A Method for Multiple Sequence Alignment Based on Particle Swarm Optimization

Fasheng Xu[1] and Yuehui Chen[2]

[1] School of Science,
University of Jinan, Jinan, 250022, P.R. China
[2] School of Information Science and Engineering,
University of Jinan, Jinan, 250022, P.R. China

Abstract. Sequence Alignment is a basic information disposal method in Bioinformatics. However, it is difficult to deal with multiple sequence alignment problem(MSA). In this paper, an improved particle swarm optimization is designed to solve MSA. In the algorithm, each particle represents an alignment and flies to the particle which has the best solution by some rules. Moreover, in order to expand the diversity of the algorithm and enhance the possibility of finding the optimal solution, three operators are designed, that is, gaps deletion, gaps insertion, and local search operator. Simulation results show that for MSA proposed algorithm is superior to Clustal X.

Keywords: Multiple Sequence Alignment; Bioinformatics; Particle Swarm Optimization.

1 Introduction

Multiple alignments of protein sequences are important in many applications, including phylogenetic tree estimation, secondary structure prediction and critical residue identification. Sequence alignment is by far the most common task in bioinformatics. Procedures relying on sequence comparison are diverse and range from database searches to secondary structure prediction . Sequences can be compared two by two to scour databases for homologues, or they can be multiply aligned to visualize the effect of evolution across a whole protein family. However, it is difficult to align multiple sequence. A common heuristic is to seek a multiple alignment that maximizes the SP score (the summed alignment score of each sequence pair), which is NP complete[1]. Therefore the design of algorithms for multiple sequence alignment has been a very active research area.

Many efforts have been made on the problems concerning the optimization of sequence alignment. Needleman and Wunsch [2] presented an algorithm for sequence comparison based on dynamic programming (DP), by which the optimal alignment between two sequences is obtained. The generalization of this algorithm to multiple sequence alignment is not applicable to a practical alignment that consists of dozens or hundreds of sequences, since it requires huge CPU time proportional to N^K, where K is the number of sequences each with length N. Stochastic methods such as Gibbs sampling can be used to search for a maximum objective score [3], but have not been widely adopted. A more popular strategy is the progressive method [4][5], which first estimates a phylogenetic tree. A profile (a multiple alignment treated as a sequence by regarding each column as a symbol) is then constructed for each node in the binary tree. If the node is a leaf, the profile is the corresponding sequence; otherwise its profile is produced by a pair-wise alignment of the profiles of its child nodes . Current progressive algorithms are typically practical for up to a few hundred sequences on desktop computers, the best-known of which is CLUSTALW [6]. A variant of the progressive approach is used by T-Coffee [7], which builds a library of both local and global alignments of every pair of sequences and uses a library-based score for aligning two profiles. On the BAliBASE benchmark [8][9], T-Coffee achieves the best results, but has a high time and space complexity that limits the number of sequences it can align to typically around one hundred. There are also some non-deterministic approaches using genetic algorithms, such as SAGA [10], which was reported to find optimal alignments even in search spaces of a considerable size (more that 10 sequences). The approach is to use a progressive alignment as the initial state of a stochastic search for a maximum objective score (stochastic refine refinement). Alternatively, pairs of profiles can be extracted from the progressive alignment and re-aligned, keeping the results only when an objective score is improved (horizontal refinement)[11].

In this paper, we developed a method for multiple sequence alignment based on the particle swarm optimization(PSO). The organization of this paper is as follows. In Section 2, the sequence alignment problem is introduced. Then, the outline procedure of the improved particle swarm optimization for multiple sequence alignment problem is designed in Section 3. To verify the feasibility and efficiency of the proposed approach, an empirical example is presented in Section 4. Some concluding remarks are given in Section 6.

