ICOT Technical Report: TR-0864

TR-0864

Comprehensive Study on Iterative Algorithms of Multiple Sequence Alignment

by
M. Hirosawa (Kazusa DNA), Y. Totoki,
M. Hoshida (Matsushita) & M. Ishikawa

March, 1994

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191～5

Institute for New Generation Computer Technology

Comprehensive Study on Iterative Algorithms of Multiple Sequence Alignment

Makoto Hirosawa[1], Yasushi Totoki, Masaki Hoshida[2]
and Masato Ishikawa[3]

Institute for New Generation Computer Technology (ICOT), 1-4-28 Mita, Minato-ku, Tokyo 108, Japan

[1]Present address: Kazusa DNA Research Institute, 4-10-12 Chuo, Chuo-ku, Chiba-shi 260, Japan

[2]Present address: Tokyo Information Research Laboratory, Matsushita Electric Industrial Co., 4-5-15 Higashi-shinagawa, Shinagawa-ku, Tokyo 140, Japan

[3]To whom reprint requests should be sent

Key Words : Iterative Improvement, Dynamic Programming, Protein Similarity Analysis

Running Title : Iterative Algorithms of Multiple Sequence Alignment

Abstract

Multiple sequence alignment is an important problem in the biosciences. To date, most multiple alignment systems have employed a tree-based algorithm, which combines the results of 2-way dynamic programming in a tree-like order of sequence similarity. The alignment quality is not, however, high enough when the sequence similarity is low. Once an error occurs in the alignment process, that error can never be corrected. Recently, an effective new class of algorithms has been developed. These algorithms iteratively apply dynamic programming to partially aligned sequences to improve their alignment quality. The iteration corrects any errors that may have occurred in the alignment process. Such an iterative strategy requires heuristic search methods to solve practical alignment problems. Incorporating such methods yields various iterative algorithms. This paper reports our comprehensive comparison of iterative algorithms. We proved that performance improves remarkably when using a tree-based iterative method, which iteratively refines an alignment whenever two sub-alignments are merged in a tree-based way. We propose a tree-dependent restricted partitioning technique to efficiently reduce the execution time of iterative algorithms.

Introduction

The similarity analysis of protein/DNA sequences with multiple alignment is an important method for predicting function and structure, and for drawing phylogenetic trees of creatures. Many algorithms have been developed to help biologists align sequences.

Once a similarity value between characters is determined, Dynamic Programming (DP) (Needleman and Wunsch, 1970) can be used to theoretically solve a multiple alignment problem. N-way DP aligns N sequences simultaneously and derives the optimal alignment of these sequences. Computational time to solve a practical alignment problem, however, is incredibly long. Computational time with N-way DP is in the order of the N-th power of sequence length. With increasing computer power, 3-way DP (Murata, 1985) has become feasible. Search space restriction (Carrillo and Lipman, 1988) is making N-way DP on several similar sequences manageable, but N-way DP is not still fast enough to solve practical alignment problems.

To keep computational time within manageable limits, most multiple alignment systems employ 2-way DP as a base and combine the results of 2-way DP in a tree-like order of sequence similarity (Barton, 1990; Feng and Doolittle, 1986; Taylor 1987;

Higgins *et al.*, 1992). These algorithms, called tree-based algorithms, require small computational time, but they don't produce high-quality alignment when sequence similarity is low. Once an error occurs in the alignment process, the error can never be corrected.

Alignment algorithms that target high quality alignment, even when sequence similarity is low, have also been developed. Hirosawa *et al.*, (1993a) employed 3-way DP as the basis for an initial alignment, then refined the alignment by simulated annealing (Ishikawa *et al.*, 1993). Nevertheless, the algorithm still takes longer than tree-based algorithms.

