812	0.946	0.971	–	0.974	0.942	0.968	0.954	**0.974**
	1gpb	0.947	0.868	0.948	0.982	–	0.983	0.976	0.984	0.983	**0.986**
	1taq	0.826	0.525	0.826	0.931	–	0.945	0.938	0.959	0.944	**0.965**
Ref. 2	2pia	0.766	0.761	0.768	0.763	0.730	0.826	0.847	0.850	0.839	**0.887**
	1pamA	0.757	0.755	0.758	0.623	0.66	0.859	0.857	**0.863**	0.853	0.845
	1aboA	0.65	–	–	0.489	**0.812**	0.796	0.723	0.791	0.679	0.762
	1idy	0.515	–	–	0.548	**0.997**	0.989	0.981	0.992	0.992	0.982
	1csy	0.154	–	–	0.154	0.735	0.764	0.731	**0.885**	0.831	0.872
	1r69	0.675	–	–	0.475	0.90	0.965	0.859	0.934	0.874	**0.968**
	1tvxA	0.552	–	–	0.448	0.891	0.920	0.944	**0.974**	0.944	0.971
	1tgxA	0.727	–	–	0.773	0.835	0.878	0.867	0.878	0.850	**0.884**
	1ubi	0.482	–	–	0.492	0.795	0.767	0.732	0.778	0.794	**0.825**
	1wit	0.557	–	–	0.694	0.825	0.851	0.875	0.815	0.774	**0.890**
	2trx	0.870	–	–	0.870	0.982	**0.986**	0.959	**0.986**	**0.986**	0.940
	1sbp	0.217	–	–	0.374	0.778	0.765	0.782	0.772	0.778	**0.796**
	1havA	0.480	–	–	0.448	0.792	0.879	0.884	0.846	0.884	**0.898**
	1uky	0.656	–	–	0.476	0.625	0.808	0.845	0.891	0.872	**0.894**
	2hsdA	0.484	–	–	0.498	0.745	0.796	**0.856**	0.829	0.742	0.762
	3grs	0.192	–	–	0.282	0.755	0.746	0.717	0.751	**0.781**	0.724
	Kinase	0.848	–	–	0.867	0.712	0.799	0.825	**0.888**	0.812	0.800

Table 2 (continued)

References	Name of dataset	CLUST-ALW/X	MSA-GA	MSA-GA w/prealign	SAGA	RBT-GA	GAPAM	VDGA_ Decomp_2	VDGA_Decomp_3	VDGA_Decomp_4	BSAGA
	1ajsA	0.324	–	–	0.311	0.892	0.899	**0.906**	0.905	0.902	0.857
	1cpt	0.660	–	–	0.776	0.584	0.875	0.869	0.812	0.853	**0.901**
	1lvl	0.746	–	–	0.726	0.567	0.781	0.803	0.819	0.816	**0.870**
	1ped	0.834	–	–	0.835	0.78	0.912	0.935	**0.947**	0.943	0.821
	2myr	**0.904**	–	–	0.825	0.675	0.822	0.806	0.830	0.808	0.810
	4enl	0.375	–	–	0.739	0.812	0.896	0.890	0.889	0.899	**0.901**
Ref. 3	Kinase	0.619	0.58	0.619	0.758	0.697	0.825	0.870	0.890	0.887	**0.890**
	1pamA	0.743	0.703	0.744	0.579	0.525	0.835	**0.853**	0.788	0.792	0.500
	1idy	0.273	–	–	0.364	0.546	0.601	0.446	0.599	0.569	**0.864**
	1r69	0.524	–	–	0.524	0.374	0.709	0.724	0.733	0.765	**0.795**
	1ubi	0.146	–	–	0.585	0.31	0.386	0.398	0.414	0.410	**0.567**
	1wit	0.565	–	–	0.484	0.780	0.758	0.833	0.873	0.867	**0.876**
	1uky	0.130	–	–	0.269	0.35	0.468	0.469	0.481	**0.526**	0.481
	1ajsA	0.163	–	–	0.186	0.18	0.311	0.383	0.453	0.408	**0.455**
	1ped	0.627	–	–	0.646	0.425	0.775	0.848	**0.893**	0.783	0.851
	2myr	0.538	–	–	0.494	0.33	**0.813**	0.586	0.651	0.519	0.564
	4enl	0.547	–	–	0.672	0.680	0.800	0.836	0.866	0.866	**0.872**
Ref. 4	1dynA	0.000	0.038	0.034	–	–	0.033	0.029	0.033	0.031	**0.051**
	Kinase2	0.630	**0.710**	0.635	–	–	0.384	0.330	0.542	0.478	0.550
Ref. 5	2cba	0.628	0.422	0.621	–	–	**0.852**	0.839	0.835	0.846	0.755
	S51	0.75	0.528	0.730	–	–	**0.835**	0.650	0.743	0.756	0.744

