PAAA: A Progressive Iterative Alignment Algorithm Based on Anchors

Ahmed Mokaddem and Mourad Elloumi

Research Unit of Technologies of Information and Communication,
Higher School of Sciences and Technologies of Tunis, Tunisia
moka.ahmed@yahoo.fr,
Mourad.Elloumi@fsegt.rnu.tn

Abstract. In this paper, we present a new iterative progressive algorithm for *Multiple Sequence Alignment* (MSA), called *Progressive Iterative Alignment Algorithm Based on Anchors* (PAAA). Our algorithm adopts a new distance, called *anchor distance* to compute the distance between two sequences, and a variant of the UPGMA method to construct a guide tree.

PAAA is of complexity $O(N^4 + N * L^2)$ in computing time, where N is the number of the sequences and L is the length of the longest sequence.

We benchmarked PAAA using different benchmarks, e.g., BALIBASE, HOMSTRAD, OXBENCH and BRALIBASE, and we compared the obtained results to those obtained with other alignment algorithms, e.g., CLUSTALW, MUSCLE, MAFFT and PROBCONS, using us criteria the *Column Score* (CS) and the *Sum of Pairs Score* (SPS). We obtained good results for protein sequences.

Keywords: Multiple sequence alignment, progressive alignment, algorithms, complexities, distances.

1 Introduction

Multiple Sequences Alignment (MSA) is a good way to compare biological sequences. It consists in optimizing the number of matches between the residues occurring in the same order in each sequence. MSA is an NP-complete problem [22]. There are several approaches to solve this problem. *Progressive approach* is the most used and the most effective one, it operates in three steps:

1. Pairwise comparisons: during this step, we compute a distance between every pair of sequences. We can compute this distance directly, i.e., without constructing pairwise alignments, or indirectly, i.e., after the pairwise alignments. The goal of this step is to estimate the similarity between pairs of sequences in order to distinguish the sequences that are the first to be aligned. The distances computed between all pairs of sequences are stored in a symmetric diagonal matrix, called *distance matrix*. Among used distances, we mention *percent identity*, *percent of similarity* [19], *Kimura distance* [9], *k-mer distances* [2] and *normalized score* [23].

2. Sequences clustering: during this step, we define the branching order of the sequences by construction a *guide tree*. Clustering method uses the distance matrix constructed in the previous step. The most used methods are UPGMA [16] and *Neighbor-Joining* [15].
3. Sequences integration: during this step, we construct the alignment of sequences using the branching order. We mention *aligning alignment approach* [23] and *profile-profile alignment approach* [6]. The last one is the most used method.

Among progressive alignment algorithms, we mention CLUSTALW [18], TCOFFEE [12], GRAMALIGN [14] and KALIGN [11]. A Progressive approach can apply a refinement step. Refinement step is used in order to enhance the multiple alignment scores. In fact, a set of operations can be applied iteratively on the alignment under convergence, i.e., the score of the alignment obtained at iteration N is less than the score of the alignment obtained at iteration N-1. Different strategies of refinement have been developed [21]. These algorithms are called *hybrid algorithms* or *iterative progressive alignment algorithms* and are almost the most efficient. Among hybrid alignment algorithms, we mention MUSCLE [2], PROBCONS [1], MAFFT [8], PRRP [7] and MSAID [24].

In this paper, we present a new iterative progressive alignment algorithm, called PAAA. The main differences between our algorithm and other progressive ones consist in:

– First, the use of a new distance, called *anchor distance*, in the pairwise comparisons step.
– Then, the use of the *maximal distance* instead of the *minimal distance*, in the construction of the *guide tree*, in the sequences clustering step.

PAAA implements also a refinement step. We benchmarked PAAA using different benchmarks, e.g., BALIBASE, HOMSTRAD, OXBENCH and BRALIBASE, and we compared the obtained results to those obtained with other alignment algorithms, e.g., CLUSTALW, MUSCLE, MAFFT and PROBCONS, using us criteria the *Column Score* (CS) and the *Sum of Pairs Score* (SPS). We obtained good results for protein sequences.

