A Hybrid Method Applied to Multiple Sequence Alignment Problem

LAMICHE CHAABANE[1] and MOUSSAOUI ABDELOUAHAB[2]
[1]Department of Computer Science
University of Mohamed Boudiaf, M'sila
Box 166, Ichebilia, M'sila
ALGERIA
lamiche07@gmail.com
[2]Department of Computer Science
Setif1 University
ALGERIA
moussaoui.abdel@gmail.com

Abstract: - Multiple sequence alignment (MSA) is an important tool in biological analysis. However, it is difficult to solve this class of problems due to their exponential time complexity when the number of sequences and their lengths increase. In this research paper, we present a new method for multiple sequence alignment problem, based on classical tabu search (TS) and simulated annealing (SA) techniques. The developed approach is called (TSSA), it is implemented in order to obtain an optimal results of multiple sequence alignment. The essential idea of our TSSA approach is the integration of the metropolis criterion of SA algorithm to construct elite solutions list, which can be exploited by TS algorithm to intensify the search around of best elements of this list. At the same time, it helps to diversify the search when TS restarting with worse solutions from the constructed elite solutions list. Computational results on a wide range of datasets taken from the BaliBASE database have shown the degree to which simulated annealing enhance the performance of tabu search algorithm and demonstrate the superiority of TSSA algorithm over the six well-known multiple sequence alignment methods PRRP, ClustalW, SAGA, DiAlign, ML_PIMA and MultiAlign. The proposed method also finds solutions faster than binary particle swarm optimization (BPSO) algorithm, TS-R algorithm, AIS method and IMSA approach.

Key-Words: - Multiple sequence alignment, tabu search, simulated annealing, TSSA, elite solutions list.

1 Introduction

Multiple sequence alignment (MSA) of nucleic or amino acid sequences continues to play a very central role to the advancement of understanding in molecular biology. Sequence alignments can be used to (i) determine evolutionary distances between organisms and infer phylogenetic relationships, (ii) discover conserved motifs that might be important at the levels of transcription, translation, and/or structure, (iii) improve our understanding and prediction of molecular structures. The size of the MSA problem space increases dramatically with the number of sequences in the alignment and their length. As a result, MSA problems are NP-hard [1].

Several methods were proposed in the literature. The highest scoring alignment can be found through a dynamic programming (DP). The dynamic programming principle contains three (3) major steps: the first step in DP involves the creation of a matrix with *M+1* column and *N+1* row, where *M* and *N* are the lengths of the sequences to be aligned. In the second step, each cell in the created matrix is filled with calculated values using scoring scheme for matches, mismatches and gaps. Finally, the trace back step simply determines the actual alignment that results in a maximum score.

Theoretically, dynamic programming method can be applied to any number of sequences; however, because it is computationally expensive in both time and memory. It is rarely used for more than three or four sequences in its most basic form. Algorithms such as the Needleman-Wunsch [2] and Smith-Waterman [3] are based in this approach.

To circumvent this limitation, most multiple alignment methods implement approximate heuristic algorithms. Currently, the main approach to multiple sequence alignment is the progressive method of Feng and Doolittle [4] implemented in ClustalW [5], MULTAL [6] and T-COFFEE [7] for instance. This method is very fast and its major disadvantage is the problem of the local minima and consequently it can lead to poor quality solutions.

The iteration-based approach is also applied to the multiple sequence alignment. This method uses algorithms that produce an alignment and tries to improve it over successive iterations. This approach includes hidden markov models [8][9], simulated annealing [10][11], Tabu Search [12], Ant Colony Algorithm [13], Genetic Algorithms [14][15][16], among others. The disadvantage is that metaheuristics do not guarantee optimal solutions, but solutions generated can be very close to optimal solution in a reasonable processing time.

In this research study, we develop a novel hybrid model called (TSSA) which combines tabu search and metropolis criterion of simulated annealing procedure. In addition to, it includes a set of techniques to generate the neighborhood structure, an efficient intensification/diversification phase based on SA algorithm to improve solution quality and a restarting search as a second diversification strategy to explore other promising regions in the research space of alignments.

The remaining of the paper is organized as follows: section 2 presents a brief review of the researches related to the proposed framework. In section 3, preliminaries concepts of both simulated annealing and tabu search algorithms are described. Our TSSA algorithm is detailed in section 4. In section 5, the simulation results are provided and discussed. Finally, some concluding remarks are given in section 6.