Recently, Berger and Munson (1991) developed an iterative improving strategy for multiple alignment algorithms, and Gotoh (1993) focused on linear gap penalty in the iterative improving algorithm. This algorithm iteratively generates a next possible alignment with group-to-group 2-way DP. The groups of sequences are decided by randomly partitioning the whole alignment. When the next possible alignment is better than the present alignment, the new alignment becomes an input to the next iteration. The algorithm can remedy errors that occur in the alignment process. The computational time, however, is still too long for the user to wait for a prospective high-quality alignment result.

In our previous study (Ishikawa *et al.*, 1992), we revised the iterative improving strategy by introducing a best-first search and a restricted partitioning technique (Figure 1). The algorithm iteratively generates candidate alignments, with the best candidate selected as an input to the next iteration. The candidate alignments are obtained from the following heuristic partitioning. As partitioning divides N sequences into k sequences and $N - k$ sequences, a smaller k tends to provide a larger improvement when using group-to-group DP. The restricted partitioning technique preferentially selects partitions that have a small k, such as one or two. Although the algorithm was originally developed for a parallel computer, performance on a sequential computer is also good.

In this study, we chose some practical iterative alignment algorithms and compared their performance with a conventional tree-based algorithm.

System and Methods

The programs described in this paper are written in C language. They were tested on a SUN Sparcstation-10/model-30 (CPU: 36MHz). All programs are available from the authors upon request.

Algorithms

Tree-based algorithm

Various tree-based algorithms of multiple sequence alignment have been devised. From among them, we choose a typical algorithm to evaluate the performance of tree-based algorithms. A tree-based algorithm uses 2-way dynamic programming (DP) in a group-to-group manner (Barton, 1990) to align two sub-alignments.

In this algorithm, similarity between each pair of sequences is first estimated with its pairwise alignment score obtained by DP. Using a matrix of the similarity scores, UPGMA method (Sneath and Sokal, 1973) constructs a guided tree. Sequences are merged to form a multiple alignment based on the bottom-up branching order of the guided tree. Each node of the tree shows two bunches of sequences to which group-to-group DP is applied.

The group-to-group DP optimizes the alignment between groups. The score to be optimized is the summation of all pairwise alignment scores between the groups. The pairwise alignment score is derived from a similarity value between amino acids and a linear relation of gap penalty: $a + bk$ where k is the length of gap and a and b are the opening and extending gap cost. The optimizing operation in DP is the same as *Algorithm C*, explained in detail by Gotoh (1993). In the other algorithms described below, the same type of DP is used to align two sub-alignments.

Round-robin iterative algorithm

Barton and Sternberg (1987) proposed the simplest iterative improvement concept for achieving refinement against a resulting alignment obtained by a tree-based algorithm. In the method, group-to-group DP realigns each sequence against the whole alignment, except for the current sequence. This process is repeated in a round-robin manner.

A round-robin iterative algorithm applies the refinement method to an initial arbitrary state of multiple alignment: normally there are no gaps in the sequences to be

aligned. Accordingly, sequence S_1 is aligned with the alignment of sequences $S_2...S_n$ (having first removed any gaps that are common to $S_2...S_n$). S_2 is then realigned with the alignment of $S_1, S_3...S_n$. This process is repeated until S_n has been realigned with $S_1, S_2...S_{n-1}$. The complete cycle is repeated until no change occurs.

Random iterative algorithm

The original iterative improvement algorithm starting from a no-gap alignment was found by Berger and Munson (1991). Random numbers play the following important role in the iterative algorithm.

First, an initial N sequence alignment is input into an iteration cycle. The sequences are divided by random numbers into two groups: a k sequence alignment and an $N-k$ sequence alignment. The two partial alignments are then recombined by group-to-group DP. Since the score of the resulting alignment is always better than or equal to the previous one, the new alignment is set at the starting point of the next iteration cycle. In this way, application of the iteration cycle gradually improves the whole alignment. The iteration terminates when a specific number of continuous cycles give no improvement. The quality of the final result depends mainly on how effective partitions have been tested in the iteration cycles.