The rest of the paper is organized us follows. In section 2, we present some preliminaries. In section 3, we present PAAA and in section 4, we compute its time complexity. In section 5, we present an illustrative example. In section 6, we present the experimental results obtained with PAAA using different benchmarks and we compare the obtained results with those of other alignment algorithms. Finally, in section 7, we present our conclusion and give some perspectives.

2 Preliminaries

Let A be a finite alphabet, a *sequence* is an element of A^*, it is a concatenation of elements of A. The length of a sequence w, denoted by $|w|$, is the number of the characters that constitute this sequence. A portion of w beginning at the position i and ending at the position j, $0 \leq i \leq j \leq |w|$, is called *subsequence*.

Let f =$\{w_1, w_2, \ldots, w_N\}$ be a family of sequences, we say that a subsequence x is an *anchor* to the sequences of f, if and only if, x is a subsequence of every sequence of f.

Let f be a set of sequences, a list of anchors that appears in the same order, without overlapping in all the sequences of f will be denoted by *AC(f)*.

Let f be a set of sequences, *aligning* the sequences of f consists in optimizing the number of matches between the characters occurring in the same order in each sequence. When $|w| \succ 2$, aligning the sequences of f is called *Multiple Sequence Alignment* (MSA).

A *profile* is a sequence that represents an alignment. It is constructed by selecting for each column of the alignment the residue that has the maximum occurrences in this column.

A *guide tree* is a binary tree where a leaf represents a sequence, a node represents a group of sequences, and therefore the corresponding alignment. The nodes define the branching order and the root defines the final alignment.

The *anchor distance* between two sequences w and w' is calculated using the following formula:

$$D(w, w') = 1 - \frac{\sum_{a \in AC(\{w,w'\})} |a|}{min(|w|, |w'|)} \quad (1)$$

Selected anchors are fundamental in the computation of the anchor distance. In our case, we select the longest common anchors. Hence, the more two sequences have longer anchors, the more these sequence are similar.

3 Algorithm

Let f be a set of sequences, by adopting PAAA, we operate as follows:

- During the first step, we compute the anchor distance between each pair of sequences of f and store the computed distances in a distance matrix *M*. The computation of the anchor distance between a pair of sequences can be done by adopting our algorithm described in [3].
- During the second step, we use the distance matrix M to build the guide tree T. This is done by adopting a variant of UPGMA [16] method. We made some modifications to the UPGMA method as follows : First, we select the maximum value from the distance matrix M, instead of the minimum one, and we align the corresponding sequences. Then, we align the aligned sequences with another one following the branching order of the guide tree. We reiterate this process until we obtain a MSA of all the sequences. We adopt two dynamic programming algorithms, the first one [5] for aligning two sequences, and the second one [6] for aligning two profiles. The profile of two alignments is constructed using the profile of each alignment.
- Finally, during the last step, we make a refinement to the resulting alignment in order to correct the mistakes that could have been produced in the progressive approach. The method adopted is the one of algorithm MUSCLE [2]. We operate as follow: First, we use the MSA obtained to construct

a new guide tree using a new distance matrix. Then, we remove an edge from the new guide tree to obtain two new guide subtrees. Next, we align the sequences in each guide subtree. Finally, we align the two alignments associated with the two guide subtrees to obtain a global MSA. We repeat this step until convergence, i.e., no improvement is observed on the MSA obtained during the last iteration, compared with the MSA obtained during the previous one.

4 Time Complexity

In this section, we study time complexity of our algorithm PAAA. Let N be the number of the sequences and L be the length of longest sequence.

