Contents lists available at ScienceDirect

Genomics

journal homepage: www.elsevier.com/locate/ygeno

Review

A review on multiple sequence alignment from the perspective of genetic algorithm

Biswanath Chowdhury [a,*], Gautam Garai [b]

[a] Department of Biophysics, Molecular Biology and Bioinformatics, University of Calcutta, Kolkata, WB, 700009, India
[b] Computational Sciences Division, Saha Institute of Nuclear Physics, Kolkata, WB 700064, India

ARTICLE INFO

Article history:
Received 26 April 2017
Received in revised form 27 May 2017
Accepted 27 June 2017
Available online 29 June 2017

Keywords:
Multiple sequence alignment
Genetic algorithm
Multiobjective function

ABSTRACT

Sequence alignment is an active research area in the field of bioinformatics. It is also a crucial task as it guides many other tasks like phylogenetic analysis, function, and/or structure prediction of biological macromolecules like DNA, RNA, and Protein. Proteins are the building blocks of every living organism. Although protein alignment problem has been studied for several decades, unfortunately, every available method produces alignment results differently for a single alignment problem. Multiple sequence alignment is characterized as a very high computational complex problem. Many stochastic methods, therefore, are considered for improving the accuracy of alignment. Among them, many researchers frequently use Genetic Algorithm. In this study, we have shown different types of the method applied in alignment and the recent trends in the multiobjective genetic algorithm for solving multiple sequence alignment. Many recent studies have demonstrated considerable progress in finding the alignment accuracy.

Contents

1. Introduction

In an organism, DNA is the genetic material that acts as a medium to transmit genetic information from one generation to another [25]. All

* Corresponding author.
E-mail addresses: bchowdhury2410@gmail.com (B. Chowdhury), gautam.garai@saha.ac.in (G. Garai).

http://dx.doi.org/10.1016/j.ygeno.2017.06.007
0888-7543/© 2017 Elsevier Inc. All rights reserved.

the living things diverge over time from the common ancestor by evolution through changes in their DNA [8]. Hence, the ability to sequence the DNA of an organism is one of the most important and primary requirement in biological research. Many researchers have shown interest to invent tools that perform DNA sequencing. Earlier people used to sequence a few tens or hundreds of nucleotides at a time, however, DNA sequencing is now performed by high throughput sequencing machines with *billions* of bases being sequenced in a day. With the advent of "Next Generation" sequencing (NGS) [50] techniques, the throughput of sequence production increases many folds and reduces costs by orders of magnitude. All the nucleotide databases are now loaded with a vast amount of experimentally generated raw sequence data. In contrast to the sequencing techniques, experimental methods for structure determination are time-consuming, costly, and therefore, will not be able to keep pace with the flood of newly characterized sequences. Hence, the gap between sequences stored in databases and their corresponding functional and/or structural information is increasing rapidly. The use of various computational approaches is, therefore, the faster alternative to predict the structural and functional information of these nucleotide sequences. Presently, the computational biology and bioinformatics are very interesting fields in biological research. Sequence alignment (SA) is usually the first step performed in bioinformatics to understand the molecular phylogeny of an unknown sequence. This is done by aligning the unknown sequence with one or more known database sequences to predict the common portions as the residues perform functional and structural role tend to be preserved by natural selection in the course of evolution [93]. The optimum alignment arranges two or more sequences in such a way that a maximum number of identical or similar residues are matched [59]. The sequences may be nucleotide sequences (DNAs or RNAs) or amino acid sequences (Proteins). The rearrangement process may introduce one or more spaces or gaps in the alignment. A gap indicates a possible loss or gain of a residue; thus, evolutionary insertion or deletion (indel), translocations and inversion events can be observed in SA. Alignment or mapping is also an essential step after sequencing technique. Deep sequencing in NGS generally produces many *short reads* sequences (<~200 bases). To find the corresponding part of each sequenced read in its reference sequence, one needs to align the reads against the reference sequence. Alignment is also required for Gene annotation done by analyzing short RNA-seq reads derived from mRNA and mapping them to the reference genome. The sequence reads must evenly cover each transcript along its both ends. But the most challenging part is the Sequence reads are much shorter than the biological transcripts. Therefore, short reads are needed to align themselves to find the common end sequences among the reads and thus assembling them to construct the entire transcript and its coverage in reference genome along with the splice sites. Many short read aligners are developed in last few years like Bowtie2 [42], BWA-SW [48], and GSnap [88].

Two types of sequence alignment namely, pairwise sequence alignment (PSA) and multiple sequence alignment (MSA), are performed. PSA considers two sequences at a time whereas MSA aligns multiple (more than two) related sequences. MSA is more advantageous than PSA as it considers multiple members of a sequence family and thus provides more biological information. MSA is also a prerequisite to comparative genomic analyses for identification and quantification of conserved regions or functional motifs in a whole sequence family, estimation of evolutionary divergence between sequences and even for ancestral sequence profiling [40]. Sequence alignment at amino acid level is more relevant than nucleotide level as protein is the key functional biological molecules and hence carries structural and/or functional information [101]. The alignment, thus, has a strong connection to structural biology as well [57]. Hence, the SA, specifically MSA is the starting point of any biological macromolecular research field where MSA acts as an open window for viewing evolutionary, functional, and structural perspective of biological macromolecules in a concise format [12]. Different scoring methods are used in the sequence alignment to know the level of identity or similarity. Nucleotide scoring is a simple identification scheme where identical bases in both sequences are assigned positive scores. In contrast, for protein, a similarity score is also being counted (along with identity score) denoting the amino acids having similar physicochemical properties. The substitution matrices mostly consulted for protein sequence alignment are Point Accepted Mutation (PAM) [15], and BLOcked SUbstitution Matrix (BLOSUM) [26].

The method of SA can be of two types: global alignment [64] and local alignment [81]. Global alignment is done when the similarity is counted over the entire length of the sequences. Several MSA techniques accomplish global alignment [90,67] but difficulties arise when sequences are only homologous over local regions where clear block of ungapped alignment common to all of the sequences or if there is presence of shuffled domains among the related sequences [27]. In such cases, local alignment is performed to know the local similar regions among the sequences [4,45,56]. When there is a large difference in the lengths of the sequences to be compared, local alignment is generally performed [41].

For PSA, Dynamic Programming (DP) always provides the optimum alignment for a given objective function by finding the optimal alignment path using trace-back process [78,80]. The objective function is used for assessing the quality of alignment of a set of input sequences. The optimized alignment function is rarely biologically optimum when more than three sequences are considered as in MSA [65]. The computation is also a complex task and demands high computer resources [98]. Moreover, DP suffers from high-dimensional problems in MSA, as the number of sequences is equal to the number of dimensions. If two or more optimal paths are available and need to trace backward, the complexity of the back tracing grows exponentially. The computation of an exact MSA is NP-Complete and therefore the exact method is not applied in MSA for all but unrealistically small datasets [98]. Most MSA procedures are therefore heuristics or approximate in nature, which provides feasible alignment solution within a short and limited timeframe. As available heuristics do not provide the best solution and because of rapidly growing database sizes, development of new high performing MSA method to find good sequence alignment is still under research. MSAs are often hard to achieve because of the complex relationship often exists among related sequences, and sometimes because of lack of evolutionary history [41]. Three categories of approaches are frequently used in MSA namely, exact, progressive, and iterative alignment approaches. Exact algorithms usually deliver high quality alignment that very close to optimal [49,84]. It tries to simultaneously align multiple sequences and thus need to depend on DP. Due to the drawback of the exact method in alignment [99] as stated above, most MSA follows other two categories.

2. Progressive alignment

Hogeweg and Hesper [29] first formulated it. Progressive is a heuristics approach where complex MSA problem is separated into subproblems. This solves direct MSA problem indirectly with PSA. This approach assembles all sequences progressively where best pairwise alignment is first taken into account. Progressive alignment uses guide tree [20] to solve MSA problem where each leaf represents a sequence to be aligned. Each visited internal node is associated with an MSA of the sequences in its corresponding subtree. Finally, MSA of all considered sequences is associated with the root node (Fig. 1). Progressive alignment technique is used in several alignment programs such as MULTAL [86,87], MAP [33], PCMA [72], MULTALIGN [13], CLUSTAL [28, 91], T-Coffee [67], KAlign [44], MUMMALS/PROMALS [70,71], and others. Among them, the most widely used method is ClustalW [91]. It first performs the global pairwise alignment [64] of the sequences and develops a distance matrix. It then builds a guide tree based on the matrix values. Finally, it generates a consensus alignment by gradually adding sequences following the guide tree where the closest sequence

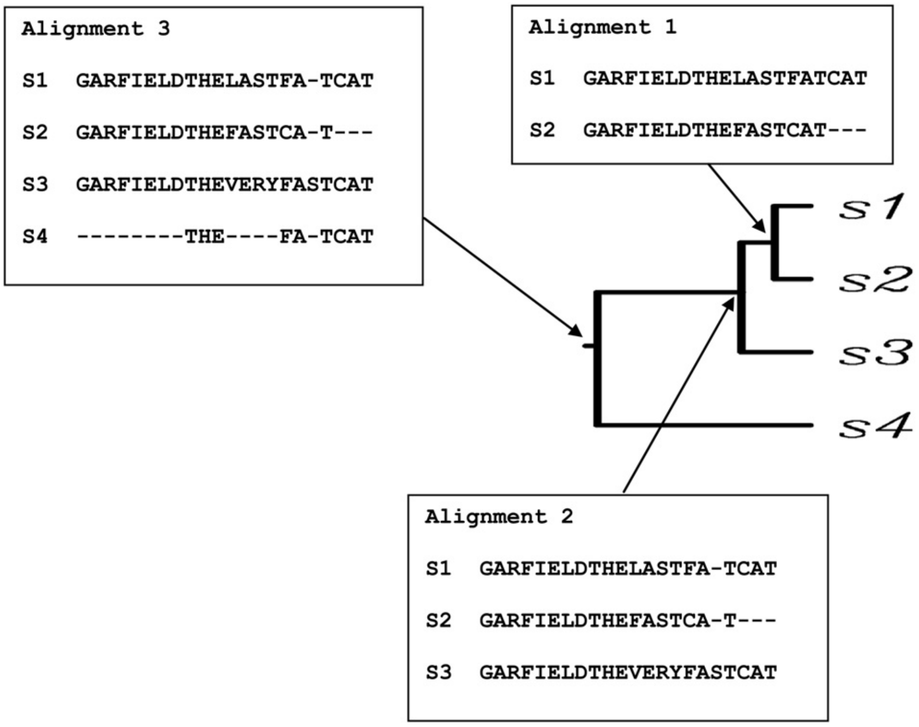

Alignment 3
S1 GARFIELDTHELASTFA-TCAT
S2 GARFIELDTHEFASTCA-T---
S3 GARFIELDTHEVERYFASTCAT
S4 --------THE----FA-TCAT
Alignment 1
S1 GARFIELDTHELASTFATCAT
S2 GARFIELDTHEFASTCAT---
s1
s2
s3
s4
Alignment 2
S1 GARFIELDTHELASTFA-TCAT
S2 GARFIELDTHEFASTCA-T---
S3 GARFIELDTHEVERYFASTCAT

Fig. 1. This is an example of how a progressive alignment performs MSA. The alignment structure and the guide tree are constructed by ClustalW. Each node in the guide tree is associated with an alignment. Based on the pairwise distance measures, s1 and s2 are aligned first because of smallest distance pairs (smallest branch length) and later s3 is added to them. Finally, the root node associates all the sequences considered in the MSA. The alignment shows that ClustalW failed to optimize the alignment of 'CAT' region of s2 with other sequences' 'CAT' regions.

pairs (smallest branch length in guide tree) are aligned first and thus, it gradually adds the next sequences. However, the greedy nature of these approaches cannot allow to modify the gaps and hence, the alignment cannot be altered in the later stage. There is a possibility that they can be trapped in local minima for this greediness [92]. Fig. 1 shows that ClustalW failed to get the optimum MSA due to its greedy nature in modifying the positions of gaps thus trapped in a local optimal alignment. Another drawback is that any progressive MSA is influenced by the initial alignment. As a result, any error made at that stage is propagated to the final MSA results. On the other hand, the iterative alignment methods iteratively modify the alignment by realigning the sequences or sequence groups and thus, overcome the drawback of the progressive method.