Table 3 Wilcoxon signed-rank test statistics for pairwise comparison of BSAGA with other alignment techniques

Alignment techniques	BSAGA less than	BSAGA greater than	Ties	*Z* value	*P* value (two-tailed)	if ($P < 0.05$)	Null hypothesis
MSA-GA	4	22	0	− 3.467	0.001	Yes	Reject
MSA-GA w/prealign	5	21	0	− 2.959	0.003	Yes	Reject
GAPAM	18	37	1	− 2.074	0.038	Yes	Reject
SAGA	13	39	0	− 4.412	0.000	Yes	Reject
RBT-GA	6	28	0	− 4.437	0.000	Yes	Reject
VDGA_Decomp_3	22	31	3	− .292	0.770	No	Retain
CLUSTALW	8	48	0	− 5.710	0.000	Yes	Reject

Table 4 Comparative alignment results of different approaches along with BSAGA on BAliBASE 3.0

	RV11		RV12		RV20		RV30		RV40		RV50		Overall	
	Q	TC	*Q*	TC	*Q*	TC	*Q*	TC	*Q*	TC	*Q*	TC	*Q*	TC
BSAGA	**0.810**	**0.622**	0.935	0.851	0.929	0.548	**0.908**	**0.656**	0.929	0.650	0.860	0.606	**0.895**	**0.655**
H4MSA	0.762	0.600	**0.952**	**0.890**	**0.941**	**0.558**	0.880	0.635	0.935	**0.668**	0.894	**0.636**	0.894	0.665
HMOABC	0.747	0.576	0.951	0.887	0.935	0.513	0.887	0.634	**0.938**	0.655	0.892	0.610	0.892	0.646
ClustalW	0.500	0.229	0.865	0.717	0.852	0.222	0.725	0.276	0.789	0.398	0.742	0.312	0.746	0.359
ClustalΩ (Sievers et al. 2011)	0.590	0.362	0.906	0.794	0.912	0.453	0.862	0.579	0.901	0.587	0.862	0.537	0.839	0.551
DIALIGN-TX (Subramanian et al. 2008)	0.505	0.268	0.882	0.757	0.878	0.308	0.761	0.389	0.834	0.452	0.821	0.470	0.780	0.441
FSA (Bradley et al. 2009)	0.503	0.272	0.924	0.822	0.865	0.189	0.689	0.263	0.861	0.478	0.789	0.420	0.772	0.408
FSA-maxsn (Bradley et al. 2009)	0.618	0.366	0.937	0.848	0.901	0.343	0.814	0.483	0.916	0.585	0.872	0.566	0.843	0.532
Kalign2	0.605	0.369	0.912	0.793	0.900	0.362	0.813	0.480	0.883	0.508	0.820	0.440	0.822	0.492
MAFFT-EINSi	0.660	0.440	0.936	0.839	0.926	0.451	0.861	0.592	0.914	0.575	0.899	0.598	0.866	0.583
MAFFT-GINSi	0.607	0.347	0.927	0.825	0.905	0.391	0.853	0.532	0.886	0.516	0.884	0.550	0.844	0.527
MAFFT-LINSi	0.671	0.450	0.936	0.842	0.926	0.457	0.855	0.573	0.919	0.601	0.899	0.566	0.868	0.582
MSAProbs (Liu et al. 2010)	0.682	0.444	0.946	0.870	0.928	0.469	0.865	0.611	0.923	0.610	**0.908**	0.610	0.875	0.603
MUMMALS	0.669	0.420	0.943	0.845	0.906	0.431	0.848	0.498	0.871	0.488	0.879	0.533	0.853	0.536
MUSCLE	0.683	0.441	0.945	0.862	0.928	0.479	0.875	0.623	0.925	0.604	0.894	0.593	0.875	0.600
ProbCons (Do et al. 2005)	0.670	0.420	0.941	0.860	0.917	0.412	0.845	0.547	0.900	0.536	0.894	0.579	0.861	0.559
PRANK (Loytynoja and Goldman 2005)	0.462	0.216	0.837	0.621	0.801	0.124	0.578	0.064	0.747	0.342	0.673	0.209	0.683	0.263
ProbAlign (Roshan and Livesay 2006)	0.695	0.457	0.946	0.867	0.926	0.444	0.853	0.569	0.922	0.612	0.889	0.555	0.872	0.584
T-Coffee (Notredame et al. 2000)	0.657	0.414	0.945	0.859	0.916	0.406	0.837	0.478	0.896	0.554	0.895	0.591	0.858	0.550