2 Related Works

A brief review of some related works in the multiple sequence alignment field using iterative methods such as tabu search is presented in this section. Riaz et *al*. [12] presented a tabu search algorithm to align multiple sequences. The framework of his work consists to implement the adaptive memory features typical of tabu search in order to obtain multiple sequences alignment. Two called aligned and unaligned initial solutions are used as starting points for this version algorithm. Aligned initial solutions are generated using Feng and Doolittle's progressive alignment algorithm [4]. Unaligned initial solutions are constructed by inserting a fixed number of gaps into different sequences at regular intervals. The quality of an alignment is measured by the COFFEE objective function [17]. In order to move from one solution to other, the algorithm moves gaps around within a single sequence and performs block moves. This tabu search uses a recency-based memory structure. Thus, after gaps are moved, the tabu list is updated to avoid cycling and getting trapped in a local solution.

In Ref. [18], C. Lightner proposed a several tabu searches which can progressively align sequences. Tabu A is the first tabu search version. An MSA, the basic idea of the Tabu A algorithm was determined by initially aligning the first two sequences using dynamic programming (DP) technique. Then, sequences were added one by one until the entire MSA was obtained. Whenever the tabu search moved to a new solution, the entire MSA was determined using DP and progressively adding on each sequence.

A modified version of Tabu A is Tabu B, that reduced the number of computation required to generate an MSA. For the initial solution, Tabu A divided the MSA up into subgroups. After each subgroup was aligned separately, all the subgroups were aligned together to form an MSA.

One drawback to Tabu A and B was that sequences were progressively added to the alignment using only the pairwise DP procedure. Thus, Tabu A' was implemented to improve the overall SP score of Tabu A. An MSA for Tabu A' was determined by initially aligning the first three sequences using DP. Subgroups of three sequences were each aligned separately. Then, all subgroups with three sequences were progressively aligned together to form an MSA.

Tabu search C is the third proposed version. It used and improved upon the best features from Tabu A, Tabu B and Tabu A', A guide tree, intensification and diversification procedures were new components incorporated into Tabu C. The intensification phase iteratively refines the best MSA, so that gaps introduced early in the alignment can be removed or switched around in the diversification phase, a new MSA is generated that is not in the neighborhood of the current solution.

In Ref. [19], the authors described a simple genetic algorithm (GA) for the MSA of biological sequences. In that, two methods are used for creating initial candidate solutions: 1) from pairwise alignments, where sequences are aligned in pairs alone, or 2) from a combination of pairwise alignments and a user-defined multiple sequence alignment. Produced results by the two approaches in MSA-GA are compared with the alignments generated by ClustalW. The authors in [20] proposed an algorithm based on PSO algorithm to address the multiple sequence alignment problem. Simulation results using SP score measure and nine BaliBASE tests case showed that the proposed PSOMSA algorithms has superior performance when compared to Clustal X program.

V. Cutello et al. [21] presented an immune inspired algorithm (IMSA) to tackle the multiple sequence alignment problem. This algorithm includes a new method to generate the initial population (CLUSTALW-seeding) and two specific ad-hoc mutation operators. Experimental results on BALIBASE v.1.0 show that IMSA is superior to PRRP, CLUSTALX, SAGA, DIALIGN, PIMA, MULTIALIGN and PILEUP8; while on BALIBASE v.2.0 the algorithm shows interesting results in terms of SP score.

In Ref. [22], the authors proposed a pattern mining approach for solving MSA problem. In their work, experimental evaluation involving 108 protein families demonstrates situations where their approach outperforms a set of the state-of-the-art MSA algorithms. A Hybrid algorithm using a GA and cuckoo search algorithm to improve multiple sequence alignment is presented in [23]. The obtained results are compared with ClustalW by using five different datasets. Recently, an efficient method by using multi-objective genetic algorithm (MSAGMOGA) to discover optimal alignments is proposed in [24]. Experiments on the BAliBASE 2.0 database confirmed that MSAGMOGA obtained better results than MUSCLE, SAGA and MSA-GA methods.

3 Preliminaries

In this section, we briefly review basic concepts of both simulated annealing algorithm and tabu search algorithm which are used in our proposed approach to solve MSA problem.

3.1 Overview of Simulated Annealing

Simulated annealing is introduced by Kirkpatrick et al. [25]. The basic concept of simulated annealing algorithms is from observing the change of energy in the process in which materials solidify from the liquid state to the solid state. When the system's temperature decreases gradually in the annealing process, supposed the energy of the new state is lower than the energy of the current state, the system will accept the new state and replace it with the current state. Otherwise, the probability of the new state to be accepted is decided by the following formula:

$$P = e^{\frac{-\Delta E}{KT}} \tag{1}$$

where p denotes the probability that the system accepts the new state, T is the current system temperature, k is the Boltzmann constant, ΔE is the difference between the energy of the new state and the energy of the current state. In the annealing process, a lower energy indicates a more stable state and it indicates a "better" state. When the system starts to anneal, an annealing schedule must be decided first, which contains the initial temperature, the frozen temperature, and the way the temperature decreases.