3. Guide tree

Guide tree guides the merging order of sequences based on the pairwise distances calculated for all the possible sequence pairs to be aligned in MSA (Fig. 1). For guide tree construction, UPGMA [82] or neighbor-joining [77] method is applied. However, the guide tree causes error in progressive alignment if an error is introduced while measuring the distances or at the time of tree construction. Ultimately, the error is reflected in the final alignment. The problem can be solved by iterative methods by repetitively modifying the guide tree and calculating the distance measurement. Hence, Iterative methods are biologically more sound approaches.

4. Iterative approach

The iterative method generally performs post-processing by making changes in the alignment made by progressive methods. It modifies the construction of guide tree. Programs that use guide tree reestimation are MAFFT [34,35], MUSCLE [18], PRIME [103], PRRP [24] and MUMMALS/PROMALS. They compute new distance matrices using an MSA obtained by progressive alignment. This, in turn, constructs a new guide tree that leads to a second round of progressive alignment (Fig. 2). Some iterative methods perform by repeatedly dividing the aligned sequences into two groups and then realigning those groups until the alignment process is converged [24,99].

During the refinement step of progressive alignment, MUSCLE divides the MSA into two sub-groups, each of which contains some of the input sequences as shown in Fig. 2. This is called "profile-profile alignment". In a profile, a multiple alignment is treated as a sequence by considering each column as a symbol as shown in Fig. 3. PRIME also performs group-to-group sequence alignment in the refining stage where groups are aligned by a pairwise method.

5. Hidden Markov model based alignments

One of the popular statistical models like Hidden Markov model is also taken into account for sequence alignment. One of the popular methods is ProbCons [16], which uses a new scoring function based on probabilistic consistency. It is also a progressive approach that uses a combination of probabilistic modeling and consistency-based alignment techniques. Therefore, by incorporating multiple pairwise sequence conservation information along with probabilistic consistency model, probcons able to produce improved results compared to other consistent methods. MUMMALS extends the ProbCons approach to allow for more sophisticated HMM structures.

Another probabilistic progressive method, PROMALS shows very effective for distantly related protein homologs where the sequence level identity is below 10%.

MSAprob [51] produces MSA based on pairwise posterior probability matrix. The posterior probability matrix is constructed using pair-HMM and partition function posterior probabilities computation. A pair-HMM calculates the pairwise probability matrix using Forward and Backward algorithms [17]. The partition function of alignments calculates another pairwise probability matrix by producing suboptimal alignments using DP.

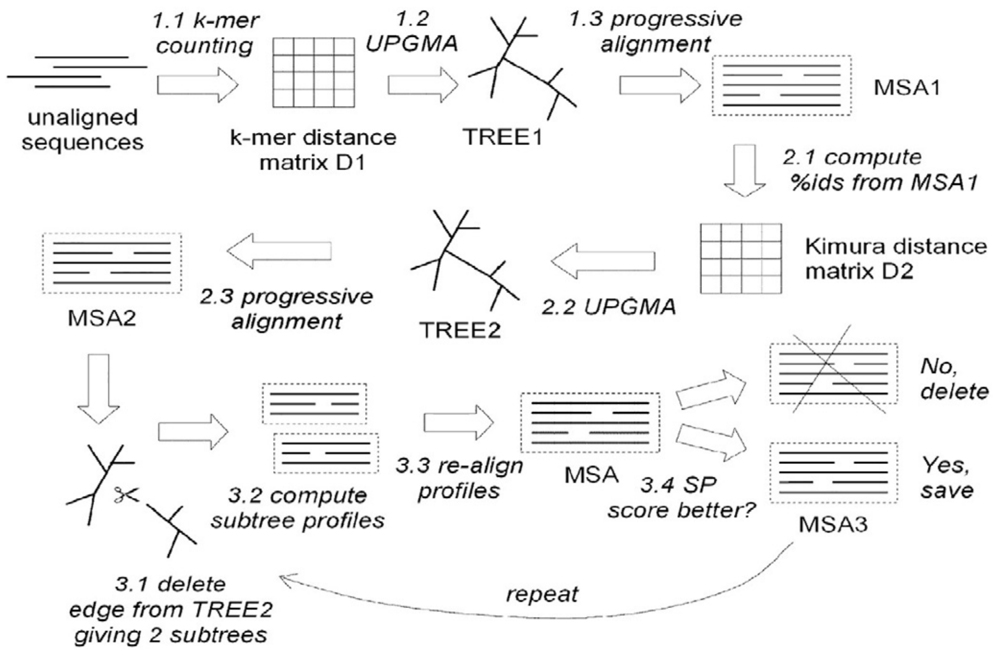

1.1 k-mer counting
1.2 UPGMA
1.3 progressive alignment
unaligned sequences
k-mer distance matrix D1
TREE1
MSA1
2.1 compute %ids from MSA1
Kimura distance matrix D2
2.2 UPGMA
TREE2
2.3 progressive alignment
MSA2
No, delete
Yes, save
3.1 delete edge from TREE2 giving 2 subtrees
3.2 compute subtree profiles
3.3 re-align profiles
MSA
3.4 SP score better?
MSA3
repeat

Fig. 2. This illustrates the flow of MUSCLE algorithm (adapted from [18]). The first stage is called *draft progressive* where the first progressive alignment or MSA1 (1.3) is built by following a guide tree (1.2) which is constructed based on a k-mer distance matrix (formed from the unaligned sequences) (1.1). Next, in the *improvement stage*, another distance matrix (2.1) is estimated (Kimura) from the MSA1 and a new MSA (MSA2) (2.3) is formed following the second guide tree (2.2). The last stage is called the *refinement stage* where the tree (formed in the previous stage) is divided into two subtrees (3.1) and profiles of two subtrees are computed (3.2) and realign them to form a full MSA (3.3). If the MSA score is improved then it is kept else discarded (3.4). The last refinement stage is repeated until convergence.

6. Pruning technique

To reduce the search space of MSA solution, some methods use pruning techniques. The principle of Divide and Conquer Algorithm (DCA) program [83,84] is to divide and concatenate the MSA. This approach first identifies the "optimal cut" points using pairwise projected alignments for partitioning a large multiple alignment into smaller subproblems. Each of the small parts is aligned separately and then joins to produce the final MSA. Similarly, some MSA programs [39,49] uses the Carrillo–Lipman bound [7] technique to determine constraints of an optimal multiple alignment.

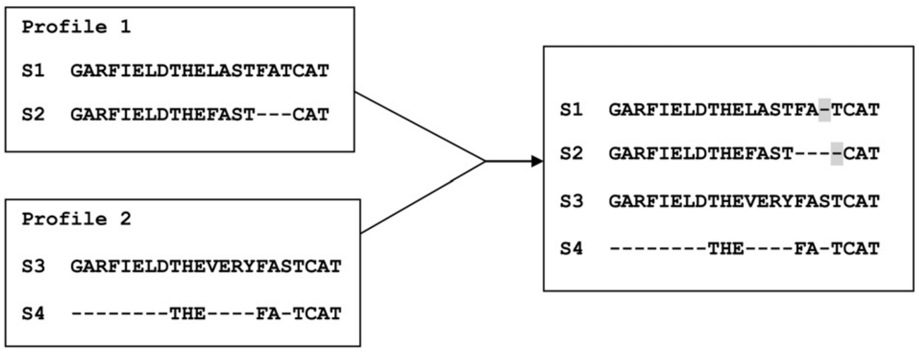
7. Alignment scoring technique

The sequence alignment techniques quantitatively measure the quality of an alignment by considering a scoring model. The most commonly used scoring model chosen by several MSA methods [89,43,73] is Sum Of Pair (SOP).

7.1. Sum Of Pair

It is an extension of the typical pairwise scoring technique to a multiple alignment approach. Here, a pairwise matched residue gets a positive score, a mismatch gets a negative score (for protein substitution scoring matrices are consulted as described before), and a space or gap gets a negative score. Nevertheless, it has a major drawback as the computational time and required memory grows exponentially with the number of sequences [98] to maximize the score. Therefore, heuristic, mostly progressive alternative are required for input data [20]. For MSA, all the possible sequence pairs are scored first based on pairwise scoring and after that, all the paired scores are summed to get the total SOP score. Total number of pairwise projections to be done for k numbers of sequences are $k \times (k-1)/2$ [60]. The SOP score is defined by the equation as follows.

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



```
Profile 1

S1   GARFIELDTHELASTFATCAT

S2   GARFIELDTHEFAST---CAT

Profile 2

S3   GARFIELDTHEVERYFASTCAT

S4   --------THE----FA-TCAT
```

```
S1   GARFIELDTHELASTFA-TCAT

S2   GARFIELDTHEFAST----CAT

S3   GARFIELDTHEVERYFASTCAT

S4   --------THE----FA-TCAT
```

Fig. 3. Two profiles (1 and 2) are aligned to each other in such a way that the columns are conserved in the results. When two profiles are aligned with each other then gaps (gray background) are inserted for proper alignment. However, the aligned columns in profile 1 and 2 are not destructed. The scores of aligned columns are determined by the profile function, which assigns a high score to a column containing similar amino acids.

where δ (S_i, S_j) is the pairwise alignment score between the two aligned sequences S_i and S_j; k is the total number of sequences. The process of SOP score evaluation is shown in Fig. 4.