The random iterative algorithm requires a huge number of iteration cycles to solve a practical problem. N-sequence alignment has $2^{N-1}-1$ ways of partitioning: more than 2,000,000 partitions when $N = 22$. To be practicable, a heuristic technique is needed to significantly restrict search space and reduce execution time. We studied three restricted partitioning techniques: single-type partitioning, double-type partitioning, and tree-dependent partitioning.

single-type partitioning: The number of sequences in the smaller sub-alignment of partitioning is restricted to one, while the other sub-alignment has $N-1$ sequences when the number of aligned sequences is N. Since the number of possible partitions is N, the order of partitioning complexity is reduced from 2^N to N with this partitioning technique.

double-type partitioning: The number of sequences in the smaller sub-alignment of partitioning is restricted to one or two, while the other sub-alignment has $N-1$ or $N-2$ sequences. Possible partitions are $N(N+1)/2$. The order of partitioning complexity N^2 is bigger than that of single-type partitioning.

tree-dependent partitioning: Partitioning is restricted to the ways indicated by branches of a guided tree (Figure 2). Branch separations are $2N - 3$ when the number of sequences is N (Allison *et al.*, 1992). Construction of the guided tree is based on a current multiple alignment at the beginning of each iteration cycle. This technique adequately considers the similarity of aligned sequences. Although this partitioning technique requires overhead for constructing the guided trees, the order of partitioning complexity is the same N as that of the single-type partitioning.

The three techniques were incorporated in a random iterative algorithm. In the iteration cycle, random numbers are used to select each possible partition at the same probability. These techniques allow the iterative algorithm to solve a practical multiple alignment problem.

Best-first iterative algorithm

The random iterative algorithm selects a partition randomly, whereas the best-first iterative algorithm tests all possible partitions and selects the best alignment (Figure 1). The iteration terminates when all possible partitions give no improvement. Restricted partitioning techniques are also required in the algorithm to solve practical problems.

Tree-based iterative algorithm

The tree-based iterative algorithm consists of the iterative improvement strategy and the tree-based algorithm. Each alignment is refined by an iterative algorithm, just after the two sub-alignments are merged in a tree-based way (Subbiah and Harrison, 1989). The search schemes, such as random and best-first, bring variety to the tree-based iterative algorithm. Restricted partitioning techniques can reduce execution time.

Experimental results

Alignment score

Experimental results are compared under the same scoring system of multiple sequence alignment. The N-sequence alignment score is the summation of $N(N-1)/2$ pairwise alignment scores. Each score is the summation of every similarity value between aligned amino acids and of every penalty of gap inserted in the sequence pair. The similarity

values are from table PAM250 (Dayhoff 1978). The gap penalty is defined as a linear relation $a + bk$, where the opening and extending gap costs are $a = -7$ and $b = -1$. An out-gap penalty is defined without the opening gap cost as bk, where the extending out-gap cost is $b = 0$.

Test sequence sets

We gathered thirty sequences of different protein kinase as mother sequences. We then cut eighty amino acids, starting from the ATP-binding site, out of each mother sequence. Then we obtained thirty test sequences; each had a sequence length of eighty. Twenty-two randomly selected test sequences formed the test set of sequences. Repeating the random selection thirty times gave us thirty different test sets whose homologous ratios were 27 to 29 percent. Each experiment was executed on the thirty sets. Figure 3 shows a typical alignment of the test sets. The alignment was generated by the tree-based iterative algorithm with best-first search and tree-dependent partitioning. The alignment score was 14,545.

Performance comparison

Figure 4 shows performance of the algorithms. Each algorithm was executed on the thirty test sets to optimize the alignment score. In RIAS, RIAD and RIAT, the average of three trials with distinct random numbers is displayed. The trials started from the no-gap alignment and terminated when no improvement occurred over 300 iteration cycles.