When the system's temperature decreases, a new state will be randomly generated by the system and the energy of the new state will be calculated. The energy of the new state will be compared with the energy of the current state and we can get the difference ΔE. If ΔE is smaller than or equal to zero, then the new state will be accepted by the system. Otherwise, the system will accept the new state according to the probability p calculated by formula expressed in equation (1).

We can see that when the system's temperature decreases gradually, the probability p decreases to zero, and the system finally inclines to the most stable state, i.e., the optimal solution.

3.2 Overview of Tabu Search

Tabu search (TS), initially suggested by Glover and Laguna [26], it is an iterative improvement approach designed for obtaining (near) global optimum solutions to combinatorial optimization problems.

The idea of TS can be described briefly as follows: starting from an initial solution, TS iteratively moves from the current solution X to its best improved solution Y in the neighborhood of X, or, if one does not exist, chooses the least worsening solution, until a superimposed stopping criterion becomes true. In order to avoid cycling to some extent, moves which would bring us back to a recently visited solution should be forbidden or declared tabu for a certain number of iterations.

This is accomplished by keeping the attributes of the forbidden moves in a list, called a tabu list. The size of the tabu list must be large enough to prevent cycling, but small enough not to forbid too many moves. The process is terminated if the number of iteration reaches a fixed number of iterations I_{max} or after some number of consecutive iterations without an improvement in the objective function value [27].

4 Proposed Method

Such as cited above, Tabu search (TS) and simulated annealing (SA) are two very effective meta-heuristic methods for many combinatorial optimization problems. They are both based on the local search algorithm since they start from an initial solution and use a generation mechanism to perturb from the current solution to a new one in an iterative manner. Unlike the local search, in order to avoid being trapped in local optima, TS and SA employ different distinctive rules [28].

However, SA uses the metropolis criterion to exploit the search space. Whereas, TS takes advantage of the history records that are collected during the search process, so both TS and SA may climb out of local optima.

In this study, after a series of experiments with unsatisfactory results while employing pure SA and pure TS methods, we decided to use a hybrid tabu search/simulated annealing algorithm to solve the MSA problem. In this hybrid model called TSSA algorithm, SA is used to find promising elite solutions in the search history and TS intensifies the search around those elite solutions. The main idea of the developed algorithm is a simple implementation of TS intensification strategy. By means of the metropolis criterion of SA procedure, sufficiently “good” solutions inside the solution space are determined during the operation of TS algorithm.

The “good” solutions found are stored in a bounded length elite solutions list. Each new “good” solution is added on the top of elite solutions list when it is discovered. The current first solution of the elite list is always chosen (and removed) as a starting solution to resume the TS algorithm. With each new and different starting solution, the algorithm is performed for a pre-specified number of iterations. The elite solutions list is always updated with the new “good” solutions encountered during the search. The old “good” solutions can be deleted from the list to keep the length stable. Given a suitable temperature (*T*) to be used in metropolis criterion, elite solutions in the list should be prevented from being exhausted.

The algorithm terminates when the total number of iterations reaches to a given value or the optimal objective value is attained in case it is known. As any meta-heuristic approach, in order to implement the TSSA algorithm, a set of decisions should be chosen. For the considered problem all components of our method are clarified below:

4.1 Objective Function

Each multiple sequence alignment algorithm has its own objective function for the alignment of sequences. To be used in sequence alignment, an objective function should be explicitly defined as a measure of overall alignment quality.

In our study, COFFEE function [17] is used as an objective function. It works by first generating the pairwise library of the sequences in the alignment and then it calculates the level of identity between the current multiple alignment and the pairwise library. The global score measuring the quality of the alignment is computed by the following formula.

$$f(A) = \frac{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} W_{ij} * Score(A_{ij})}{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} W_{ij} * Len} \qquad (2)$$

where *N* is the number of sequences; *Len* is the length of the multiple alignment; W_{ij} is the percent identity between the two aligned sequences S_i and S_j; A_{ij} is the pairwise projection of sequences S_i and S_j obtained from the multiple alignment; and *Score*(A_{ij}) is the overall level of identity between A_{ij} and the corresponding pairwise alignment. When COFFEE is used as objective function, an optimal MSA solution is one which achieves the maximum COFFEE score.