7.2. Gap penalty

In natural evolution, indel is a rare process among homologous sequence and therefore gap should be penalized in the scoring. A number of programs (e.g., CLUSTALW, MUSCLE, and MAFFT) use biases to the gap positions and try to place gaps where previous gaps were opened during each pairwise merge step (Fig. 4). The rationale is that gap opening events in a group of sequences likely represent a single evolutionary event. In addition, gaps are unlikely allowed to the hydrophobic groups of amino acids which fold properly to get inside the protein core, whereas gaps are tolerable to the surface groups of hydrophilic amino acids of a protein. In the evolutionary process, the indels comprise an entire subsequence and often occur from a single mutational event rather than many small indels. Hence, an opening gap is penalized more than extending gaps and thus allows more trailing gaps once a gap is opened. This scoring scheme is called affine gap penalty. It is evaluated as follows.

$$W = \gamma + \delta \times (k-1) \qquad (2)$$

where W is the total gap penalty, γ is the opening gap penalty, δ is gap extension penalty, and k is the total gap length.

7.3. Consistency based scoring

The drawback of SOP score in progressive alignment is that the SOP scoring system cannot overcome the greedy nature of progressiveness. An alternate methodology that improves the situation is the formulation of 'consistency based' scoring [23,68,97].

The objective of such scoring scheme is to improve the early alignments stages by incorporating information of outgroup sequences during each pairwise alignment. The pairwise sequence information guides to build MSA. For e.g., the pairwise alignment of sequences 1 and 2 and the alignment of sequences 2 and 3 will guide to construct the alignment of sequences 1 and 3 (Fig. 5). The pairwise alignments consist of global as well as local alignments. Thus, MSA is built that maximize the consistency with pairwise alignments. This scoring is used in the T-Coffee, DIALIGN [55,56], PCMA, MUMMALS, PROMALS, and Align-m [95] alignment algorithms. The layout of T-Coffee is illustrated in Fig. 6. However, this scoring scheme is also not free from any limitation. The main drawback of this scoring method is its higher time complexity.

7.4. Probabilistic scoring

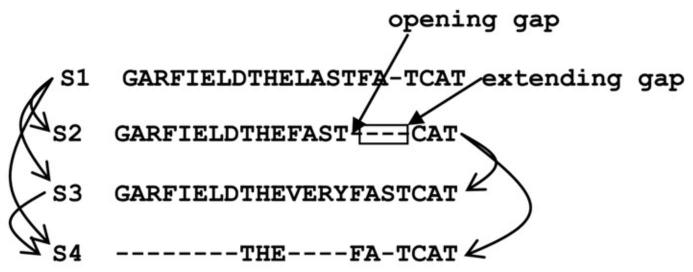
Other than SOP and Consistency based scoring methods, different probabilistic scoring models are also applied. For e.g. MUSCLE uses a profile function called the log expectation score to apply pairwise alignment to profiles. This scoring is based on background probability of an amino acid and its joint probability with another amino acid being aligned to each other. Other statistical based alignment methods use probabilistic consistency model, pairwise posterior probability as described in Section 5.

8. Stochastic approaches

Since solving an MSA using deterministic approach is a very complex optimization task, stochastic methods are now a very promising alternative field of research. The stochastic method is useful for complex, poorly defined optimization problems. Evolutionary Algorithms (EAs) are one of the stochastic approaches that are iterative in nature. EAs are optimization techniques that are guided by domain or problem specific knowledge. For SA problem, EA based algorithms produce an alignment and then try to improve it over successive iterations. EAs received major attention for SA problem by many researchers over the decades. The EA based approaches used in SA include simulated annealing [10,31,38], Tabu Search [75,76], Ant Colony Algorithm [9,58,100], Bee Colony Optimization [47,102]. Other than these above-mentioned EA methods, Genetic Algorithm (GA) is possibly the most popular and familiar optimization technique.

8.1. Genetic Algorithm

GA [21,30] is a stochastic method, inspired by the natural genetic mechanisms. It follows the "survival of the fittest" mechanism. It performs well for optimizing a complex, large, and/or multidimensional problem and provides an optimal or a near-optimal solution. Initially, the search starts with a set of probable solutions of a problem in the search space. Solutions are encoded by a set of strings called *chromosomes*. The collection of such chromosomes is called the *population*. Each string of the population is associated with an *objective function* (OF), which evaluates the fitness of a string and determines how much it fits to go to the next generation. Hence, the objective function is named as the *fitness function*. The fittest strings are selected based on their fitness. Genetic operators like *crossover* and *mutation* are then performed on the selected strings to yield the next generation. The above processes of selection, crossover, and mutation are iteratively continued until the maximum number of generations is reached or


opening gap
S1 GARFIELDTHELASTFA-TCAT
extending gap
S2 GARFIELDTHEFAST---CAT
S3 GARFIELDTHEVERYFASTCAT
S4 --------THE----FA-TCAT

SOP = δ (S1, S2) + δ (S1, S3) + δ (S1, S4) +δ (S2, S3) +δ (S2, S4) + δ (S3, S4).

i =	**1**	**1**	**1**	**2**	**2**	**3 (*k*-1)**
j=	**2**	**3**	**4**	**3**	**4**	**4 (*k*)**

Total number of pairwise comparisons = 4×(4-1)/2 = 6.

Fig. 4. An example of the process of SOP score evaluation.

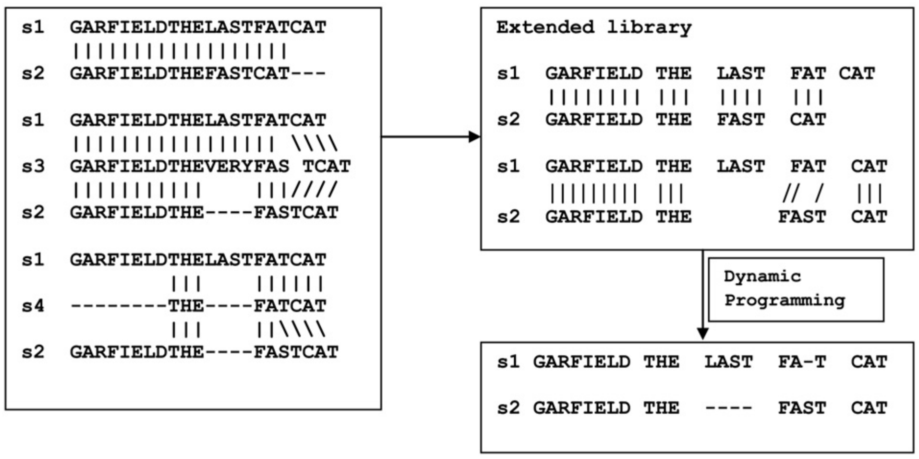


Fig. 5. This figure indicates how the pairwise alignment (S1 and s2) is modified based on the residual information of the other sequences (s3 and s4). This process is called alignment library extension. The three possible alignments of s1 and s2 are shown (s1-s2, s1-s3-s2, s1-s4-s2). These alignments are combined to get an extended library. This library is resolved by getting the correct alignment using dynamic programming.

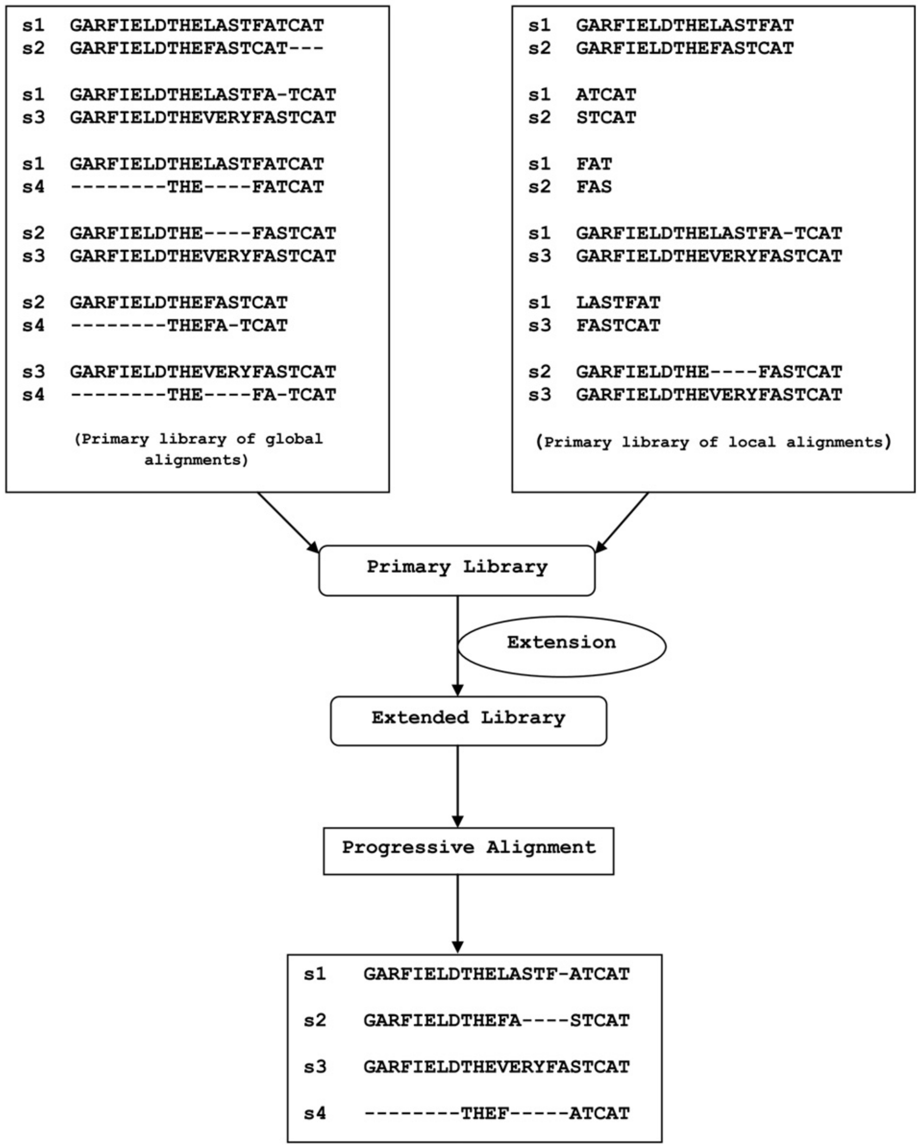

s1 GARFIELDTHELASTFATCAT
s2 GARFIELDTHEFASTCAT---
s1 GARFIELDTHELASTFA-TCAT
s3 GARFIELDTHEVERYFASTCAT
s1 GARFIELDTHELASTFATCAT
s4 --------THE----FATCAT
s2 GARFIELDTHE----FASTCAT
s3 GARFIELDTHEVERYFASTCAT
s2 GARFIELDTHEFASTCAT
s4 --------THEFA-TCAT
s3 GARFIELDTHEVERYFASTCAT
s4 --------THE----FA-TCAT
(Primary library of global alignments)
s1 GARFIELDTHELASTFAT
s2 GARFIELDTHEFASTCAT
s1 ATCAT
s2 STCAT
s1 FAT
s2 FAS
s1 GARFIELDTHELASTFA-TCAT
s3 GARFIELDTHEVERYFASTCAT
s1 LASTFAT
s3 FASTCAT
s2 GARFIELDTHE----FASTCAT
s3 GARFIELDTHEVERYFASTCAT
(Primary library of local alignments)
Primary Library
Extension
Extended Library
Progressive Alignment
s1 GARFIELDTHELASTF-ATCAT
s2 GARFIELDTHEFA----STCAT
s3 GARFIELDTHEVERYFASTCAT
s4 --------THEF-----ATCAT

Fig. 6. This illustrates the layout of T-Coffee. Pairwise local and global alignments are first computed to produce a primary library which then extended to be used for MSA construction in a progressive manner.

any other termination condition is satisfied. Thus, GA is able to find the best solution (chromosome) which is either global or near global optimal solution. The GA is characterized by four parameters namely, population size, crossover rate, mutation rate, and elitist selection. The fitness function defines the problem. The advantage of GA is that the fitness function is clearly separated from the other parts of the algorithm and therefore, the only modification in the fitness function is enough to change the GA for another new problem.