3.4 Result comparison of BSAGA with other approaches for BAliBASE 3.0

To compare the results of BSAGA with other methods for the version 3.0 of BAliBASE, we have considered the comparative results presented in H4MSA (Rubio-Largo et al. 2016b) and HMOABC (Rubio-Largo et al. 2016a) and listed here in Table 4 (bold faced data define best scores). Authors of those methods were already shown their methods performed better than others and proved that statistically which is well described in the respective papers. From Table 4, we observed that the overall performance of the proposed BSAGA was similar to H4MSA and HMOABC. However, for the subset RV11, having the lowest identity (0–20%), the BSAGA performed better than H4MSA and HMOABC. In addition to that, BSAGA

Table 5 Comparative alignment results of different approaches along with BSAGA on SABmark

	Superfamily		Twilight		Overall	
	Q	TC	*Q*	TC	*Q*	TC
BSAGA	**0.743**	**0.592**	**0.609**	**0.459**	**0.676**	**0.525**
H4MSA	0.737	0.589	0.583	0.436	0.660	0.512
ClustalW	0.588	0.372	0.315	0.151	0.452	0.262
ClustalΩ	0.617	0.414	0.355	0.181	0.486	0.298
DIALIGN-TX	0.571	0.349	0.299	0.123	0.435	0.236
FSA	0.531	0.311	0.248	0.104	0.399	0.208
FSA-maxsn	0.607	0.392	0.341	0.155	0.474	0.274
Kalign2	0.584	0.384	0.337	0.181	0.461	0.283
MAFFT-EINSi	0.631	0.422	0.377	0.181	0.504	0.302
MAFFT GINSi	0.631	0.409	0.379	0.195	0.505	0.302
MAFFT-LINSi	0.642	0.430	0.392	0.189	0.518	0.309
MSAProbs	0.662	0.459	0.428	0.228	0.545	0.344
MUMMALS	0.681	0.486	0.448	0.245	0.564	0.365
MUSCLE	0.655	0.449	0.414	0.215	0.535	0.332
ProbCons	0.655	0.449	0.425	0.223	0.540	0.336
PRANK	0.555	0.331	0.298	0.127	0.427	0.229
ProbAlign	0.654	0.441	0.424	0.226	0.539	0.334
T-Coffee	0.657	0.458	0.425	0.243	0.541	0.350

performed consistently well for other subsets as well. Moreover, H4MSA and HMOABC used multi-objective approach and produced a set of non-dominated solutions which is complicated in nature. In addition to that, both approaches used the deterministic heuristic method Kalign to improve the performance and reduce the chance to get stuck up in a local optimum. On the other hand, the proposed BSAGA optimized only the gap positions using GA and instead of producing non-dominated set of solutions, it produced two best solutions; one for SOP and another for TCC.

3.5 Result comparison of BSAGA with other approaches for SABmark

Similar to BAliBASE, we have compared the results of BSAGA with other methods for the SABmark v1.65 and listed in Table 5 (bold faced data define best scores). From Table 5, we observed that the performance of BSAGA was similar to H4MSA and H4MSA performed significantly better than others (Rubio-Largo et al. 2016b). However, the overall performance of BSAGA in terms of *Q* and TC scores was better than H4MSA. For both subsets (Superfamily and twilight), the BSAGA showed better *Q* and TC scores than H4MSA. The second subset 'twilight' represents the worst-case scenario of sequence alignment as the sequences share less than 25% identity (Van Walle et al. 2005). For that dataset also, the BSAGA obtained better results than other methods mentioned in Table 5.