4.2 Initial Solution

The generation of an initial solution is an important step towards getting a final improved alignment. A good initial solution can effectively converge faster and hence cut the computational cost. In our developed method, the initial solution is constructed by computing the progressive alignment using an algorithm similar to the one proposed by Feng and Doolittle [4]. The three steps to generate the aligned initial solution are as follows:

(1) Calculate a distance matrix of all *N* (*N*-1)/2 pairwise distances for the *N* input sequences by a global alignment algorithm using Needleman-Wunsch algorithm [2].

(2) Construct a guide tree according to the distance matrix by linking the least distant pairs of sequences followed by successively more distant pairs.

(3) Align each node of the guide tree in the order that it is added to the tree until all sequences have been aligned.

4.3 Neighborhood Generation and Move Selection Mechanisms

Several move mechanisms can be applied to a current alignment to generate a new candidate alignments. Basically, all the move sets are related to change the positions of the gaps ('-') in the sequences. The proposed move mechanisms to generate neighborhood structure are as follows :

1) Shuffle (i, j, k, direction): this operation shuffles the left/right (*direction*) gaps from the gap column (including gap *j*) in the sequence *i* and its left/right (*direction*) *k* consecutive characters.

The parameters *i, j* and *direction* in the move sets rules may be randomly determined. But *k* may be determined by certain distribution function, for example uniformly distribution or inverse function related to the size of *k.* Only experiment can tell which is the best distribution function for *k*.

2) Gap block Move : local changes are facilitated through gap blocks. A gap block is a subsequence consisting of 1 or more consecutive gaps in 1 or more aligned sequences. To make this move, a random gap in a random sequence is picked. Then the gap block is extended vertically through the other sequences containing a gap at that position. Afterwards, the gap block is extended horizontally to both sides if all the chosen sequences contain a gap there. Finally, the gap block is moved to a randomly chosen new position in the alignment [29]. The procedure of this mechanism is illustrated in Fig.1 below :

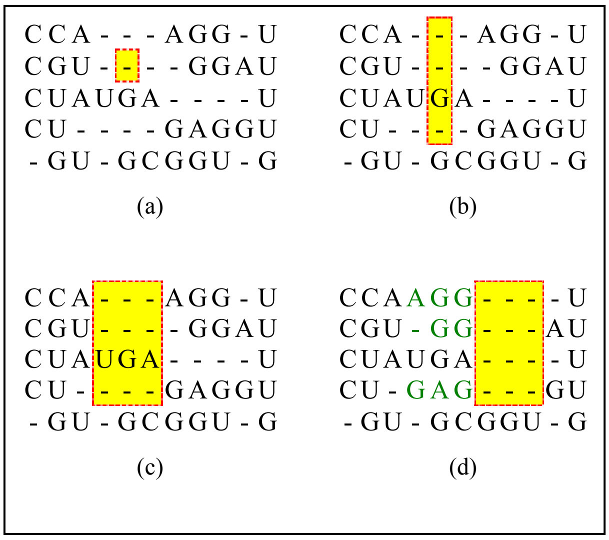

Fig. 1. Gap block move mechanism. (a) choose a gap at random in a sequence, (b) extend vertically, (c) extend horizontally and (d) move to new position in alignment.

We have to resort to such a level of move generation that helps to complete the alignment process in reasonable time. In a typical arrangement, in each iteration the algorithm generates one single sequence move for each sequence by shuffle mechanism and the block moves comprising every possible block of gaps in the alignment. The moves are generated is a stochastic fashion. In case of single sequence moves, the patch of gap(s) as well as the its new location in the sequence are determined randomly. For block moves, every rectangular block of gaps is moved to some random location, either left or right to its current position.

4.4 Tabu List

The size of the tabu list (*TLL: tabu list length*) is one of the important features of tabu search. It is its ability to avoid being trapped in local optima. The size of a tabu list can affect the search performance. Although a longer list may prevent cycling, it requires more scanning and may limit the search domain.

To avoid this problem, in our proposed approach we chose a varying tabu list size during the search as follows: starting by an initial TLL, if there is no improvement of the current solution after a certain number of consecutive iterations, the length is increased in order to insert other possible solutions from the search space. Whereas this length is decreased to allow an intensified searches. This size may be modified to $TLL+X$ or $TLL-X$ during the search process and X is fixed experimentally.

4.5 Aspiration Criteria

Is another important element of tabu search algorithm arises when the move under consideration has been found to the associated with each entry in the tabu list, even when there is no danger of cycling, or it may lead to an overall stagnation of the search process. It is thus necessary to use algorithmic devices that will allow one to cancel tabus. These are called aspiration criteria. In our approach we choose the simplest and most commonly used aspiration criterion, allows a tabu move when it results in a solution with an objective value better than that of the current best-known solution (since the new solution has obviously not been previously visited). The main goal of this idea is to reduce the probability of missing good solutions.