To solve the biological sequence alignment problem, several researchers have applied GA other than the conventional approaches. The popular software developed by Notredame and Higgins [66] is SAGA. The SAGA was tested with two different OFs, one is SOP scoring with natural affine gap penalty and other OF is SOP with quasi-natural affine gap penalties. In the quasi-natural gap penalties, an additional gap opening penalty is charged for any gap in a sequence that starts after and ends before a gap in the second sequence. SAGA uses two types of crossover: one-point and uniform. It uses 20 different types of complex mutation operators other than two crossover operations to obtain optimum alignment. Mutation operations primarily rearrange the positions of gaps in MSA by adding or shuffling them. The primary drawback of this algorithm is that it is time consuming due to the repeated use of the fitness function.

Zhang and Wong [104] implemented GA for sequence alignment along with schema concept of GA. A schema describes a subset of strings that have the same elements at certain positions on the strings. In this approach, GA conducts sequence alignment by identifying matches and mismatches (deletions, insertions, and substitutions). Alignment was based on the number of fully matched columns. This approach focused on the identification of matched columns and therefore, it is applicable to only very highly similar sequences and performs poorly for low similarity sequences.

The Global Criterion for Sequence Alignment (GLOCSA) [3] is used to assess the MSA quality of DNA sequences. It is a global approach that rates the alignment as a whole where all sequences considered simultaneously without taking individual pairwise alignment scores. GA is the useful optimizer of this new scoring function and hence, the proposed method is named as GLOCSA-Guided Genetic Algorithm (GGGA). GLOCSA is composed of three criteria: 1) it rates the whole alignment at a time with the *Mean Column Homogeneity*, 2) it favors fewer larger blocks of concentrated *gaps* more with *Reciprocal of Gap Blocks*, and 3) it favors smaller alignment matrices with *Columns Increment*. *Mean Column Homogeneity* weights more the alignment columns those contain same residues than a column that has more diversity in the residues. GGGA represents a compact alignment using five different mutation operators without any crossover. GGGA can be used for completely unaligned sequence sets. However, it is more efficient for refining alignments produced by other existing tools (for e.g., MUSCLE was used for pre-alignment).

Naznin et al. [63] also proposed a GA based method named Vertical Decomposition with Genetic Algorithm (VDGA) for MSA where they divide the sequences vertically into two or more subsequences and solve them individually using a guide tree approach. Finally, they combine them all to generate a new MSA. This decomposition of the solution is applied in the initial generation and in child generations as well.

Naznin et al. [62] developed another progressive alignment method using genetic algorithm (GAPAM) where the initial population was generated by randomly created guide trees in three stages. In the first stage, the guide tree is generated by using the DP distance table where the distance values are calculated from mismatches in pairwise alignments. This is followed by MSA construction from the guide tree. Based on the MSA, another distance table, called *kimura* distance table is constructed followed by a guide tree and MSA in the second stage. In the third stage, guide trees are generated by randomly selected sequences from either the stage one or the stage two generated guide trees. Then again, it shuffles the sequences inside the trees. The main drawback of this method is that it requires large computation to generate only the initial population.

Shyu and Foster [79] developed an encoding scheme of chromosome inspiring from the concept of biological consensus sequence. The chromosomes are broken into twenty pieces to represent 20 amino acids and four pieces for 4 nucleotides. Hence, this technique employs a chromosome into a number of parallel chromosomes, where each of them represents the relative occurrences and locations of a particular residue in an alignment. The fitness is determined by how the chromosomes fit together to derive the final alignment. The advantage of this encoding is that the generation number is independent on the number of sequences being aligned. Nevertheless, the performance fluctuated with increasing sequence length and GA frequently failed to converge. The search space grows exponentially with the increasing number of parallel chromosomes and therefore GA evolves much slower for protein sequence alignment.

MSA-GA [22] is another GA based method where the initial population is seeded with the prealigned sequences using the alignment results of Needleman–Wunsch algorithm. For initial seeding, it does have the problem of being entrapped in the local optima. The method also has an option to follow a greedy hill climbing algorithm [53] to improve the alignments. The hill-climbing algorithm evaluates the objective function by successively adding extra gaps in a block of gaps. When the addition of gap worsens the results, the gap is removed and it moves on to the next block. The hill climbing algorithm makes this method very computationally expensive.

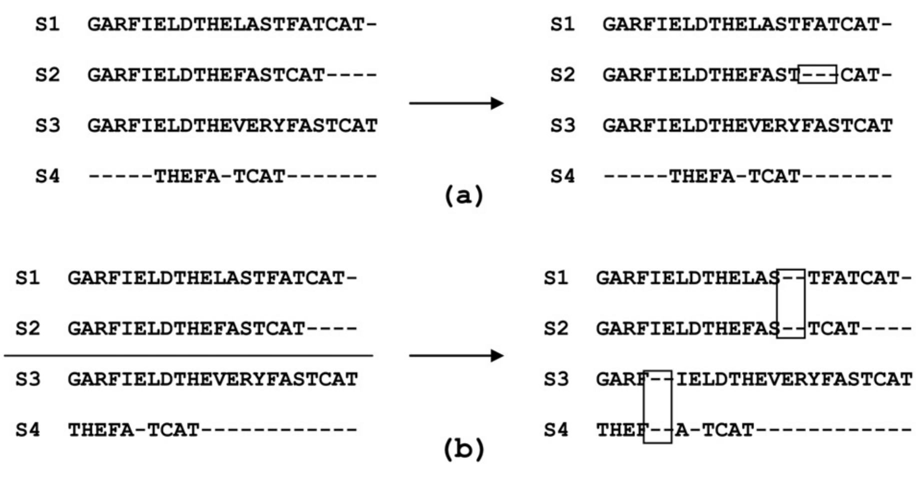

S1 GARFIELDTHELASTFATCAT-
S2 GARFIELDTHEFASTCAT----
S3 GARFIELDTHEVERYFASTCAT
S4 -----THEFA-TCAT-------
S1 GARFIELDTHELASTFATCAT-
S2 GARFIELDTHEFAST---CAT-
S3 GARFIELDTHEVERYFASTCAT
S4 -----THEFA-TCAT-------
(a)
S1 GARFIELDTHELASTFATCAT-
S2 GARFIELDTHEFASTCAT----
S3 GARFIELDTHEVERYFASTCAT
S4 THEFA-TCAT------------
S1 GARFIELDTHELAS--TFATCAT-
S2 GARFIELDTHEFAS--TCAT----
S3 GARF--IELDTHEVERYFASTCAT
S4 THEF--A-TCAT------------
(b)

Fig. 7. Gap insertion operation; (a) in this operation sequence 2 is selected randomly and a block of gaps (within box) is inserted at a random position; (b) this divides the alignment horizontally into two groups and a block of gaps of random size is inserted at random position in each of the two groups.

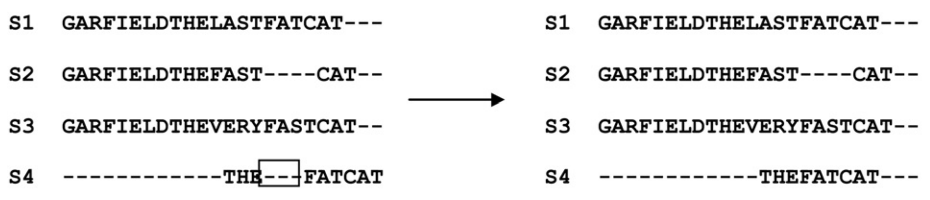

S1 GARFIELDTHELASTFATCAT---
S2 GARFIELDTHEFAST----CAT--
S3 GARFIELDTHEVERYFASTCAT--
S4 ------------THE---FATCAT
S1 GARFIELDTHELASTFATCAT---
S2 GARFIELDTHEFAST----CAT--
S3 GARFIELDTHEVERYFASTCAT--
S4 ------------THEFATCAT---

Fig. 8. In the gap deletion operation, a gap block (within the rectangular box) of sequence 4 is randomly selected and deleted.

Narimani et al. [61] proposed a new method where population initialization does not completely rely on the other heuristic algorithm. Part of the population is generated randomly and the rest of it by the clustalW method. This is an intermediate approach between a completely random and heuristic method. Hence, it overcomes the situation to be trapped in local optima for complete initial seeding. It also introduced a new Cross-mutation for better vertical recombination.

PWMAligner [6] generates a population of Positional Weight Matrices (PWMs) and each of them represents multiple alignment profile of the input sequences. The matrix uses each row to represent each residue and each column for each residual position in the MSA profile. A PWM calculates scores at each position independently from the symbols at other positions. It implements GA as the optimizing tool to optimize the PWM scoring to find the optimum alignment.

Agarwal and Chauhan [1] introduced a new genetic algorithm in their method for solving MSA where only the probable gap positions in an alignment are used to construct a chromosome.

Several other approaches are also present where GA is combined with another optimization technique to improve the alignment technique. RBT-GA [85] is one such approach, which is based on the combination of a novel Rubber Band Technique and a Genetic Algorithm for solving Multiple Sequence Alignment problem. RBT's Sticky Poles are used to identify the most likely biologically related locations in the input sequences (motifs) of a chromosome in a population. GA is then used to improve the quality of different solutions.

In another approach named GA-ACO [46], the Ant Colony Optimization is used as a local search after the GA operation to escape from local optima. ACO is dependent on the searching behavior of biological ants that find the shortest route from a food source to the nest. The ACO search process moves on the sequences to choose the matching residues. The selected probability of a residue or a gap is determined by the matching score (pheromone) deposited on matching residues. If the larger amount of pheromone is left on a route, the greater is the probability of selecting the route. Finally, the highest alignment score is determined with an analogy of the shortest route traveled by the ants. Once the ants determine the optimal route, the stopping criterion is reached. In GA-ACO, a sliding window technique is used to find the mismatch blocks and ACO is applied to improve those poorly aligned regions.

Chen et al. [11] proposed a partitioned approach to divide the set of sequences into several subsections vertically using an automated partitioning strategy. GA is used to find the optimum cut-off points for partitioning in every iteration. ACO is used to align the sequences of each subsection. Finally, the MSA is obtained by assembling the result of each subsection.