2 The Sequence Alignment Problem

In bioinformatics, the most important data sets are biological sequences, including DNA sequences and protein sequences. A DNA sequence can be seen as symbols of the ACGT four strings, a protein sequence can be seen as 20 kinds of proteins string symbols. In the process of evolution there can insert, delete or mutate elements of the sequences. Thus, in order to highlight the similarities of the sequences it is often convenient to insert gaps in them, leading to a higher number of symbol matches. The similarity of aligned sequences is measured using a scoring function, which is based on a matrix that assigns a score to every pair of symbols (based on a mutation probability). For proteins, the most commonly used matrices are PAM(Percent Accepted Mutation) and BLOSUM (Blocks Substitution Matrix)[12]. Additionally, a penalization to the insertion of gaps is required in order to avoid the insertion of an excessive number of them. The process of finding an optimum (or at least good) match between the sequences is called sequence alignment.

3 Improved Particle Swarm Optimization for MSA

3.1 Particle Swarm Optimization

Particle Swarm Optimization (PSO) was first introduced by Kennedy and Eberhart [13][14] in 1995 and partly inspired by the behavior of large animal swarms such as schooling fish or flocking birds. PSO conducts search using a population of a random solutions, corresponding to individual. In addition, each potential solution called particles is also assigned a randomized velocity. Each particle in PSO flies in the hyperspace with a velocity which is dynamically adjusted according to the flying experiences of its own and its colleagues. Each particle adjusts its position according to their own and their neighboring-particles experience, moving toward two points: the best position so far by itself called Pbest and by its neighbor called Gbest at every iteration. The particle swarm optimization concept consists of, at each time step, changing the velocity each particle toward its Pbest and Gbest.

Suppose that the search space is D dimensional, then the ith particle of the swarm can be represented by a D dimensional vector $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})'$. The particle velocity can be represented by another D dimensional vector $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})'$. The best previously visited position of the ith particle is denoted as $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})'$. Defining g as the index of the best particle in the swarm, and let the superscripts denote the iteration number, then the position of a particle and its velocity are updated by the following equations :

$$v_{id}^{k+1} = wv_{id}^{k} + c_1 r_1^k (p_{id}^k - x_{id}^k) + c_2 r_2^k (p_{gd}^k - x_{id}^k) \tag{1}$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \tag{2}$$

where $d = 1, 2, \ldots, D$, $i = 1, 2, \ldots, N$, and N is the size of swarm; w is called inertia weight: c_1, c_2 are two positive constants, called cognitive and social parameter respectively; r_1, r_2 are random numbers, uniformly distributed in $[0, 1]$; and $k = 1, 2, \ldots$ determines the iteration number.

3.2 Improved Particle Swarm Optimization

The sequence alignment problem can be considered as an optimization problem in which the objective is to maximize a scoring function. Thus, the PSO algorithm was adapted to be used with biological sequences. In the adapted PSO algorithm, a particle represents a sequence alignment. Thereby, as the main mechanism of the PSO algorithm is the movements of the particles towards the leader, suitable operators to implement this mechanism are proposed. The general algorithm is shown as follows[15]:

```
PSOMSA ( )
1. Generate a set of initial particles
2. Determine the leader particle gbest
3. Repeat until the termination criterion is met
```

```
        a. Measure distance between gbest and every particle
        b. Move every particle towards gbest
        c. Determine the leader partic
```

The termination criteria can be a maximum number of iterations, or a number of iterations after which the best score do not improve. The implicit idea of the PSO algorithm is that a set of particles that are randomly sparse over a search space will progressively move to regions which will provide better solutions to the problem, until the swarm finds a solution that it cannot improve anymore.

Next, some implementation details will be discussed, such as the particle representation, scoring function and the implementation of the particle movement mechanism.

Data Representation. In general, a swarm is made up by a set of particles, and a particle of the swarm is designated as the leader (gbest).Additionally, each particle preserves a memory with its best historical location (pbest). As mentioned above, in the adapted PSO algorithm, a particle will correspond to a sequence alignment. An alignment is then represented as a set of vectors, where each vector specifies the positions of the gaps for one of the sequences to be aligned. Thus, a coordinate of the particle corresponds to a sequence to be aligned, and is represented with a vector of size s, where s is the maximum allowed number of gaps, which may be different for each sequence. Therefore, a set of n sequences to be aligned correspond to an n-dimensional search space.