4.6 Intensification and Diversification Strategies

Generally, an intensification procedure revisits and examines the portions of the search space that seem "promising" in order to make sure that the best solutions in these areas are found.

Our TSSA algorithm has both diversification and intensification effect on the search. The use of the metropolis criterion in determining the elite solutions allows the selection of less attractive solutions in terms of the objective value. This means, at any TS iteration, some solutions worse than the best solution in the neighborhood can be selected to be stored in the elite list. Thus, restarting the algorithm with these different solutions guides to diversify the search to another unexplored regions.

At the same time, it helps to intensify the search around those sufficiently “good” solutions. In this case, in order to find other best solutions the neighborhood search space size is doubled (*2*Ns*) and the tabu list length can be reduced progressively to enhance the search in these promising regions of the search space.

We can note here, that the temperature *T* of SA has an important effect on the selected elite solutions and consequently on the solution quality obtained from the TSSA algorithm. If the temperature is too low, the algorithm may be terminated earlier due to the elite solutions being quickly exhausted, whereas if the temperature is too high, it can not guarantee that the “effective” elite solutions are selected [28]. It can be updated by the formula $T=T_0/f(A_{BEST})$, where T_0 is a fixed parameter given at the start of the algorithm and the elite solutions list size is taken as 50.

4.7 Termination Criteria

The last element necessary for tabu search is termination criteria. In our study, the search can be stopped when certain number of iterations *ITMAX* is completed.

The pseudo–code of our developed TSSA algorithm is given below :

Pseudo-code of TSSA Algorithm

```
Begin
Step 0: Generate an initial aligned alignment $A_{INIT}$.
        Initialize $ITMAX, Ns, TLL, T_0, X, m1, m2, m3$
Step 1: Set the current alignment $A_{CURR} = A_{INIT}$ and the
        best alignment $A_{BEST} = A_{INIT}$.
        Set $Init\_Neigh\_Size := Ns$
Step 2: Iterate the following steps for $ITMAX$ iterations
    Step 2.1: Generate $Ns$ neighboring alignments $A_i$
        ($i=1,2, ..., Ns$) of the current solution $A_{CURR}$
    Step 2.2:
        Chose A’ such that :
            $A' := \underset{A \in N(A_{CURR})}{Max} f(A)$
        If $A' \notin TabuList$ Or ($A' \in TabuList$) And
        ($f(A') > f(A_{CURR})$) Then
            $A_{CURR} := A'$ and go to step 2.3
        Else
        Try next test alignment. If all test solutions are
        tabu alignments, go to step 2.1.
    Step 2.3:
        Update the temperature T by $T := T_0 / f(A_{BEST})$
        $diff := f(A_{CURR}) - f(A_{BEST})$
        If ($f(A_{CURR}) > f(A_{BEST})$) Or
            ($\exp(diff / T) > random[0.1]$) Then
            Insert $A_{CURR}$ into $EliteSolutionList[\,]$
        End if
        If $f(A_{CURR}) > f(A_{BEST})$ Then
            $A_{BEST} := A_{CURR}$
        End if
        Insert $A_{CURR}$ into $TabuList[\,]$
        Insert $A_{BEST}$ into $BestSolutionList[\,]$
    Step 2.4:
        If (No Improvement after $m1$ iterations) Then
            $TLL=TLL+X$
        Else
            If ($\exists$ consecutive $m2$ Improvements) Then
            $TLL=TLL-X$
            End if
        End if
    Step 2.5:
        If (No Improvement after $m3$ iterations) Then
            Set $A_{CURR} := EliteSolutionList[1]$
            If $A_{CURR}$ is in $BestSolutionList[]$ Then
                $Ns := Init\_Neigh\_Size * 2$
            Else $Ns.=Init\_Neigh\_Size$
            End if
            Go to Step 2
        End if
Step 3: Record the best alignment $A_{BEST}$ and Stop.
End.
```

5 Experimental Results

Our approach is implemented using Java programming language and personnel computer with 2.66 GHz Intel Pentium IV processor. To assess the efficiency and accuracy of our TSSA approach, several experiments were designed using the classical BaliBASE benchmark alignment database [30] [31] that has been developed to evaluate and compare multiple alignment programs containing high-quality (manually refined) multiple sequence alignments, it is divided into two versions: the first version [30] contains 141 reference alignments and it is divided into five hierarchical reference sets containing twelve representative alignments. Moreover, for each alignment the *core blocks* are defined. They are the regions which can be reliably aligned and they represent 58% of residues in the alignments. The remaining 42% are in ambiguous regions which cannot be reliably aligned.