The above-mentioned approaches so far discussed are based on sequential GA. Apart from the sequential GA, *parallelism* is an added advantage of GA. It not only speeds up the execution, but also improves the efficiency in convergence. *Parallelism* does not improve the quality of the solution. However, to improve the quality of solutions, an island parallel genetic algorithm (iPGA) is proposed [2,52]. It is inspired by the natural process of migration. In this approach, the total population of GA is subdivided into a number of subpopulations and each of them is assigned to different processors that are connected with each other. Let us consider the size of each subpopulation be n and the number of processors be P, then the total population size should be $n \times P$. The initial subpopulation at each processor is created randomly. The best part of this approach is the migration, which actually allows a specific number of individuals (migrants) of one subpopulation to be hopped to other subpopulation of another processor after executing a certain number of generations. This migration continues separately for each subpopulation for a certain number of generations. An overlapping generation technique is used where half of the population of parent generation survives in the next generation and the child population will replace the other half having the weakest individuals during each generation.

8.1.1. Multi-objective function optimization

At present, MSA is not considered as a research problem of a single objective optimization. Instead, MSA is now treated as a multiobjective optimization problem. In this technique, each criterion, which needs to be optimized, is treated as a separate objective function. The optimum solution of an MSA is not obtained by considering only one criterion

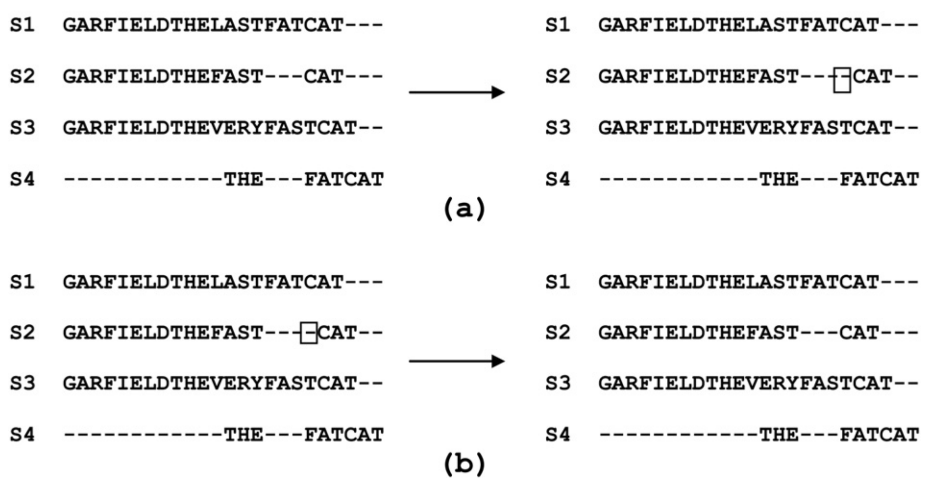

S1 GARFIELDTHELASTFATCAT---
S2 GARFIELDTHEFAST---CAT---
S3 GARFIELDTHEVERYFASTCAT--
S4 ------------THE---FATCAT
S1 GARFIELDTHELASTFATCAT---
S2 GARFIELDTHEFAST----CAT--
S3 GARFIELDTHEVERYFASTCAT--
S4 ------------THE---FATCAT
(a)
S1 GARFIELDTHELASTFATCAT---
S2 GARFIELDTHEFAST----CAT--
S3 GARFIELDTHEVERYFASTCAT--
S4 ------------THE---FATCAT
S1 GARFIELDTHELASTFATCAT---
S2 GARFIELDTHEFAST---CAT---
S3 GARFIELDTHEVERYFASTCAT--
S4 ------------THE---FATCAT
(b)

Fig. 9. This shows the increment of gap length by one unit in (a); and decrement of gap length by one unit in (b). Sequence 2 and one of its gap block is selected randomly and a gap is inserted to or deleted from the block is highlighted with a square box.

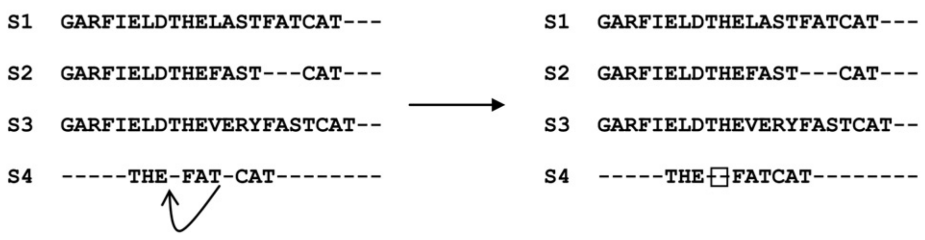

S1 GARFIELDTHELASTFATCAT---
S2 GARFIELDTHEFAST---CAT---
S3 GARFIELDTHEVERYFASTCAT--
S4 -----THE-FAT-CAT--------
S1 GARFIELDTHELASTFATCAT---
S2 GARFIELDTHEFAST---CAT---
S3 GARFIELDTHEVERYFASTCAT--
S4 -----THE--FATCAT--------

Fig. 10. This illustrates the merge gap operation. Here two discretely placed gaps in sequence 4 are merged together as indicated by an arrow.

like alignment scoring but will be obtained by optimizing multiple criteria simultaneously. However, improvement of one objective may deteriorate one or more other objectives. For e.g., an MSA with better alignment score may contain a very large numbers of gaps, which is biologically insignificant; or may contains lower numbers of conserve column in an alignment which is also undesirable. Thus, getting a single solution having all the optimized objectives simultaneously is merely impossible. Instead, a set of solutions is considered [105] as defined in multiobjective optimization. The main important feature of this technique is the selection procedure that based on a non-dominated solution selection. This non-dominated set of solutions is called the Pareto optimal solutions. The Pareto set of solutions consists of all those solutions where no other solution exists that can improve one of the objective functions without worsening some other objective [19]. In other words, to define dominancy, a solution x_i dominates another solution x_j if the following two conditions are true:

- the solution x_i is not worse than x_j in all objectives;
- the solution x_i is strictly better than x_j in at least one objective.

Therefore, it is very hard to achieve from the computational viewpoint.

Recently few works have been done on multiobjective MSA (MMSA) where GA is used as an optimizer of different objectives. The MO-SAStrE [69] is based on the nondominated sorting genetic algorithm that considers three objectives functions: STRIKE score, non-gaps percentage, and total conserved columns. The STRIKE score [37] helps to accurate the alignment by using at least one known structure, retrieved from the PDB [5]. It finds the contacts between amino acids in an alignment considering the structure. This method is proven to provide very promising alignment solution but the main drawback is the limited availability of the 3D structures.

In another multiobjective approach, MSAGMOGA [36], the large numbers of non-dominated alignments are obtained with respect to three objectives: affine gap penalty minimization and similarity maximization and support maximization. The similarity measure is done by constructing PWMs to observe the dominance value of each residue to represent an alignment column. By support maximization, it defines the inclusion of a number of sequences in an alignment that increases the quality. Greater support value means the stronger alignment covered by the included sequences.

The parallel genetic algorithm has also been used in MSA for solving with multiple objectives. Parallel Niche Pareto AlineaGA [14] is an example of a multiobjective GA that searches for the pareto front. This approach considers both the sum-of-pairs and the identity of the alignments. It used migration technology, hence, belongs to the class of iPGA as discussed before in parallel-GA (Section 8.1). The niching method helps to distribute the population of solutions and thus increases the diversity of the population. This allows parallel convergence into different good solutions by reducing the fitness of highly similar solutions. Equivalence Class Sharing (ECS) [32] is used for niching technique. For selection, two randomly selected individuals are evaluated against a set of individuals or solutions based on the fitness function. The candidate who dominates all the solutions in the comparison set is selected. However, when both candidates are eligible then ECS sets the priority towards the individual that presents the smallest number of solutions in its neighborhood.

All the above-mentioned GA-based methods (single and multi-objective) are mostly dependent on the mutation operation since it plays a very significant role in solving MSA problem. The objective of mutation operation is to maintain diversity within the chromosomal population and thus, reduces the chance of premature convergence to a local optimal solution. Mutation exploits the search space by inducing random walk through it.

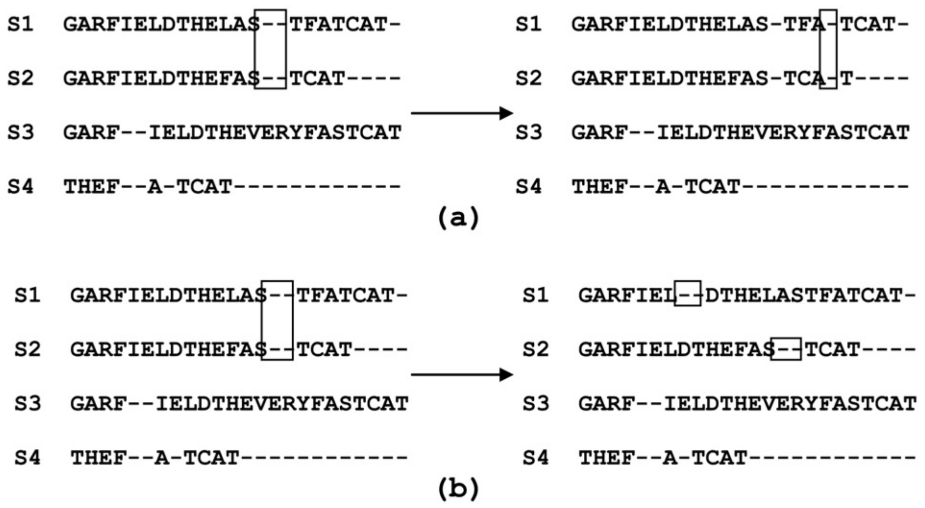

S1 GARFIELDTHELAS--TFATCAT-
S2 GARFIELDTHEFAS--TCAT----
S3 GARF--IELDTHEVERYFASTCAT
S4 THEF--A-TCAT-----------
S1 GARFIELDTHELAS-TFA-TCAT-
S2 GARFIELDTHEFAS-TCA-T----
S3 GARF--IELDTHEVERYFASTCAT
S4 THEF--A-TCAT-----------
(a)
S1 GARFIELDTHELAS--TFATCAT-
S2 GARFIELDTHEFAS--TCAT----
S3 GARF--IELDTHEVERYFASTCAT
S4 THEF--A-TCAT-----------
S1 GARFIEL--DTHELASTFATCAT-
S2 GARFIELDTHEFAS--TCAT----
S3 GARF--IELDTHEVERYFASTCAT
S4 THEF--A-TCAT-----------
(b)

Fig. 11. An example of split gap mutation operation; (a) this operation splits a gap block vertically into two and one of them is shifted and highlighted using a box diagram; (b) the gap block is split horizontally and one of them is moved to another position as indicated by a box shape.