TA	Tree-based algorithm
RRIA	Round-robin iterative algorithm
TA+RRIA	Algorithm refined with RRIA after alignment by TA
RIAS	Random iterative algorithm with single-type partitioning
RIAD	Random iterative algorithm with double-type partitioning
RIAT	Random iterative algorithm with tree-dependent partitioning
BIAS	Best-first iterative algorithm with single-type partitioning
BIAD	Best-first iterative algorithm with double-type partitioning
BIAT	Best-first iterative algorithm with tree-dependent partitioning
TRRIA	Tree-based round-robin iterative algorithm
TBIAS	Tree-based best-first iterative algorithm with single-type partitioning

TBIAD Tree-based best-first iterative algorithm with double-type partitioning

TBIAT Tree-based best-first iterative algorithm with tree-dependent partitioning

The resulting alignment scores are compared in the upper part of Figure 4. The scores obtained from the same test set are connected by dotted lines. Each score is normalized by all connected scores; the difference from the average of thirteen scores is divided by the average itself. Bold lines connect the average scores of the thirty test sets. Average execution time of the thirty test sets is also shown in the lower part. The comparison yielded the following information.

(i) Although the tree-based algorithm (TA) is the fastest, its average score is the worst.

(ii) On the average, the best-first iterative algorithms (BIAS, BIAD and BIAT) take more execution time but yield better scores than the random iterative algorithms (RIAS, RIAD and RIAT).

(iii) Round-robin iterative improvement after TA alignment (TA+RRIA) shows better performance in both average score and execution time than the random iterative algorithms or the best-first iterative algorithms.

(iv) The tree-based iterative algorithms (TRRIA, TBIAS, TBIAD and TBIAT) yield the best average scores of all algorithms, and their execution times compare favorably with those of other algorithms.

(v) The tree-based iterative algorithms show no significantly different average scores in partitioning technique. The tree-based iterative algorithm with round-robin search (TRRIA) is the fastest.

(vi) Average scores among the random iterative algorithms and among the best-first iterative algorithms differ significantly in partitioning technique. Tree-dependent partitioning yields the best performance, although it takes nearly twice as long to execute as single-type partitioning.

Discussion

Our comprehensive study on iterative algorithms proved that the tree-based iterative algorithms work better for optimizing the multiple alignment score than the other

iterative algorithms or the conventional tree-based algorithm. Test sequence sets for random and best-first iterative algorithms did not show better performance than the algorithm using round-robin iterative improvement after tree-based alignment.

Tree-dependent partitioning tends to yield the best performance among the restricted partitioning techniques, which reduce the execution time of iterative algorithms. Performance of the tree-based iterative algorithm did not increase significantly, even when tree-dependent partitioning was used. This lack of increase may have resulted because the sequence similarity in the test sets was not low enough for tree-dependent partitioning to produce a prominent effect.

The sum-of-pair scoring system was used in our experiments. Other scoring systems, such as tree and star system (Altschul and Lipman, 1989) could not be incorporated in the iterative algorithms. Regardless of the scoring system, however, the optimal-score alignment is not always the most significant result in a biological sense. In addition to optimizing alignments under some scoring system, it is also important to refine them using biological knowledge (Hirosawa *et al.*, 1993b).

Acknowledgements

The authors would like to thank Dr. Osamu Gotoh of the Saitama Cancer Center for his valuable discussions.