- Time complexity of the first step is $O(N^2 * L^2 * log(L))$. In fact, we compute the anchor distance between every pair of sequences by adopting our algorithm, described in [3], that is of complexity $O(L^2 * log(L))$ in computing time. We have $O(N^2)$ anchor distances to compute then time complexity of the first step is $O(N^2 * L^2 * log(L))$.
- Time complexity of the second step is $\mathrm{O}(N^2)$. Indeed, the construction of the guide tree requires a running time of $\mathrm{O}(N^2)$ [16].
- Finally the refinement step requires a time complexity of $O(N^4 + N * L^2)$ [2]. Hence our algorithm PAAA is of complexity $\mathrm{O}(N^4 + N * L^2)$ in computing time.

5 An Illustrative Example

Let f be a set of sequences $f = \{w_1, w_2, w_3, w_4, w_5\}$

w_1: *seqvecccce*; w_2: *stqesmcedd*; w_3: *tescd*; w_4: *teqvenced*; w_5: *tescmcccctes*

We compute anchor distance between every pair of sequences:

$D(w_1, w_2)$ = 0,5; $D(w_2, w_3)$ = 0,6; $D(w_3, w_4)$ = 0,2; $D(w_4, w_5)$ = 0,56

$D(w_1, w_3)$ = 0,6; $D(w_2, w_4)$ = 0.34; $D(w_3, w_5)$ = 0,2;

$D(w_1, w_4)$ = 0.34; $D(w_2, w_5)$= 0,4;

$D(w_1, w_5)$ = 0,4;.

We store these distances in the distance matrix M

Table 1. Distance matrix

	w_1	w_2	w_3	w_4	w_5
w_1					
w_2	50				
w_3	60	0			
w_4	34	34	20		
w_5	40	40	20	56	

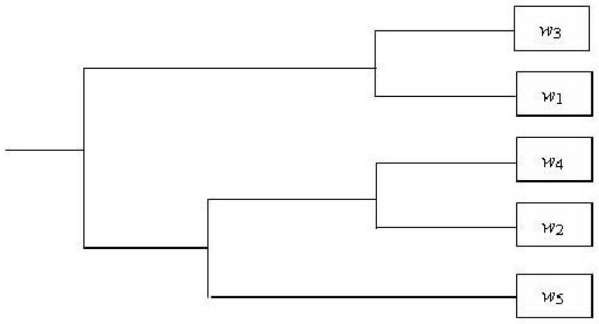
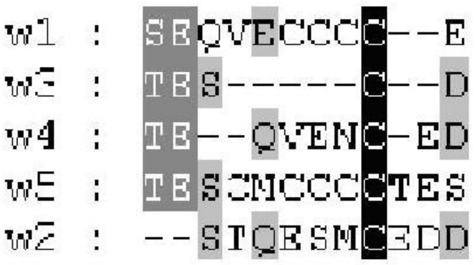


Fig. 1. Guide Tree



```
w1 : SEQVECCCC--E
w3 : TES-----C--D
w4 : TE--QVENC-ED
w5 : TESCMCCCCTES
w2 : --STQESMCEDD
```

Fig. 2. MSA

Finally, we align the sequences following the branching order. Hence we obtain the MSA in Fig.2..

Figure 2 was obtained thanks to GeneDoc [25].

6 Experimental Study

In order to asses PAAA performances; we used different benchmark for protein and DNA sequences. Each benchmark contains multiple alignments that have been constructed in a manual, or automatic, way with the help of biologists and have been grouped in different categories, according to the nature of the set of the sequences.

These alignments are considered as *reference alignments*. We used the following benchmarks: BALIBASE [20], OXBENCH [13], HOMSTRAD [17] and BRALIBASE II [4] for DNA sequences.

We compared the results of PAAA with other multiple alignment programs that are the most widely used and the most efficient namely, CLUSTALW2 [18], MUSCLE [2], MAFFT [8] and PROBCONS [1]. In order to compare alignments, we use two criteria:

– CS: The *Column Score* [19] :

$$CS = \sum \frac{C_i}{L} : \tag{2}$$

Where, C_i=1 if all the residues in the test aligned are aligned in the reference alignment otherwise C_i=0 and L represents the total number of column in the *core blocks*.