4 Conclusion

MSA is an NP complete problem, and therefore, obtaining the global optimum solution is unrealistic. The complexity of an alignment structure is increased with the increased number and length of the sequences. In addition to that, MSA cannot be solved optimally by considering only one parameter or objective. MSA is defined by more than one parameter that needs to be optimized to obtain better alignment result. Many researchers are working to find a better way to solve an MSA using multi-objective optimization criteria. The proposed method BSAGA is an integer-based technique that simplifies the MSA problem by considering only the gap positions in an alignment without considering the sequences. This makes the method independent of the length of sequences. BSAGA optimizes only the positions of gap iteratively by applying the proposed GA operators on them to find an optimal alignment structure. BSAGA considered SOP and TCC for optimizing MSA. However, other multi-objective-based methods use a sorting technique to produce a set of solutions (Pareto optimal solutions) that requires a lot of computation and memory. BSAGA overcomes that limitation by selecting solutions from one part of the population based on SOP objective function while from the rest part based on TCC objection function. Therefore, it produces two best solutions (one based on SOP and other based on TCC) by the end of the method instead of producing a set of non-dominated solutions. This helps to reduce the computational complexity and memory requirement. User does not find difficulty to choose a best between the two solutions than choosing a best among a set of solutions. The proposed modified crossover and mutation operators also provided advantages for BSAGA. From the result section, it is shown that the BSAGA outperforms other existing heuristic alignment techniques irrespective of the datasets of BAliBASE and SABmark. Here, we have not considered the structural information of protein due to its limited availability. In future, this procedure can be modified by incorporating more MSA parameters that are beneficial to improve the MSA results. In addition, this method can be combined with other optimization technique to improve the performance even more.

Acknowledgements The authors would like to sincerely thank the reviewers for helpful and constructive suggestions to improve the quality of the paper. The authors would also like to thank Prof. Dr. Ansuman Lahiri of Biophysics, Molecular Biology and

Bioinformatics, University of Calcutta, for his helpful suggestions and ideas while doing this work.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Human and animal participants This article does not contain any studies with human participants or animals performed by any of the authors

References

Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE (2000) The protein data bank. Nucl Acids Res 28:235–242

Bradley RK et al (2009) Fast statistical alignment. PLoS Comput Biol 5:e1000392

Chuong BD, Kazutaka K (2008) Protein multiple sequence alignment. Methods Mol Biol 484:379–413

Corder GW (2009) Foreman DI: nonparametric statistics for non-statisticians: a step-by-step approach. Wiley, New York

Deb K et al (2002) A fast and elitist multiobjective genetic algorithm: Nsga-II. IEEE Trans Evol Comput 6:182–197

DeRonne KW, Karypis G (2013) Pareto optimal pairwise sequence alignment. IEEE/ACM Trans Comput Biol Bioinform 10:481–493

Do CB, Mahabhashyam MS, Brudno M, Batzoglou S (2005) ProbCons: probabilistic consistency-based multiple sequence alignment. Genome Res 15:330–340

Edgar RC (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucl Acids Res 32:1792–1797

Ehrgott M (2005) Multicriteria optimization. Springer, Berlin

Eusuff M, Lansey K, Pasha F (2006) Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Eng Optim 38:129–154

Feng DF, Doolittle RF (1987) Progressive sequence alignment as a prerequisite to correct phylogenetic trees. J Mol Evol 25:351–360

Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Boston

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

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

Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. Proc Natl Acad Sci 89:10915–10919

Heringa J, Taylor WR (1997) Three-dimensional domain duplication, swapping and stealing. Curr Opin Struct Biol 7:416–421

Hogeweg P, Hesper B (1984) The alignment of sets of sequences and the construction of phylogenetic trees: an integrated method. J Mol Evol 20:175–186

Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor

Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical report-tr06. Erciyes University, Engineering Faculty, Computer Engineering Department

Katoh K, Misawa K, Kuma K, Miyata T (2002) MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. Nucl Acids Res 30:3059–3066

Katoh K, Kuma K, Toh H, Miyata T (2005) MAFFT version 5: improvement in accuracy of multiple sequence alignment. Nucl Acids Res 33:511–518

Kaya M, Sarhan A, Alhajj R (2014) Multiple sequence alignment with affine gap by using multi-objective genetic algorithm. Comput Methods Prog Biomed 114:38–49

Kemena C, Taly JF, Kleinjung J, Notredame C (2011) STRIKE: evaluation of protein MSAs using a single 3D structure. Bioinformatics 27:3385–3391

Lam AY, Li VO (2010) Chemical-reaction-inspired metaheuristic for optimization. IEEE Trans Evol Comput 14:381–399

Lassmann T, Frings O, Sonnhammer ELL (2009) Kalign2: high performance multiple alignment of protein and nucleotide sequences allowing external features. Nucl Acids Res 37:858–865

Lee ZH, Su SF, Chuang CC, Liu KH (2008) Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. Appl Soft Comput 8:55–78

Lipman DJ, Altschul SF, Kececioglu JD (1989) A tool for multiple sequence alignment. Proc Natl Acad Sci 86:4412–4415

Liu Y, Schmidt B, Maskell DL (2010) MSAProbs: multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities. Bioinformatics 26:1958–1964