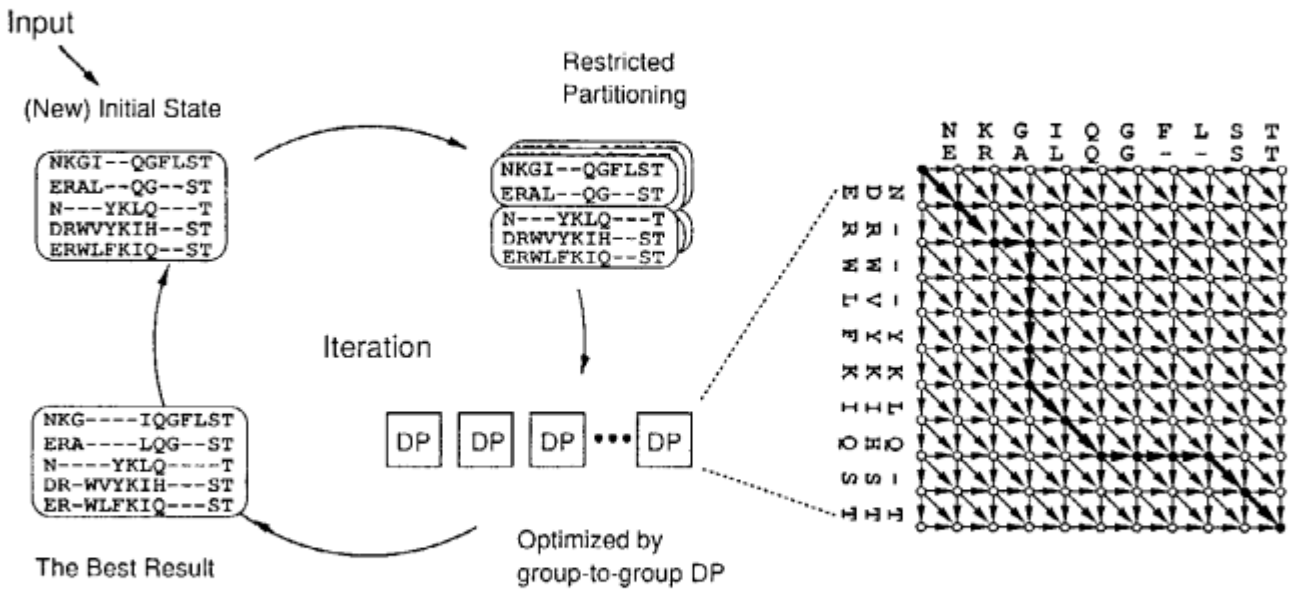

Fig. 1. Scheme of best-first iterative algorithm. A current alignment is separated into many pairs of sub-alignments. Each pair of sub-alignments is realigned by dynamic programming. The best score result is regarded as a new current alignment. The iteration is repeated as long as a current alignment improves.