– SPS: The *Sum of Pairs Score* [19] :

$$SPS = \frac{\sum_1^M S_i}{\sum_1^{M_r} S_r} \tag{3}$$

Where S_i is the pairs of residues correctly aligned in the column i, M is the number of columns in test alignment, S_r is the residue in the reference alignment and M_r is the number of columns in the reference alignment.

We ran PAAA on datasets from the BALIBASE benchmark. In order to compute the SPS and the CS, we used the *bali_score* program, a program provided with the BALIBASE benchmark, that generates these scores based on predetermined reference alignments. For OXBENCH and HOMSTRAD benchmarks, we computed the Q-scores and TC-scores, using the *Q-score* program. TC score and Q-score correspond respectively to the CS and SPS of the BALIBASE benchmark. The results of CS and SPS obtained with a dataset from BALIBASE are respectively represented in Table 2 and Table 3.

Table 2. Results obtained with BALIBASE using CS

Dataset	CLUSTALW2	MUSCLE	MAFFT	PROBCONS	PAAA
1DOX(REF1)	0,919	0,860	0.860	0.907	0,950
1PYSA(REF1)	0.922	0.920	0.910	0.930	0.940
1ZIN(REF1)	0.920	0,970	0,920	0,940	0,980
1UBI(REF2)	0.482	0.120	0.260	0.380	0,210
1CSY(REF2)	0,110	0,240	0,110	0,110	0,290
1ABOA(REF2)	0.650	0.230	0.330	0.330	0.420
1VLN(REF4)	0.879	0.710	0.700	0.800	0.770
KINASE1(REF4)	0.000	0.650	0.710	0.750	0.490
1DYNA(REF4)	0.000	0.090	0.130	0.090	0.180
1PYSA(REF4	0.000	0.350	0.360	0.360	0.330
1UKY (REF3)	0,190	0,180	0,170	0,190	0,210

Table 3. Results obtained with BALIBASE using SPS

Dataset	CLUSTALW2	MUSCLE	MAFFT	PROBCONS	PAAA
1DOX(REF1)	0.914	0.924	0.920	0.907	0.969
1PYSA(REF1)	0.957	0.930	0.931	0.952	0.952
1ZIN(REF1)	0,960	0,985	0,960	0,965	0,987
1UBI(REF2)	0.822	0.805	0.791	0.843	0.754
1CSY(REF2)	0,826	0,835	0,770	0,809	0,828
1VLN(REF4)	0.925	0.907	0.901	0.963	0.947
KINASE1(REF4)	0.869	0.862	0.875	0.908	0.802
1DYNA(REF4)	0.226	0.396	0.517	0.578	0.461
1PYSA(REF4)	0.555	0.667	0.675	0.673	0.646
1ABOA(REF2)	0.825	0.814	0.804	0.804	0.787
1UKY(REF3)	0,529	0,529	0,581	0,566	0,571

We obtained more good results for CS than SPS thanks to the anchors selection step. In fact, we selected exactly repeated subsequences and hence promoted the CS.

We also benchmarked PAAA on a dataset from OXBENCH and we compared obtained TC scores and Q-scores with those of typical softwares CLUSTALW, MUSCLE and MAFFT. OXBENCH uses TC score and Q-score. We obtain the following results that represent the average of TC scores and Q-scores of a set of 100 datasets from OXBENCH.

This table shows that PAAA returns the best TC score and Q score for the dataset from OXBENCH. We assessed also PAAA in on set of families extracted from HOMSTRAD database and we obtained the following results.