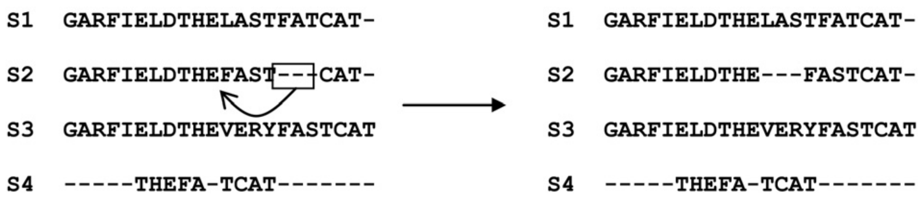

S1 GARFIELDTHELASTFATCAT-
S2 GARFIELDTHEFAST---CAT-
S3 GARFIELDTHEVERYFASTCAT
S4 -----THEFA-TCAT-------
S1 GARFIELDTHELASTFATCAT-
S2 GARFIELDTHE---FASTCAT-
S3 GARFIELDTHEVERYFASTCAT
S4 -----THEFA-TCAT-------

Fig. 12. In this gap shuffling operation, the sequence two is randomly selected and its gap block is assigned to a different position in that sequence as indicated by an arrow.

8.1.2. Mutation operation

In an alignment, the biological sequences consist of residues and a probable number of gaps. Hence, to improve the MSA result, one needs to optimize the arrangement and the number of gaps inside an alignment. Mutation does this work by inserting, deleting, increasing, decreasing, merging, splitting, and shuffling the gaps in the aligned sequences.

Table 1
Comparative alignment results of different GA-based algorithms along with Clustal W/X on BAliBASE 2.0.

Reference	Name of dataset	CLUSTALW/X	MSA-GA	MSA-GA w/prealign	SAGA	RBT-GA	GAPAM	VDGA_Decomp_2	VDGA_Decomp_3	VDGA_Decomp_4	MO-SAStrE
Ref. 1	1idy	0.500	0.427	0.438	0.342	–	0.565	0.550	0.651	**0.654**	–
	1tvxA	0.042	0.295	0.209	0.278	–	0.316	0.316	**0.316**	0.310	–
	1uky	0.392	0.443	0.405	**0.672**	–	0.402	0.416	0.459	0.464	0.403
	Kinase	0.479	0.295	0.488	**0.862**	–	0.487	0.531	0.545	0.548	0.808
	1ped	0.592	0.501	0.687	**0.746**	–	0.498	0.443	0.482	0.451	0.716
	2myr	0.296	0.212	0.302	0.285	–	0.317	0.347	0.359	0.282	**0.544**
	1ycc	0.643	0.650	0.653	0.837	–	**0.845**	0.752	0.839	0.685	–
	3cyr	0.767	0.772	0.789	0.908	–	**0.911**	0.797	0.898	0.797	–
	1ad2	0.773	0.821	0.845	0.917	–	0.956	**0.959**	0.950	0.941	–
	1ldg	0.880	0.895	0.922	**0.989**	–	0.963	0.914	0.946	0.903	–
	1fieA	0.932	0.843	0.942	0.947	–	**0.963**	0.926	0.960	0.927	–
	1sesA	0.913	0.620	0.913	0.954	–	**0.982**	0.917	0.962	0.923	–
	1krn	0.895	0.908	0.895	**0.993**	–	0.960	0.942	0.960	0.892	–
	2fxb	**0.985**	0.941	**0.985**	0.951	–	0.970	0.978	0.978	0.978	–
	1amk	0.945	0.965	0.959	0.997	–	**0.998**	0.982	0.984	0.982	–
	1ar5A	0.946	0.812	0.946	0.971	–	**0.974**	0.942	0.968	0.954	–
	1gpb	0.947	0.868	0.948	0.982	–	0.983	0.976	**0.984**	0.983	–
	1taq	0.826	0.525	0.826	0.931	–	0.945	0.938	**0.959**	0.944	–
Ref. 2	2pia	0.766	0.761	0.768	0.763	0.730	0.826	0.847	0.850	0.839	**0.879**
	1pamA	0.757	0.755	0.758	0.623	0.66	0.859	0.857	0.863	0.853	**0.913**
	1aboA	0.65	–	–	0.489	**0.812**	0.796	0.723	0.791	0.679	–
	1idy	0.515	–	–	0.548	**0.997**	0.989	0.981	0.992	0.992	–
	1csy	0.154	–	–	0.154	0.735	0.764	0.731	**0.885**	0.831	–
	1r69	0.675	–	–	0.475	0.90	**0.965**	0.859	0.934	0.874	–
	1tvxA	0.552	–	–	0.448	0.891	0.920	0.944	**0.974**	0.944	–
	1tgxA	0.727	–	–	0.773	0.835	**0.878**	0.867	**0.878**	0.850	–
	1ubi	0.482	–	–	0.492	0.795	0.767	0.732	0.778	0.794	**0.911**
	1wit	0.557	–	–	0.694	0.825	0.851	0.875	0.815	0.774	**0.917**
	2trx	0.870	–	–	0.870	0.982	**0.986**	0.959	**0.986**	**0.986**	–
	1sbp	0.217	–	–	0.374	0.778	0.765	**0.782**	0.772	0.778	–
	1havA	0.480	–	–	0.448	0.792	0.879	**0.884**	0.846	**0.884**	–
	1uky	0.656	–	–	0.476	0.625	0.808	0.845	**0.891**	0.872	–
	2hsdA	0.484	–	–	0.498	0.745	0.796	**0.856**	0.829	0.742	0.855
	3grs	0.192	–	–	0.282	0.755	0.746	0.717	0.751	0.781	**0.864**
	Kinase	0.848	–	–	0.867	0.712	0.799	0.825	**0.888**	0.812	–
	1ajsA	0.324	–	–	0.311	0.892	0.899	**0.906**	0.905	0.902	–
	1cpt	0.660	–	–	0.776	0.584	**0.875**	0.869	0.812	0.853	–
	1lvl	0.746	–	–	0.726	0.567	0.781	0.803	0.819	0.816	**0.825**
	1ped	0.834	–	–	0.835	0.78	0.912	0.935	**0.947**	0.943	–
	2myr	**0.904**	–	–	0.825	0.675	0.822	0.806	0.830	0.808	–
	4enl	0.375	–	–	0.739	0.812	0.896	0.890	0.889	0.899	**0.912**
Ref. 3	Kinase	0.619	0.58	0.619	0.758	0.697	0.825	0.870	0.890	0.887	**0.918**
	1pamA	0.743	0.703	0.744	0.579	0.525	**0.835**	0.853	0.788	0.792	–
	1idy	0.273	–	–	0.364	0.546	**0.601**	0.446	0.599	0.569	–
	1r69	0.524	–	–	0.524	0.374	0.709	0.724	0.733	**0.765**	–
	1ubi	0.146	–	–	0.585	0.31	0.386	0.398	0.414	0.410	**0.590**
	1wit	0.565	–	–	0.484	0.780	0.758	0.833	**0.873**	0.867	–
	1uky	0.130	–	–	0.269	0.35	0.468	0.469	0.481	0.526	**0.673**
	1ajsA	0.163	–	–	0.186	0.18	0.311	0.383	0.453	0.408	**0.586**
	1ped	0.627	–	–	0.646	0.425	0.775	0.848	**0.893**	0.783	–
	2myr	0.538	–	–	0.494	0.33	**0.813**	0.586	0.651	0.519	–
	4enl	0.547	–	–	0.672	0.680	0.800	0.836	**0.866**	**0.866**	0.862
Ref. 4	1dynA	0.000	**0.038**	0.034	–	–	0.033	0.029	0.033	0.031	–
	Kinase2	0.630	0.710	0.635	–	–	0.384	0.330	0.542	0.478	**0.865**
Ref. 5	2cba	0.628	0.422	0.621	–	–	**0.852**	0.839	0.835	0.846	–
	S51	0.75	0.528	0.730	–	–	**0.835**	0.650	0.743	0.756	–

8.1.2.1. Gap insertion. In this operation, a sequence from the set of sequences considered in the alignment is chosen randomly and a block of gaps of variable size is inserted at a random position of that sequence [3,14,22,61] as shown in Fig. 7a. The gap block size is also chosen randomly. In another alternate way [2,66], the sequences in the alignment are split horizontally into two groups where the first group contains few numbers of sequences and the other contains the remaining number of sequences. For each of the two groups, a gap of randomly chosen length is inserted in all the sequences of that group at the same randomly chosen position (Fig. 7b).

8.1.2.2. Gap deletion. This operator selects a sequence randomly from the alignment and a block of gaps. It then completely deletes the gap block from that sequence [3,22] as shown in Fig. 8. If it does not find any gaps then the selected sequence remains unchanged.

8.1.2.3. Gap increment and decrement. The functions of these two operators are completely opposite to each other. These two operators first choose an existing block of gaps of variable length from a randomly selected sequence. After that, the gap increment operator increases the gap size by adding an extra gap in that block (Fig. 9a). On the other hand, the gap decrement operator decreases the size of the gap by removing a gap [3,22,36,61] (Fig. 9b).

8.1.2.4. Gap merging. This operator randomly chooses two discretely placed gaps from a randomly selected sequence and merge them together [14,46,66] (Fig. 10). Lee et al. [46] used another variant of it that selects all the separated gaps that are closed to the randomly chosen gap and merges them to the selected gap position.

8.1.2.5. Gap splitting. This operator was used in SAGA [66]. It selects a block of gaps, splits the length of gaps vertically into two, and moves one-half either to the left or to the right (Fig. 11a). The splitting is also done horizontally by splitting the alignment into two where the first group contains a number of sequences and the other contains the remaining number of sequences. It then moves a gap block of one of the groups either to the left or to the right (Fig. 11b).

8.1.2.6. Gap shuffling. Shuffling of gaps defines the changing of positions of gaps randomly in an alignment. In a sequence chosen at random, this operator selects a gap block, and then a position is selected randomly in that sequence. The gap block is then shifted to the later randomly selected position (Fig. 12). The gap block length may vary from just one gap to more than one gap. The newly selected gap position may be left or right from the previous gap position [1–3,14,36,46,66,69]. Instead of choosing gap position, a variant of this operator chooses a residue and shifts it to a new position [2,46,66].

In the guide tree approach as defined by Naznin et al. [62,63], the mutation operation is performed by randomly shuffling two sequence positions with each other within a guide tree and thus construct a new MSA based on the modified guide tree.