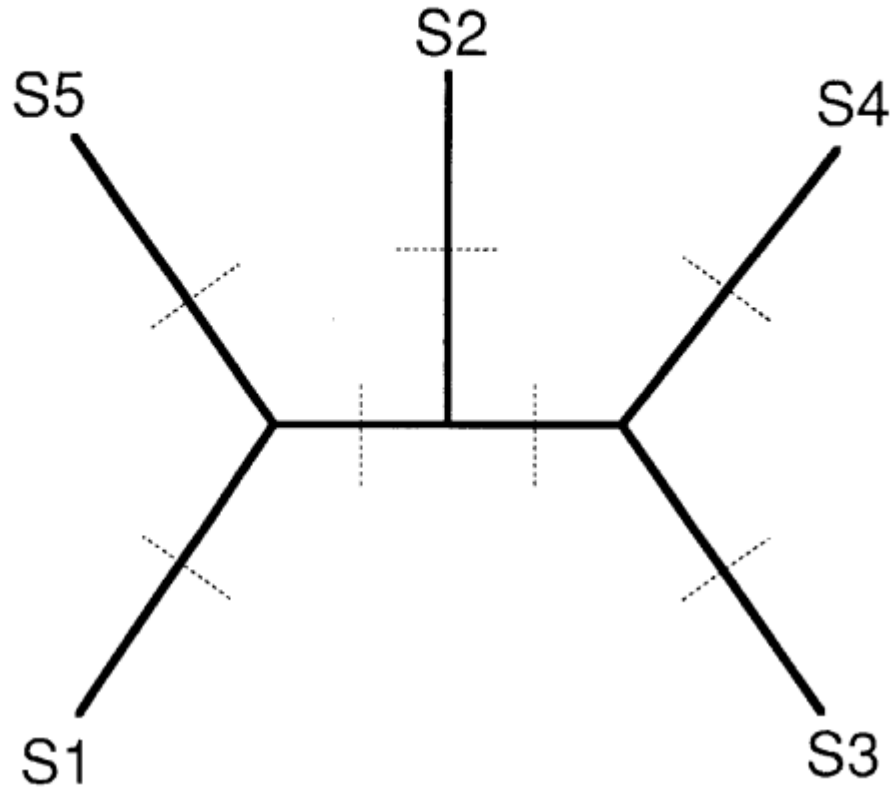

Fig. 2. Tree-dependent partitioning. Each dotted line on a branch indicates a partitioning in a guided tree, which presents similarity distances among sequences. The five sequences are divided into $(S1, S5)$ and $(S2, S4, S3)$, $(S1, S5, S2)$ and $(S4, S3)$, and so on. Partitioning for N sequences is $2N - 3$.

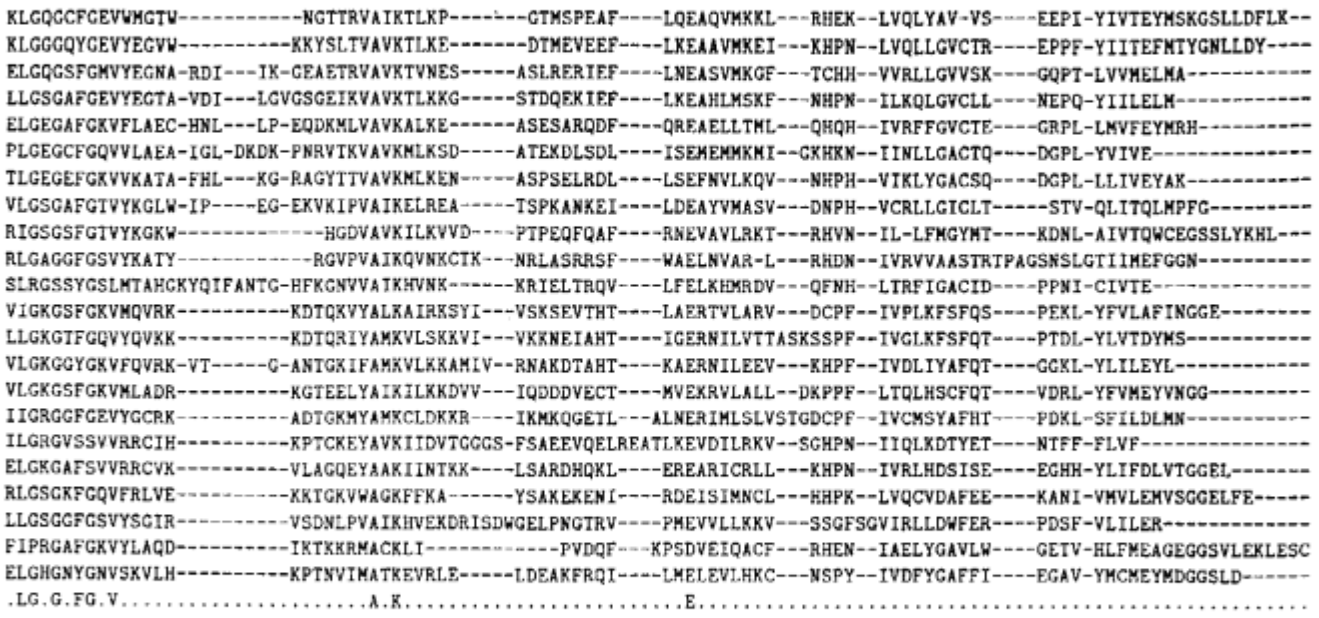

Fig. 3. A typical multiple sequence alignment obtained in the experiments. Each sequence is a part of protein kinase that includes the ATP-binding site. The last row contains the eighty-percent consensus sequence.