Reference 1 contains alignments of equidistant sequences with similar length. Reference 2 contains alignments of a family (closely related sequences with > 25% identity) and 3 ”orphan” sequences with < 20% identity, reference 3 consists of up to four families with < 25% identity between any two sequences from different families and references 4 and 5 contain sequences with large N/C-terminal extensions or internal insertions.

In the second version [31], all alignments present in the first version have been manually verified and it includes three new reference sets: repeats, circular permutations and transmembrane proteins. It consists of 167 reference alignments with more than 2100 sequences. The three new references contain 26 protein families with 12 distinct repeat types, 8 transmembrane families and 5 families with inverted domains. The measure which is used to evaluate alignment quality is the SP score, it refers the Sum of Pairs score, calculated by the ”*baliscore.c*” program.

Using an extensive set of tests on all the 139 datasets provided by BaliBASE, the average SP score obtained by our approach compared with produced alignment results of both T. Riaz et al.’ tabu search method (TS-R) [12] and other popular multiple sequence alignments techniques including PRRP [32], ClsustalW [33], SAGA [14], DiAlign [34], ML_PIMA [35] and MultiAlign [36] are summarized in table 1 below :

Table 1. Comparative results using BaliBase score measurement.

Aligner	Ref1 (81)	Ref2 (23)	Ref3 (11)	Ref4 (12)	Ref5 (12)	Avg.
PRRP	0.865	0.541	0.532	0.323	0.700	**0.592**
ClustalW	0.841	0.945	0.723	0.821	0.858	**0.838**
SAGA	0.825	0.954	0.777	0.780	0.868	**0.841**
DiAlign	0.767	0.384	0.314	0.853	0.836	**0.631**
ML_PIMA	0.789	0.371	0.372	0.705	0.572	**0.562**
MultiAlign	0.815	0.517	0.303	0.292	0.627	**0.511**
TS-R[12]	0.760	0.889	0.715	0.773	0.905	**0.808**
Our TSSA	0.917	0.960	0.845	0.863	0.923	**0.901**

The results presented in table 1, demonstrate clearly the superior capability and the potent of our developed method to obtain global multiple alignment. However, on all tested problems the SP scores optimized by our approach are higher than the other mentioned methods. The average SP score is comparable to the results generated by both ClustalW and SAGA programs on the reference 2. In the rest of references, the improvement of the average SP values is very significant.

In order to prove the performance of our method, a second experiment using another set of tests is performed where the objective is to compare our developed TSSA method with other published works including IMSA approach [21], AIS approach [37] and BPSO algorithm [20]. The BAliBASE SP score (SPS) results for both small and large datasets taken from BAliBASE database are portrayed in table 2 below:

Table 2. TSSA versus AIS and IMSA approaches: SPS comparative results for the BaliBase test sets.

Instance	N	AIS [37]	IMSA[21]	Our TSSA
laboA	5	0.646	0.759	**0.910**
45lc	5	0.538	0.773	**0.853**
9rnt	5	0.804	**0.954**	0.948
kinase	5	0.399	0.644	**0.716**
2cba	5	**0.761**	**0.754**	0.754
lppn	5	0.623	0.987	**0.989**
2myr	4	**0.385**	0.285	**0.433**
left	4	0.739	0.880	**0.928**
ltaq	5	0.817	0.946	**0.949**
lubi	17	0.393	0.897	**0.907**
kinase	18	0.270	0.905	**0.915**
lidy	27	0.346	**0.854**	0.849
Average		0.560	0.810	**0.846**

Table 3. TSSA versus BPSO approach: SPS comparative results for the BaliBase test sets.

Instance	N	LSEQ(min,max)	BPSO [20]	Our TSSA
lidy	5	(49,58)	0.7394	**0.8492**
45lc	5	(70,87)	0.7973	**0.8532**
lkrn	5	(66,82)	0.9984	0.9988
kinase	5	(263,276)	0.7064	**0.7165**
lpii	5	(247,259)	**0.7987**	0.7980
5ptp	5	(222,245)	0.9328	**0.9410**
lajsA	5	(358,387)	**0.3528**	**0.3750**
glg	5	(438,486)	0.8324	**0.9241**
ltaq	5	(806,928)	0.7633	**0.9491**
Average			0.7690	**0.8227**

N : number of sequences, LSEQ : length of sequences.

From table 2 and 3, it is clear that comparing with AIS, IMSA and BPSO, the average SPS values of TSSA algorithm are greater than or comparable with those obtained by above mentioned techniques for the short, medium and long sequences. In addition to, a significant improvements are observed in test cases with large number of sequences which shows that our approach can work well with large problem size.