Initialization. The size of the swarm (i.e., the number of particles) is determined by the user. Additionally, the length of the alignment has a minimum value given by the length of the largest sequence, and a maximum length given, for instance, as twice the length of the largest sequence. The initial set of particles is generated by adding gaps into the sequences at random position, thus all the sequences have the same length L (the typical value of L is 1.4 times of the longest sequences).

Scoring Function. The global alignment score is based on the score of the alignment of each pair of sequences. Thus each sequence should be aligned with the every other sequence. In general, the score assigned to each particle (alignment) is the sum of the scores(SP) of the alignment of each pair of sequences. The score of each pair of sequences is the sum of the score assigned to the match of each pair of symbols, which is given by the substitution matrix. This matrix includes all the possible symbols, including the gap and the related penalization. The score of a multiple alignment is then:

$$SP - Score(A) = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} s(A_i, A_j) \tag{3}$$

where the $s(A_i, A_j)$is the alignment score between two aligned sequencesA_i and A_j.

Speed Update. In the PSO algorithm each particle moves towards the leader at a speed proportional to the distance between the particle and the leader. In this paper, the speed is defined:

$$v_{id}^{k+1} = c_1 r_1^k (p_{id}^k - x_{id}^k) + c_2 r_2^k (p_{gd}^k - x_{id}^k) \tag{4}$$

where c_1 and c_2 are the weights. If the value of v_{id}^{k+1} is not an integer, rounded it.

Position Update. After the update of speed, each particle updates its coordinate(x) according to:

$$x_{id}^{k+1} = x_{id}^k + v_{id}^{k+1} \tag{5}$$

When the position updated, the sequence may become illegal(Figure 1 shows an example).

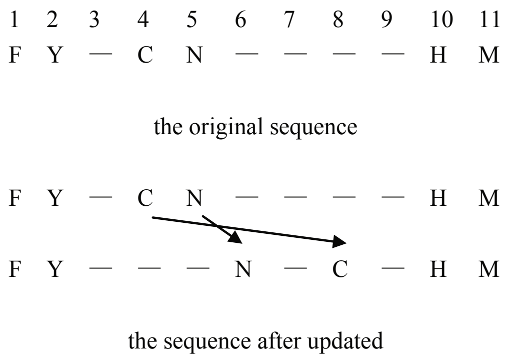

Fig. 1. An example of illegal update. The location of residue C and N is changed after update and the sequence is illegal.

An adjustment must be done in order to eliminate such illegal sequence. The adjustment is using

$$if \quad x_{id} \le x_{i(d-1)}, \quad then \quad x_{id} = x_{i(d-1)} + 1 \tag{6}$$

An example is depicted in Figure 2.

Operators. According to the traditional PSO, three types of operators are represented in PSOMSA: the insertion, deletion and local search.

1)Gaps Deletion

After a number of iterations, some columns may only have gaps. They not only useless, but also extended the length of alignment. Therefore, we remove them from the alignment.It is shown in Figure 3.