9. Benchmarking

To evaluate different alignment tools effectively, there is a need to have some benchmark sequences and "gold standard" reference alignments of those sequences. For assessing the performance of different automated alignment tools, it is required to find a quantitative accuracy relative to gold standard reference alignments. Popular benchmark databases of hand-curated reference alignments include BAliBASE [94] and HOMSTRAD [54]. A number of modern alignment databases rely on automated structural alignment protocols, including SABmark [96], PREFAB [18], OxBench [74], and BAliBASE version 3 [92]. Among them, the most widely applied database is BAliBASE (http://www.lbgi.fr/balibase/) that contains a C-language application, called *bali_score*, which measure the *SP* (Sum of Pairs) and the *TC* (Total Column) score of test alignment in comparison to the reference alignment. If the test alignment does not match at all with the reference sequences the bali_score produces 0.0 score, if the test alignment is identical with reference alignment then it gives 1.0 score. If it matches with some parts with the reference alignment, then the score is in between 0.0 and 1.0. To judge the performance, most of the alignment tools used version 2 of BAliBASE. It contains eight reference sets. Reference 1 contains a number of equidistant sequences. The highly diverged sequences are available in Reference 2. Reference 3 consists of groups of sequences with less than 25% identity. Reference 4 contains N/C terminal extensions, and Reference 5 contains internal insertions and deletions. References 6 to 8 contain repeats, circular permutations, and transmembrane proteins respectively. We have shown comparative analytic results for BAliBASE version 2 in Table 1. Here, we have shown the comparison between different well-known and relevant GA-based algorithms along with CLUSTALW, which is the most frequently used non-GA tool. The values of different tools are collected from the literatures [62,63,69]. Bold-faced figure signifies the best score among others for each dataset. A blank cell indicates the unavailability of the result.

From the above result, it is observed that the GA-based tools perform better than ClustalW. Due to unequal numbers of scoring data, we cannot draw a conclusion by considering only one tool superior to others, but it is clearly observed that for reference 1, almost every GA-based algorithm performed well. On the other hand, for other references where the sequences are not equidistance to each other, MO-SAStrE performed better than other tools.

10. Conclusion

MSA is an NP-complete problem and therefore, achieving the global optimum alignment by exact methods is not possible except a very unrealistic small number of short length sequences. Many researches focused on heuristic techniques to solve the MSA problem where the stochastic methods are very promising approaches among them. GA is one of the stochastic methods, which is very popular for complex optimization problems. Many researchers had solved the MSA problem using GA as an optimizer. MSA cannot be solved optimally by considering only one objective like maximizing the scores using any scoring approach. Many other parameters are required to be satisfied like the alignment length constraint, minimum gap length, maximize the number of conserved columns along with the maximum scoring. Therefore, recently researchers are focusing on MSA as a multi-objective optimization problem. It can, therefore, be concluded that if the multiobjective parameters for MSA are chosen correctly, then multi-objective criteria along with the global searching technique like GA would be a better alternative to enhance the quality of MSA.

Acknowledgements

The authors would like to sincerely thank the reviewers for their helpful and constructive suggestions and comments to improve the quality of the paper. The authors would also like to thank Dr. Dhananjay Bhattacharyya of Computational Science Division, Saha Institute of Nuclear Physics, for his helpful suggestion while preparing the paper.

References

[1] P. Agarwal, R. Chauhan, Alignment of multiple sequences using GA method, Int. J. Emerg. Technol. Comp. Appl. Sci. 4 (2013) 411–421.
[2] L.A. Anbarasu, P. Narayanasamy, V. Sundararajan, Multiple molecular sequence alignment by island parallel genetic algorithm, Curr. Sci. 78 (2000) 858–863.
[3] D.E. Arenas, H. Ochoterena, V.K. Rodriguez, Multiple sequence alignment using a genetic algorithm and GLOCSA, J. Artif. Evol. Appl. 2009 (2009) 963150.
[4] T.L. Bailey, C. Elkan, Fitting a mixture model by expectation maximization to discover motifs in biopolymers, Proc. of the Second Int. Conf. on Intelligent Sys. for Mol. Biol, AAAI Press, Menlo Park, Calif 1994, pp. 28–36.
[5] H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, P.E. Bourne, The protein data bank, Nucleic Acids Res. 28 (2000) 235–242.

[6] M. Botta, G. Negro, Multiple sequence alignment with genetic algorithms, in: F. Masulli, L.E. Peterson, R. Tagliaferri (Eds.), Computational Intelligence Methods for Bioinformatics and Biostatistics (CIBB 2009), Lecture Notes in Computer Science, vol. 6160, Springer, Berlin, Heidelberg 2010, pp. 206–214.
[7] H. Carrillo, D. Lipman, The multiple sequence alignment problem in biology, SIAM J. Appl. Math. 48 (1988) 1073–1082.
[8] S.B. Carroll, J.K. Grenier, S.D. Weatherbee, From DNA to Diversity: Molecular Genetics and the Evolutionary of Animal Designs, Blackwell Science, Malden, MA, 2001.
[9] L. Chen, L. Zou, J. Chen, An efficient ant colony algorithm for multiple sequences alignment, Proc. of the Third Int. Conf. on Natural Computation (ICNC 07) 2007, pp. 208–212.
[10] S.M. Chen, C.H. Lin, Multiple DNA sequence alignment based on genetic simulated annealing techniques, Int. J. Inf. Manag. Sci. 18 (2007) 97–111.
[11] Y. Chen, Y. Pan, L. Chen, J. Chen, Partitioned optimization algorithms for multiple sequence alignment, Proc. of the 20th International Conference on Advanced Information Networking and Applications (AINA'06), vol. 2, IEEE Computer Society Press, Washington DC 2006, pp. 618–622.
[12] B.D. Chuong, K. Kazutaka, Protein Multiple Sequence Alignment, Methods Mol. Biol. 484 (2008) 379–413.
[13] F. Corpet, Multiple sequence alignment with hierarchical clustering, Nucleic Acids Res. 16 (1988) 10881–10890.
[14] F.J. Da Silva, J.M. Pérez, J.A. Pulido, M.A. Rodríguez, Parallel niche pareto AlineaGA–an evolutionary multiobjective approach on multiple sequence alignment, J. Integr. Bioinform. 8 (2011) 174.
[15] M.O. Dayhoff, R.M. Schwartz, B.C. Orcutt, A model of evolutionary change in proteins, in: M.O. Dayhoff (Ed.) Atlas of Prot. Seq. and Struct., vol. 5, National Biomedical Research Foundation, Washington, DC 1978, pp. 345–352.
[16] C.B. Do, M.S. Mahabhashyam, M. Brudno, S. Batzoglou, ProbCons: probabilistic consistency-based multiple sequence alignment, Genome Res. 15 (2005) 330–340.
[17] R. Durbin, S. Eddy, A. Krogh, G. Mitchison, Biological Sequence Analysis, Cambridge University Press, 1998.
[18] R.C. Edgar, MUSCLE: multiple sequence alignment with high accuracy and high throughput, Nucleic Acids Res. 32 (2004) 1792–1797.
[19] M. Ehrgott, Multicriteria Optimization, Springer, Berlin, 2005.
[20] D.F. Feng, R.F. Doolittle, Progressive sequence alignment as a prerequisite to correct phylogenetic trees, J. Mol. Evol. 25 (1987) 351–360.
[21] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, 1989.
[22] C. Gondro, B.P. Kinghorn, A simple genetic algorithm for multiple sequence alignment, Genet. Mol. Res. 6 (2007) 964–982.
[23] O. Gotoh, Consistency of optimal sequence alignments, Bull. Math. Biol. 52 (1990) 509–525.
[24] O. Gotoh, Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments, J. Mol. Biol. 264 (1996) 823–838.
[25] D. Graur, W.H. Li, Fundamental of Molecular Evolution, 2nd ed. Sinauer Associates, Sunderland, MA, 2002.
[26] S. Henikoff, J.G. Henikoff, Amino acid substitution matrices from protein blocks, Proc. Natl. Acad. Sci. 89 (1992) 10915–10919.
[27] J. Heringa, W.R. Taylor, Three-dimensional domain duplication, swapping and stealing, Curr. Opin. Struct. Biol. 7 (1997) 416–421.
[28] D.G. Higgins, P.M. Sharp, CLUSTAL: a package for performing multiple sequence alignment on a microcomputer, Gene 73 (1988) 237–244.
[29] P. Hogeweg, B. Hesper, The alignment of sets of sequences and the construction of phylogenetic trees: an integrated method, J. Mol. Evol. 20 (1984) 175–186.
[30] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, 1975.
[31] H. Hongwei, V. Stojkovic, A simulated annealing algorithm for multiple sequence alignment with guaranteed accuracy, Third International Conference on Natural Computation (ICNC-2007), vol. 2, 2007, pp. 270–274.
[32] J. Horn, N. Nafpliotis, D.E. Goldberg, A niched pareto genetic algorithm for multiobjective optimization, First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, vol. 1, Orlando, FL 1994, pp. 82–87.
[33] X. Huang, On global sequence alignment, Comput. Appl. Biosci. 10 (1994) 227–235.
[34] K. Katoh, K. Kuma, H. Toh, T. Miyata, MAFFT version 5: improvement in accuracy of multiple sequence alignment, Nucleic Acids Res. 33 (2005) 511–518.
[35] K. Katoh, K. Misawa, K. Kuma, T. Miyata, MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform, Nucleic Acids Res. 30 (2002) 3059–3066.
[36] M. Kaya, A. Sarhan, R. Alhajj, Multiple sequence alignment with affine gap by using multi-objective genetic algorithm, Comput. Methods Prog. Biomed. 114 (2014) 38–49.
[37] C. Kemena, J.F. Taly, J. Kleinjung, C. Notredame, STRIKE: evaluation of protein MSAs using a single 3D structure, Bioinformatics 27 (2011) 3385–3391.
[38] J. Kim, S. Pramanik, M.J. Chung, Multiple sequence alignment using simulated annealing, Comput. Appl. Biosci. 10 (1994) 419–426.
[39] A.S. Konagurthu, P.J. Stuckey, Optimal sum-of-pairs multiple sequence alignment using incremental Carrillo and Lipman bounds, J. Comput. Biol. 13 (2006) 668–685.
[40] S. Kumar, A. Filipski, Multiple sequence alignment: in pursuit of homologous DNA positions, Genome Res. 17 (2007) 127–135.
[41] N.J. Lakshmi, P. Gavarraju, J.K. Jeevana, P. Karteeka, A literature survey on multiple sequence alignment algorithms, Int. J. Adv. Res. Comput. Sci. Softw. Eng. 6 (2016) 280–288.
[42] B. Langmead, S. Salzberg, Fast gapped-read alignment with Bowtie 2, Nat. Methods 9 (2012) 357–359.
[43] T. Lassmann, E. Sonnhammer, Quality assessment of multiple alignment programs, FEBS Lett. 529 (2002) 126–130.
[44] T. Lassmann, E.L. Sonnhammer, Kalign–an accurate and fast multiple sequence alignment algorithm, BMC Bioinf. 6 (2005) 298.
[45] C.E. Lawrence, S.F. Altschul, M.S. Boguski, J.S. Liu, A.F. Neuwald, J.C. Wootton, Detecting subtle sequence signals: a Gibbs sampling strategy for multiple sequence alignment, Science 262 (1993) 208–214.
[46] Z.H. Lee, S.F. Su, C.C. Chuang, K.H. Liu, Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment, Appl. Soft Comput. 8 (2008) 55–78.
[47] X. Lei, J. Sun, X. Xu, L. Guo, Artificial bee colony algorithm for solving multiple sequence alignment, IEEE Fifth Int. Conf. on Bio-Inspired Computing: Theories and Applications (BIC-TA), Changsha, IEEE, China 2010, pp. 337–342.
[48] H. Li, R. Durbin, Fast and accurate long-read alignment with Burrows-Wheeler transform, Bioinformatics 26 (2010) 589–595.
[49] D.J. Lipman, S.F. Altschul, J.D. Kececioglu, A tool for multiple sequence alignment, Proc. Natl. Acad. Sci. U. S. A. 86 (1989) 4412–4415.
[50] L. Liu, Y. Li, S. Li, N. Hu, Y. He, R. Pong, D. Lin, L. Lu, M. Law, Comparison of next-generation sequencing systems, J Biomed Biotechnol 2012 (2012) 1–11.
[51] Y. Liu, B. Schmidt, D.L. Maskell, MSAProbs: multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities, Bioinformatics 26 (2010) 1958–1964.
[52] J. Luo, L. Zhang, C. Liang, A multigroup parallel genetic algorithm for multiple sequence alignment, Artif. Intell. Comput. Intell. 7002 (2011) 308–316.
[53] T. Michalewicz, D.B. Fogel, How to Solve It: Modern Heuristics, Springer-Verlag, Heidelberg, 2000.
[54] K. Mizuguchi, C.M. Deane, T.L. Blundell, J.P. Overington, HOMSTRAD: a database of protein structure alignments for homologous families, Protein Sci. 7 (1998) 2469–2471.
[55] B. Morgenstern, K. Frech, A. Dress, T. Werner, DIALIGN: finding local similarities by multiple sequence alignment, Bioinformatics 14 (1998) 290–294.
[56] B. Morgenstern, Dialign 2: improvement of the segment-to-segment approach to multiple sequence alignment, Bioinformatics 15 (1999) 211–218.
[57] B. Morgenstern, S.J. Prohaska, D. Pöhler, P.F. Stadler, Multiple sequence alignment with user-defined anchor points, Algorithms Mol. Biol. 1 (2006) 6.
[58] J.D. Moss, C.G. Johnson, An ant colony algorithm for multiple sequence alignment in bioinformatics, in: D.W. Pearson, N.C. Steele, R.F. Albrecht (Eds.), Artificial Neural Networks and Genetic Algorithms, Springer, Berlin 2003, pp. 182–186.
[59] D.W. Mount, Bioinformatics: Sequence and Genome Analysis, Cold Spring Harbor Laboratory Press, Cold Spring Harbor. NY, 2004.
[60] M. Murata, J.S. Richardson, J.L. Sussman, Simultaneous comparison of three protein sequences, Proc. Natl. Acad. Sci. U. S. A. 82 (1985) 3073–3077.
[61] Z. Narimani, B. Hamid, A. Hassan, A new genetic algorithm for multiple sequence alignment, Int. J. Comput. Intell. Appl. 11 (2012).
[62] F. Naznin, R. Sarker, D. Essam, Progressive alignment method using genetic algorithm for multiple sequence alignment, IEEE Trans. Evol. Comput. 16 (2012) 615–631.
[63] F. Naznin, R. Sarker, D. Essam, Vertical decomposition with genetic algorithm for multiple sequence alignment, BMC Bioinf. 12 (2011) 353.
[64] S.B. Needleman, C.D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, J. Mol. Biol. 48 (1970) 443–453.
[65] C. Notredame, Recent progress in multiple sequence alignment: a survey, Pharmacogenomics 3 (2002) 131–144.
[66] C. Notredame, D.G. Higgins, SAGA: sequence alignment by genetic algorithm, Nucleic Acids Res. 24 (1996) 1515–1524.
[67] C. Notredame, D.G. Higgins, J. Heringa, T-Coffee: a novel method for fast and accurate multiple sequence alignment, J. Mol. Biol. 302 (2000) 205–217.
[68] C. Notredame, L. Holm, D.G. Higgins, COFFEE: an objective function for multiple sequence alignments, Bioinformatics 14 (1998) 407–422.
[69] F.M. Ortuño, O. Valenzuela, F. Rojas, H. Pomares, J.P. Florido, J.M. Urquiza, I. Rojas, Optimizing multiple sequence alignments using a genetic algorithm based on three objectives: structural information, non-gaps percentage and totally conserved columns, Bioinformatics 29 (2013) 2112–2121.
[70] J. Pei, N.V. Grishin, MUMMALS: multiple sequence alignment improved by using hidden Markov models with local structural information, Nucleic Acids Res. 34 (2006) 4364–4374.
[71] J. Pei, N.V. Grishin, PROMALS: towards accurate multiple sequence alignments of distantly related proteins, Bioinformatics 23 (2007) 802–808.
[72] J. Pei, R. Sadreyev, N.V. Grishin, PCMA: fast and accurate multiple sequence alignment based on profile consistency, Bioinformatics 19 (2003) 427–428.
[73] D. Pollard, C. Bergman, J. Stoye, S. Celniker, M. Eisen, Benchmarking tools for the alignment of functional noncoding DNA, BMC Bioinf. 5 (2004) 6.
[74] G.P. Raghava, S.M. Searle, P.C. Audley, J.D. Barber, G.J. Barton, OXBench: a benchmark for evaluation of protein multiple sequence alignment accuracy, BMC Bioinf. 4 (2003) 47.
[75] T. Riaz, W. Yi, K.B. Li, Multiple sequence alignment using tabu search, Proc. of Second Asia-Pacific Bioinformatics Conference (APBC), Dunedin, Australian Computer Society, Inc., New Zealand 2004, pp. 223–232.
[76] T. Riaz, W. Yi, K.B. Li, A tabu search algorithm for post-processing multiple sequence alignment, J. Bioinforma. Comput. Biol. 3 (2005) 145–156.
[77] N. Saitou, M. Nei, The neighbor-joining method: a new method for reconstructing phylogenetic trees, Mol. Biol. Evol. 4 (1987) 406–425.
[78] R.E. Sean, A memory-efficient dynamic programming algorithm for optimal alignment of sequence to an RNA secondary structure, BMC Bioinf. 3 (2002) 13.
[79] C. Shyu, J.A. Foster, Evolving consensus sequence for multiple sequence alignment with a genetic algorithm, in: E. Cantú-Paz, et al., (Eds.) Proc. Conf. Genet. and Evol.