6 Conclusion

The goal of this study is to evaluate and validate the efficacy of tabu search algorithm in which metropolis criterion is used as a diversification strategy to solve the multiple sequence alignment problem. Compared to other techniques, all obtained results using a set of BaliBASE benchmarks are encouraged and demonstrate that our developed TSSA performs better that most of the other approaches studied in this work. The technique used to generate an initial solution, efficient mechanisms to neighborhood structure generation and other features such as intensification and diversification strategies are the key of the effectiveness of our approach to improve the alignment quality for test cases that are easy or hard to align.

As a perspective of this work, there are several issues to improve our approach. Firstly, the improvement of the start solution by a specific heuristic is desired. In addition to, we can integrate other mechanisms to neighborhood generation step or incorporate other strategies in intensification/diversification phase to improve the solution quality. Secondly, to overcome the problem due to the evaluation of a huge number of solution combinations when the number of sequences is large, it is better introduce a parallel computing in the developed algorithm. Finally, comparison of the proposed method with other stat-of-the-art techniques using running time is possible to verify its effectiveness .

References:

[1] L.Wang and T. Jiang, On the Complexity of Multiple Sequence Alignment, *Journal of Computational Biology*, Vol.1, 1994, pp. 337-348.

[2] S. B. Needlman and C.D. Wunsch, A General Method Applicable to the Search for Similarities in the Amino-acid Sequence of Two Proteins, *Journal of Molecular Biology*, Vol. 48, 1970, pp. pp. 443-453.

[3] T. F. Smith and M. S. Waterman, Identification of Common Molecular Subsequences. *J. Mol. Biol.*, Vol. 147, 1981, pp. 195-197.

[4] D. F. Feng and R. F. Doolittle, Progressive Sequence Alignment as a Prerequisite to Correct Phylogenetic Trees, *J. Mol. Evol.*, Vol. 25, 1987, pp. 351-360

[5] J.D. Thompson, D.G. Higgins and 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.*, Vol. 22, 1994, pp. 4673-4680

[6] W.R. Taylor, Multiple Sequence Alignment by a Pairwise Algorithm, *Comput. Appl. Biosci.*, 1987, Vol. 3, pp. 81-87.

[7] C. Notredame, D.G. Higgins and J. Heringa, T-Coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment, *J. Mol. Biol.*, Vol. 302, 2000, pp. 205-217.

[8] S.R. Eddy, Profile hidden Markov Models, *Bioinformatics*, Vol. 14, 1998, pp. 755-763.

[9] K. Karplus, and B. Hu, Evaluation of Protein Multiple Alignments by SAM-T99 Using the BAliBASE Multiple Alignment Test Set, *Bioinformatics*, Vol. 17, 2001, pp. 713-720.

[10] J. Kim, S. Pramanik, and M. J. Chung, Multiple Sequence Alignment Using Simulated Annealing, *Computer Applications in the Biosciences*, Vol. 10, 1994, pp. 419-426.

[11] S. M. Chen and C. H. Lin, Multiple DNA Sequence Alignment Based on Genetic Simulated Annealing Techniques, *Int. Jour. of Information and Management Sciences*, Vol. 18, 2007, pp. 97-111.

[12] T. Riaz, Y. Wang and K.B. Li, Multiple Sequence Alignment Using Tabu Search, in *Proceedings of 2nd Asia-Pacific Bioinformatics Conference (APBC)*, Dunedin, New Zealand, 2004, pp. 223-232.

[13] L. Chen, L. Zou, and J. Chen, An Efficient Ant Colony Algorithm for Multiple Sequences alignment, in *Proceedings of the 3rd International Conference on Natural Computation (ICNC '07)*, 2007, pp. 208-212.

[14] C. Notredame and D.G. Higgins, SAGA: Sequence Alignment by Genetic Algorithm, *Nucleic Acids Research*, Vol. 24, 1996, pp.1515-1524.

[15] R. Gupta, P. Agarwal, and A. K. Soni, Genetic Algorithm Based Approach for Obtaining Alignmemt of Multiple Sequences, *Int. Jour. of Adv. Comp. Sci. and Applications*, Vol. 3, 2012, pp. 180-185.

[16] C. Shyu, L. Sheneman, and J.A. Foster, Multiple Sequence Alignment with Evolutionary Computation, *Genet.Prog. Evol. Mach.*, Vol. 5, 2004, pp. 121-144.

[17] C. Notredame, L. Holm, and D. G. Higgins, COFFEE: an Objective Function for Multiple Sequence Alignments, *Bioinformatics*, Vol. 14, 1998, pp. 407-422.

[18] C. Lightner, A Tabu Search Approach to Multiple Sequence Alignment, *Ph.D. dissertation*, North Corolina State University, Raleigh, North Corolina, 2008.