Table 4. Results obtained with OXBENCH using TC score and Q-score

PROGRAM/SCORES	Nbr of Dataset	TC	Q
MUSCLE	100	0,693	0,828
MAFFT	100	0,687	0,818
CLUSTALW	100	0,682	0,819
PAAA	100	0,706	0,832

Table 5. Results obtained with HOMSTRAD database using TC score and Q score

FAMILY/ALGORITHM	PAAA		MAFFT		CLUSTALW		MUSCLE	
SCORES	Q	TC	Q	TC	Q	TC	Q	TC
ornithine carbamoyltransferase	0,939	0,872	0,937	0,861	0,931	0,849	0,927	0,84
cryst	0,876	0,613	0,495	0,0166	0,874	0,657	0,652	0,021
C-typelectin	0,867	0,662	0,83	0,628	0,838	0,697	0,862	0,634
HLH	0,937	0,869	0,824	0,753	0,785	0,684	0,931	0,869
Nucleotidekinase	0,897	0,699	0,849	0,541	0,849	0,533	0,869	0,618
LUXS	0,911	0,848	0,892	0,845	0,913	0,869	0,881	0,797
Peroxidase	0,899	0,736	0,866	0,67	0,885	0,713	0,896	0,734

Table 5 shows that for this set of families extracted from HOMSTRAD benchmark our algorithm have the best scores compared to other typical ones.

We also ran PAAA on reference families from BRALIBASE II benchmark of DNA sequences. For DNA sequences, we obtained less good results than for protein. As we can see in Table 6 and Table 7, these tables represent respectively the average of CS and SPS.

These tables show that PAAA results are better than those of DIALIGN, POA and T-COFFEE. In addition, for TRNA and G2intron families PAAA is better than CLUSTAL W2, but gives less good scores than MUSCLE, MAFFT and PROBCONS.

Table 6. Results obtained with SPS on BRALIBASE II (DNA)

Method(DNA)	G2In	Rrna	SRP	tRNA	U5
PAAA	0,747	0,917	0,854	0,855	0,774
DIALIGN2.2	0,717	0,899	0,815	0,786	0,762
CLUSTALW2	0,727	0,933	0,874	0,870	0,796
T-COFFEE5.56	0,738	0,909	0,839	0,817	0,791
POAV2	0,672	0,889	0,855	0,769	0,773
MAFFT6.240L-INSi	0,789	0,939	0,875	0,918	0,828
MAFFT6.240E-INSi	0,774	0,938	0,872	0,906	0,805
MUSCLE3.7	0,764	0,940	0,871	0,873	0,797
PROBCONSRNA1.10	0,801	0,945	0,881	0,926	0,848

Table 7. Results obtained with CS on BRALIBASE II (DNA)

Method(DNA)	G2In	Rrna	SRP	tRNA	U5
PAAA	0,607	0,848	0,746	0,769	0,614
DIALIGN2.2	0,609	0,811	0,685	0,676	0,601
CLUSTALW2	0,612	0,867	0,766	0,762	0,651
T-COFFEE5.56	0,602	0,826	0,716	0,692	0,629
POAV2	0,552	0,804	0,738	0,660	0,616
MAFFT6.240L-INSi	0,652	0,875	0,768	0,846	0,685
MAFFT6.240E-INSi	0,638	0,873	0,766	0,833	0,657
MUSCLE3.7	0,632	0,880	0,766	0,780	0,643
PROBCONSRNA1.10	0,687	0,886	0,776	0,855	0,717

The obtained alignments for families of proteins and families of DNA are good, compared to those of other programs. Let us notice that the results obtained for families of proteins are better than those obtained for families of DNA thanks to the selected anchors step. In fact, the selected anchors have great influence on the distances computation. Indeed, the probability of having repeated subsequences in a DNA sequence is higher than the one of having repeated subsequences in proteins sequence. On the other hand, the more there are occurrences representing the same repeated subsequence in a sequence the more PAAA is likely to be induced in error, because of the choice of the occurrence associated with the repeated subsequence. This is why we obtain better results with proteins compared to DNA.