Comp. (GECCO'03), vol. 2724, LNCS, Springer-Verlag, Berlin Heidelberg, Chicago, IL, USA 2003, pp. 2313–2324.
[80] C. Shyu, L. Sheneman, J.A. Foster, Multiple sequence alignment with evolutionary computation, Genet. Program Evolvable Mach. 5 (2004) 121–144.
[81] T.F. Smith, M.S. Waterman, Identification of common molecular sequences, J. Mol. Biol. 147 (1981) 195–197.
[82] R.R. Sokal, C.D. Michener, A statistical method for evaluating systematic relationships, Univ. Kans. Sci. Bull. 28 (1958) 1409–1438.
[83] J. Stoye, Multiple sequence alignment with the divide-and-conquer method, Gene 211 (1998) GC45–GC56.
[84] J. Stoye, V. Moulton, A.W. Dress, DCA: an efficient implementation of the divide-and conquer approach to simultaneous multiple sequence alignment, Comput. Appl. Biosci. 13 (1997) 625–626.
[85] J. Taheri, A.Y. Zomaya, RBT-GA: a novel metaheuristic for solving the multiple sequence alignment problem, BMC Genomics 10 (Suppl. 1) (2009) S10.
[86] W.R. Taylor, Multiple sequence alignment by a pairwise algorithm, Comput. Appl. Biosci. 3 (1987) 81–87.
[87] W.R. Taylor, A flexible method to align large numbers of biological sequences, J. Mol. Evol. 28 (1988) 161–169.
[88] D.W. Thomas, N. Serban, Fast and SNP-tolerant detection of complex variants and splicing in short reads, Bioinformatics 26 (2010) 873–881.
[89] J. Thompson, F. Plewniak, O. Poch, A comprehensive comparison of multiple sequence alignment programs, Nucleic Acids Res. 27 (1999).
[90] J.D. Thompson, D.G. Higgins, T.J. Gibson, Improved sensitivity of profile searched through the use of sequence weights and gap excision, CABIOS 10 (1994) 19–29.
[91] J.D. Thompson, D.G. Higgins, T.J. Gibson, CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice, Nucleic Acids Res. 22 (1994) 4673–4680.
[92] J.D. Thompson, P. Koehl, R. Ripp, O. Poch, BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark, Proteins 61 (2005) 127–136.
[93] J.D. Thompson, B. Linard, D. Lecompte, O. Poch, A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives, PLoS One 6 (2011) 1–14.
[94] J.D. Thompson, F. Plewniak, O. Poch, BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs, Bioinformatics 15 (1999) 87–88.
[95] W.I. Van, I. Lasters, L. Wyns, Align-m – a new algorithm for multiple alignment of highly divergent sequences, Bioinformatics 20 (2004) 1428–1435.
[96] W.I. Van, I. Lasters, L. Wyns, SABmark – a benchmark for sequence alignment that covers the entire known fold space, Bioinformatics 21 (2005) 1267–1268.
[97] M. Vingron, A. Von Haeseler, Towards integration of multiple alignment and phylogenetic tree construction, J. Comput. Biol. 4 (1997) 23–34.
[98] L. Wang, T. Jiang, On the complexity of multiple sequence alignment, J. Comput. Biol. 1 (1994) 337–348.
[99] Y. Wang, K.B. Li, An adaptive and iterative algorithm for refining multiple sequence alignment, Comput. Biol. Chem. 28 (2004) 141–148.
[100] L. Wei, C. Ling, J. Chen, An efficient algorithm for multiple sequence alignment based on ant colony optimisation and divide-and-conquer method, N. Z. J. Agric. Res. 50 (2007) 617–626.
[101] J. Xiong, Essential Bioinformatics, Cambridge University Press, NY, 2006.
[102] X. Xu, X. Lei, Multiple sequence alignment based on ABC_SA, in: F. Wang, H. Deng, Y. Gao, J. Lei (Eds.), Artificial Intelligence and Computational Intelligence, Lecture notes in computer science, vol. 6320, Springer, Berlin 2010, pp. 98–105.
[103] S. Yamada, O. Gotoh, H. Yamana, Improvement in accuracy of multiple sequence alignment using novel group-to-group sequence alignment algorithm with piecewise linear gap cost, BMC Bioinf. 7 (2006) 524.
[104] C. Zhang, A.K. Wong, A genetic algorithm for multiple molecular sequence alignment, Comput. Appl. Biosci. 13 (1997) 565–581.
[105] A. Zhou, B.Y. Qu, H. Li, S.Z. Zhao, P.N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: a survey of the state of the art, Swarm Evol. Comput. 1 (2011) 32–49.