Loytynoja A, Goldman N (2005) An algorithm for progressive multiple alignment of sequences with insertions. Proc Natl Acad Sci 102:10557–10562

Miller BL, Golberg DE (1995) Genetic algorithms, tournament selection, and the effects of noise. Complex Syst 9:193–212

Mount DW (2004) Bioinformatics: sequence and genome analysis. Cold Spring Harbor Laboratory Press, Cold Spring Harbor

Narimani Z, Hamid B, Hassan A (2012) A new genetic algorithm for multiple sequence alignment. Int J Comput Intell Appl. https://doi.org/10.1142/S146902681250023X

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

Naznin F, Sarker R, Essam D (2012) Progressive alignment method using genetic algorithm for multiple sequence alignment. IEEE Trans Evol Comput 16:615–631

Needleman SB, Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 48:443–453

Notredame C (2002) Recent progress in multiple sequence alignment: a survey. Pharmacogenomics 3:131–144

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

Notredame C, Higgins DG, Heringa J (2000) T-Coffee: a novel method for fast and accurate multiple sequence alignment. J Mol Biol 302:205–217

Ortuño FM, Valenzuela O, Rojas F, Pomares H, Florido JP, Urquiza JM, Rojas I (2013) Optimizing multiple sequence alignments using a genetic algorithm based on three objectives: structural information, non-gaps percentage and totally conserved columns. Bioinformatics 29:2112–2121

Pei J, Grishin NV (2006) MUMMALS: multiple sequence alignment improved by using hidden Markov models with local structural information. Nucl Acids Res 34:4364–4374

Pei J, Grishin NV (2007) PROMALS: towards accurate multiple sequence alignments of distantly related proteins. Bioinformatics 23:802–808

Rahman RA, Ramli R, Jamari Z, Ku-Mahamud KR (2016) Evolutionary algorithm with roulette-tournament selection for solving aquaculture diet formulation. Math Probl Eng 2016:1–10

Roshan U, Livesay DR (2006) Probalign: multiple sequence alignment using partition function posterior probabilities. Bioinformatics 22:2715–2721

Rubio-Largo Á, Vega-Rodríguez MA, González-Álvarez DL (2016a) Hybrid multiobjective artificial bee colony for multiple sequence alignment. Appl Soft Comput 41:157–168

Rubio-Largo Á, Vega-Rodríguez MA, González-Álvarez DL (2016b) A hybrid multiobjective memetic metaheuristic for multiple sequence alignment. IEEE Trans Evol Comput 20:499–514

Sean RE (2002) A memory-efficient dynamic programming algorithm for optimal alignment of sequence to an RNA secondary structure. BMC Bioinform 3:13

Shyu C, Sheneman L, Foster JA (2004) Multiple sequence alignment with evolutionary computation. Genet Progr Evolvable Mach 5:121–144

Sievers F et al (2011) Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. Mol Syst Biol 7:539

Smith TF, Waterman MS (1981) Identification of common molecular sequences. J Mol Biol 147:195–197

Subramanian AR, Kaufmann M, Morgenstern B (2008) DIALIGN-TX: Greedy and progressive approaches for segment-based multiple sequence alignment. Algorithms Mol Biol 3:1–11

Taheri J, Zomaya AY (2009) RBT-GA: a novel metaheuristic for solving the multiple sequence alignment problem. BMC Genom 10(Suppl 1):S10

Thompson JD, Higgins DG, Gibson TJ (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice. Nucl Acids Res 22:4673–4680

Thompson JD, Plewniak F, Poch O (1999) BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs. Bioinformatics 15:87–88

Thompson JD, Koehl P, Ripp R, Poch O (2005) BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark. Proteins 61:127–136

Thompson JD, Linard B, Lecompte D, Poch O (2011) A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives. PLoS ONE 6:1–14

Van Walle I, Lasters I, Wyns L (2005) SABmark—a benchmark for sequence alignment that covers the entire known fold space. Bioinformatics 21:1267–1268

Wadud MS, Islam MR, Kundu N, Kabir MR (2018) Multiple sequence alignment using chemical reaction optimization algorithm. Int Conf Intell Syst Des Appl 941:1065–1074

Wang L, Jiang T (1994) On the complexity of multiple sequence alignment. J Comput Biol 1:337–348

Yamada S, Gotoh O, Yamana H (2006) Improvement in accuracy of multiple sequence alignment using novel group-to-group sequence alignment algorithm with piecewise linear gap cost. BMC Bioinform 7:524

Zhou A, Qu BY, Li H, Zhao SZ, Suganthan PN, Zhang Q (2011) Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol Comput 1:32–49

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.