[19] C. Gondro and B.P. Kinghorn, A Simple Genetic Algorithm for Multiple Sequence Alignment, *Genetics and Molecular Research*, Vol. 6, No. 4, 2007, pp. 964-982.

[20] H. X. Long, W. B. Xu, J. Sun, and W. J. Ji, Multiple Sequence Alignment Based on a Binary Particle Swarm Optimization Algorithm, *Proceedings of Fifth International Conference on Natural Computation*, 2009, pp. 265-269.

[21] V. Cutello, G. Nicosia, M. Pavone, and I. Prizzi, Protein Multiple Sequence Alignment by Hybrid Bio-inspired Algorithms, *Nucleic Acids Research*, Vol. 39, No. 6, 2011, pp.1980-1992.

[22] K. S. M. Tozammel Hossain, D. Patnaik, S. Laxman, P. Jain, C. Bailey-Kellogg, and N. Ramakrishnan, Improved Multiple Sequence Alignments using Coupled Pattern Mining, *IEEE/ACM Transaction on Computational Biology and Bioinformatics*, Vol. 6, No. 1, 2013, pp. 1-14.

[23] A. Abu-Srhan and E. Al Daoud, A Hybrid Algorithm Using a Genetic Algorithm and Cuckoo Search Algorithm to Solve the Traveling Salesman Problem and its Application to Multiple Sequence Alignment, *International Journal of Advanced Science and Technology*, Vol. 61, 2013, pp.29-38.

[24] M. Kayaa, A. Sarhanb, R. Alhajjb, Multiple Sequence Alignment with Affine Gap by Using Multi-objective Genetic Algorithm, *Computer Methods and Programs in Biomedicine*, Vol. 114, 2014, pp. 38-49.

[25] S. C. D. kirkpatrick, Jr. C. D. Gelatt, and M. B. Vecchi, Optimization by Simulated Annealing, *Science*, Vol. 220, pp. 671-680, 1983.

[26] F. Glover, M. Laguna, *Tabu Search*, Kluwer Academic Publishers, London, 1997.

[27] K. S. Al-Sultan, A Tabu Search Approach to the Clustering Problem, *Pattern Recognition*, Vol. 1, 1995, pp. 1443-1451.

[28] M. Arikan, and S. Erol, A Hybrid Simulated Annealing-tabu search Algorithm for the Part Selection and Machine Loading Problems in Flexible Manufacturing Systems. *Int. J. Adv. Manuf. Technol*, Vol. 59, 2012, pp. 669-679.

[29] S. Lindgreen1, P. Gardner, and A. Krogh1, MASTR: Multiple Alignment and Structure Prediction of Non-coding RNAs Using Simulated Annealing, *Bioinformatics*, Vol. 23, 2007, pp. 3304-3311.

[30] J.D. Thompson, F. Plewniak, and O. Poch, BAliBASE: A Benchmark Alignment Database for the Evaluation of Multiple Alignment Programs, *Bioinformatics*, Vol. 15, 1999, pp. 87-88.

[31] A. Bahr, J.D. Thompson, J.C. Thierry, and O. Poch, BAliBASE (Benchmark Alignment dataBASE): Enhancements for Repeats, Transmembrane Sequences and Circular Permuations, *Nucleic Acids Research*, Vol. 29, No. 1, 2001, pp. 232-326.

[32] O. 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.*, Vo. 264, No. 4, 1996, pp. 823-38.

[33] J.D. Thompson, D.G. Higgins, and 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.*, Vol. 22, No. 22, 1994, pp. 4673-80.

[34] B. Morgenstern, A. Dress, and T. Werner, Multiple DNA and Protein Sequence Alignment Based on Segment-to-Segment Comparison, *Proc. Natl. Acad. Sci. U S A*, Vol. 93, No. 22, 1996, pp. 12098-103.

[35] R.F. Smith and T.F. Smith, Pattern-induced Multisequence Alignment (PIMA) Algorithm Employing Secondary Structure-dependent Gap Penalties for Use in Comparative Protein Modelling, *Protein Eng.*, Vol. 5, No. 1, 1992, pp. 35-41.

[36] F. Corpet, Multiple Sequence Alignment with Hierarchical clustering, *Nucleic Acids Res.*, Vo. 16, 1988, pp. 10881-90.

[37] H. Ge, W. Zhong, W. Du, F. Qian, and L. Wang, A Hybrid Algorithm Based on Artificial Immune System and Hidden Markov Model for Multiple Sequence Alignment, *International Conference on Intelligent Systems and Knowledge Engineering (ISKE2007)*. Atlantis Press, 2007.