2) Gaps Insertion

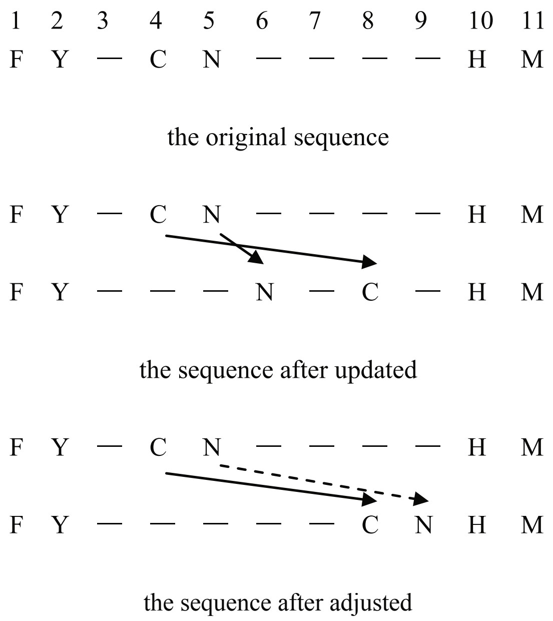

Fig. 2. An example of the adjustment of illegal sequence. The location of residue C is updated according to 5 and the residue of N is adjusted according to 6.

F	P	E	Y	V	N	K	-	-	D	F	H	I	A	G	E	S	Y
H	P	E	F	A	K	N	-	-	D	F	F	I	T	G	E	S	Y
F	P	E	Y	K	N	N	-	-	K	L	F	L	T	G	E	S	Y
F	P	E	D	L	T	R	-	-	K	I	I	L	S	G	E	S	Y

Fig. 3. An alignment with gap columns

The gaps insertion operators are added in order to avoid the PSO converge into a local optimal solution. We insert some gap columns into the current best alignment by a probability m. For example, we insert L columns into the alignment. The location to insert is defined in two ways as follows:

```
Insert L gap columns into a random location in the alignment
  or
Insert one gap column into the alignment at L random locations.
```

One of the operators is randomly selected by the program when it is running. Because the gap columns do not change the score of alignment, the gaps insertion operators do not help to improve the result of alignment. But the operators will affect the other particles, all particles' gaps will to be increased; thereby increasing the diversity of the algorithm and avoiding the algorithm converge into a local optimal solution.

3) Local Search Operator

In this paper, we use a local search to enhance the search performance. Assuming the count of gaps in the sequence which has the least gaps is L, we remove some gaps ranged from 0 to L from all the sequences. After that, we compute the score of the alignment. If the score is higher than before, the alignment is reserved; otherwise, we resume the alignment.

4 Simulation Experiments and Discussion

The proposed algorithm, called PSOMSA, was implemented in order to test its performance. The number of particles is determined taking into account the length of the sequences and the number of sequences to align. In order to test the algorithm, eight protein families of different length and different percentages of sequence identity were selected from the alignments database BALiBASE, available in [16]. One protein family was selected from each length category (short, medium and large), and one from each range of identity percentage (less that 25%, between 25% and 40% and greater that 35%). Table 1 presents the protein families used in the experiments. These protein families were previously

Table 1. Protein families used in the experiments

Reference	Name	Length	Identity%
1r69	repressor	Short	< 25
1tgxA	cardiotoxin	Short	20-40
1fmb	hiv-1 protease	Short	> 35
1mrj	alpha tricosanthin	Medium	20-40
1ezm	elastase	Medium	> 35
1cpt	cytochrome p450	Large	< 25
1ac5	b-galactoxidase	Large	20-40
1ad3	aldehyde dehydrogenase	Large	> 35

aligned using the well known algorithm Clustal X 2.0. The alignment obtained using Clustal was evaluated using a PAM250 matrix. The alignment was obtained using a penalty of 10 for each open gap and a penalty of 0.3 for each expend gap. The algorithm stopped after 10 gaps insertion operators without improving the quality of the solution. Table 2 shows the results of the experiments.

It can be found from the results that the PSOMSA algorithms has superior performance when compared to Clustal X, especially when the data has smaller sequences and shorter length. However, when the data has a longer length, the results is similar. There are still many enhancements that must be done to PSOMSA in order to achieve satisfying results. Also, new fitness functions based on different scoring methods are possible straightforward developments.