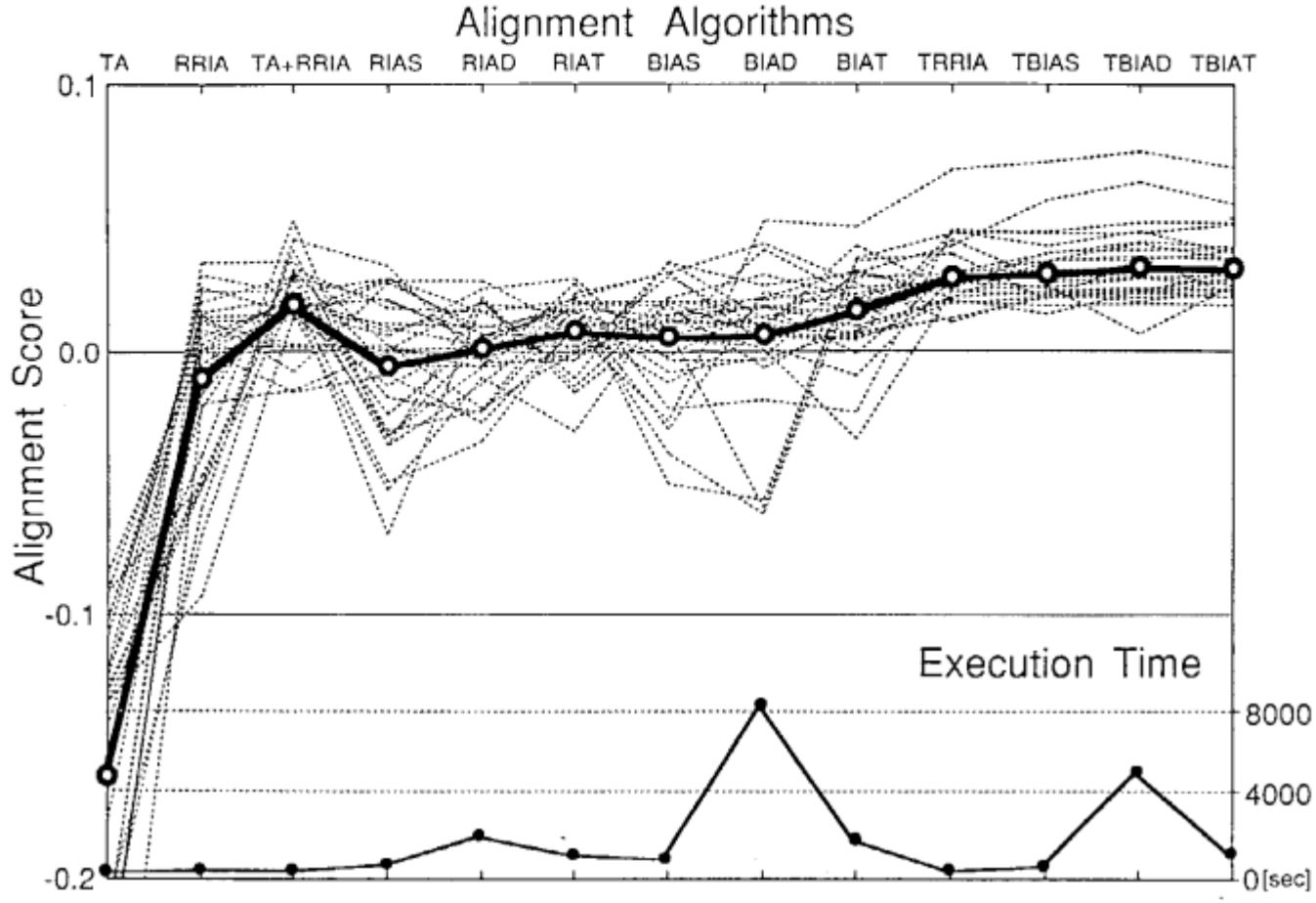

Fig. 4. Performance of alignment algorithms compared over thirty test sequence sets. TA: tree-based algorithm. RRIA: round-robin iterative algorithm. RIAS, RIAD and RIAT: random iterative algorithm with single-type partitioning, with double-type partitioning and with tree-dependent partitioning. BIAS, BIAD and BIAT: best-first iterative aligorithm with single-type partitioning, with double-type partitioning and with tree-dependent partitioning. TRRIA: tree-based RRIA, etc.

References

Allison,L., Wallace,C.S. and Yee,C.N. (1992) Minimum message length encoding, evolutionary trees and multiple-alignment. *Proc. 25th Hawaii Int'l Conf. Sys. Sci.*, **4**, 663-674.

Altschul,S.F. and Lipman,D.J. (1989) Trees, stars and multiple biological sequence alignment. *SIAM J. Appl. Math.*, **49**, 197-209.

Barton,J.G. and Sternberg,M.J.E. (1987) A strategy for rapid multiple alignment of protein sequences. *J. Mol. Biol.*, **198**, 327-337.

Barton,J.G. (1990) Protein multiple alignment and flexible pattern matching. In Doolittle,R.F.(ed), *Methods in Enzymology Vol.183*, Academic Press, 403-427.

Berger,M.P. and Munson,P.J., (1991) A novel randomized iterative strategy for aligning multiple protein sequences. *CABIOS*, **7**, 479-484.

Carrillo,H. and Lipman,D.J. (1988) The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.*, **48**, 1073-1082.

Dayhoff,M.O., Schwatz,R.M. and Orcutt,B.C. (1978) A model of evolutionary change in proteins. In Dayhoff,M.O.(ed), *Atlas of Protein Sequence and Structure Vol.5, Suppl.3*, Nat. Biomed. Res. Found., Washington D.C., 345-352.