Table 2. Experimental results

Reference	Number of Sequences	Length	Clustal X Score	PSOMSA Score
1r69	4	63-78	2.7	287.9
1tgxA	4	57-64	447.9	890.7
1fmb	4	98-104	1513.4	1578
1mrj	5	247-266	2361.1	2367.9
1ezm	4	297-380	8223.8	8628.8
1cpt	5	378-434	1267.2	1795
1ac5	4	421-485	2814.5	3105.6
1ad3	4	424-447	5710.6	5726.8

5 Conclusions

In this work an algorithm based on the PSO algorithm, was proposed to address the multiple sequence alignment problem with SP score. The proposed approach was tested using some protein families and compared with the alignments generated by the Clustal X 2.0 algorithm. From simulation results, it is shown that the proposed PSOMSA algorithms has superior performance when compared to Clustal X .

In future work, additional experimentation should be performed, including experimentation with sequences of nucleic acids, the use of other algorithms to find initial sequence alignments , the use of other scoring schemes based on PAM or BLOSUM matrices and improving the speed of PSOMSA.

Acknowledgments

This research was supported by the NSFC (60573065), the Natural Science Foundation of Shandong Province (Y2007G33), and the Key Subject Research Foundation of Shandong Province.

References

1. Wang, L., Jiang, T.: On the Complexity of Multiple Sequence Alignment. J. Comput. Biol. 1(4), 337–348 (1994)
2. Needleman, S.B., Wunsch, C.D.: A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins. J. Mol. Biol. 48, 443–453 (1970)
3. Lawrence, C.E., Altschul, S., Boguski, M., Liu, J., Neuwald, A., Wootton, J.: Detecting Subtle Sequence Signals: a Gibbs Sampling Strategy for Multiple Alignment. Science 262, 208–214 (1993)
4. Hogeweg, P., Hesper, B.: The Alignment of Sets of Sequences and the Construction of Phyletic Trees: an Integrated Method. J. Mol. E 20, 175–186 (1984)
5. Feng, D.F., Doolittle, R.F.: Progressive Sequence Alignment as a Prerequisite to Correct Phylogenetic Trees. J. Mol. E 25(4), 351–360 (1987)

6. Thompson, J.D., Higgins, D.G., Gibson, T.J.: 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(22), 4673–4680 (1994)
7. Notredame, C., Higgins, D.G., Heringa, J.: T-Coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment. J. Mol. Biol. 302(1), 205–217 (2000)
8. Bahr, A., Thompson, J.D., Thierry, J.C., Poch, O.: Balibase (Benchmark Alignment Database): Enhancements for Repeats, Transmembrane Sequences and Circular Permutations. Nucl. Acids Res. 29(1), 323–326 (2001)
9. Thompson, J.D., Plewniak, F., Poch, O.: Balibase: A Benchmark Alignment Database for the Evaluation of Multiple Alignment Programs. Bioinformatics 15(1), 87–88 (1999)
10. Notredame, C., Higgins, D.G.: Saga: Sequence Alignment by Genetic Algorithm. Nucleic Acids Res. 24(8), 1515–1524 (1996)
11. Hirosawa, M., Totoki, Y., Hoshida, M., Ishikawa, M.: Comprehensive Study on Iterative Algorithms of Multiple Sequence Alignment. Comput. Appl. Biosci. 11(1), 13–18 (1995)
12. Henikoff, S., Henikoff, J.G.: Amino Acid Substitution Matrices from Protein Blocks. Proc. Natl. Acad. Sci. USA 89(22), 10915–10919 (1992)
13. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks, pp. 1942–1948. IEEE Press, Piscataway (1995)
14. Eberhart, R., Kennedy, J.: A New Optimizer Using Particle Swarm Theory. In: Proc. 6th Int Sympossum on Micro Machine and Human Science, pp. 39–43. IEEE Press, Piscataway (1995)
15. Rodriguez, P.F., Nino, L.F., Alonso, O.M.: Multiple Sequence Alignment using Swarm Intelligence. International Journal of Computational Intelligence Research 3(2), 123–130 (2007)
16. National Center for Biotechnology Information, `http://www.ncbi.nlm.nih.gov`
17. Clustal W Download Page, `http://www.clustal.org/download/current/`