Feng,D.F. and Doolittle,R.F. (1987) Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.* **25**, 351-360.

Gotoh, O. (1992) Optimal alignment between groups of sequences and its application to multiple alignment. *CABIOS*, **9**, 361-370.

Higgins,D.G., Bleasby,A.J. and Fuchs,R. (1992) CLUSTAL V: improved software for multiple sequence alignment. *CABIOS*, **8**, 189-191.

Hirosawa,M., Hoshida,M., Ishikawa,M. and Toya,T. (1993a) MASCOT: multiple alignment system for protein sequence based on three-way dynamic programming. *CABIOS*, **9**, 161-167.

Hirosawa,M., Hoshida,M. and Ishikawa,M. (1993b) Protein multiple sequence alignment using knowledge. *Proc. 26th Hawaii Int'l Conf. Sys. Sci.*, **1**, 803-812.

Ishikawa,M., Hoshida,M., Hirosawa,M., Toya,T., Onizuka,K. and Nitta,K. (1992) Protein sequence analysis by parallel inference machine. *Proc. Fifth Gener. Comp. Sys. '92*, 294-299.

Ishikawa,M., Toya,T., Hoshida,M., Nitta,K., Ogiwara,A. and Kanehisa,M. (1993) Multiple sequence alignment by parallel simulated annealing. *CABIOS*, **9**, 267-274.

Murata,M., Richardson,J.S. and Sussman,J.L. (1985) Simultaneous comparison of three protein sequences *Proc. Natl. Acad. Sci. USA*, **82**, 3073-3077.

Needleman,S.B. and Wunsch,C.D. (1970) A general method applicable to the search for similarities in the amino acid sequences of two proteins. *J. Mol. Biol.*, **48**, 443-453.

Sneath,P.H.A and Sokal,R.R. (1973) *Numerical Taxonomy*, Freeman and Company.