7 Conclusions and Perspectives

In this paper, we have presented a new iterative progressive alignment algorithm called PAAA. This algorithm uses a new distance called *anchor distance*, in the sequence comparison step, and a new variant of UPGMA method, in the guide tree construction step. This variant uses the maximum distance from the distance matrix. PAAA makes also a refinement step. We assessed PAAA on different benchmarks and we compared its performances with other typical multiple alignment programs, using the SPS and the TC score. We obtained good results. Indeed, in several cases, for protein sequences, we obtained better scores than those of typical multiple alignment programs, e.g., MUSCLE, MAFFT, CLUSTALW and PROBCONS.

As a future work, we plan to improve PAAA by developing new methods for sequences classification and for profile-profile alignments. Finally, we intend to improve the iterative step.

References

1. Do, C.B., Mahabhashyam, M.S., Brudno, M., Batzoglou, S.: PROBCONS: Probabilistic Consistency-Based Multiple Sequence Alignment. Genome Res. 15, 330–340 (2005)
2. Edgar, R.C.: MUSCLE: Multiple Sequence Alignment with high Accuracy high Throughput. Nucleic Acids Research 32, 1792–1797 (2004)
3. Elloumi, M., Mokaddem, A.: A Heuristic Algorithm for the N-LCS Problem. Journal of the Applied Mathematics, Statistics and Informatics (JAMSI) 4, 17–27 (2008)
4. Gardner, P.P., Wilm, A., Washiet, S.: A benchmark of multiple sequence alignment programs upon structural RNAs. Nucleic Acids Research 33, 2433–2439 (2005)
5. Gotoh, O.: An Improved Algorithm for Matching Biological Sequences. J. Mol. Biol. 162, 705–708 (1982)
6. Gotoh, O.: Further Improvement in Methods of Group-to-Group Sequence Alignment with Generalized Profile Operations. CABIOS 1, 379–387 (1994)
7. Gotoh, O.: Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. J. Mol. Biol. 264, 823–838 (1996)
8. Katoh, K., Kuma, K., Toh, H., Miyata, T.: MAFFT version 5: Improvement in Accuracy of Multiple Sequence Alignment. Nucleic Acids Res. 33, 511–518 (2005)
9. Kimura, M.: The Neutral Theory of Molecular Evolution. Cambridge University Press, Cambridge (1983)
10. Lassman, T., Frings, O., Sonnhammer, L.L.: KALIGN2: High-Performance Multiple Alignment of Protein and Nucleotide Sequences Allowing External Features. Nucleic Acids Research 37, 858–865 (2009)
11. Lassman, T., Sonnhammer, L.L.: KALIGN: An Accurate and Fast Multiple Sequence Alignment Algorithm. BMC Bioinformatics 6 (2005)
12. Notredame, C., Higgins, D., Heringa, J.: T-COFFEE: A novel mMthod for Multiple Sequence Alignments. J. Mol. Biol. 302, 205–217 (2000)
13. Raghava, G.P., Searle, S.M., Audley, P.C., Barber, J.D., Barton, G.J.: OXBENCH: a Benchmark for Evaluation of Protein Multiple Sequence Alignment Accuracy. BMC Bioinformatics 4 (2003)

14. Russell, D.J., Out, H.H., Sayood, K.: Grammar-Based Distance in Progressive Multiple Sequence Alignment. BMC Bioinformatics 9 (2008)
15. Saitou, N., Nei, M.: The Neighbor-Joining Method: A New Method for Reconstructing Phylogenetic Trees. Mol. Biol. E 4, 406–425 (1987)
16. Sneath, P., Sokal, R.: Numerical Taxonomy, pp. 230–234. Freeman, San Francisco (1973)
17. Stebbings, L.A., Mizuguchi, K.: HOMSTRAD: Recent Developments of the Homologous Protein Structure Alignment Database. Nucleic Acids Research 32, 203–207 (2004)
18. Thompson, J.D., Higgins, T.J., Gibson, D.G.: CLUSTALW: Improving the Sensitivity of Progressive Multiple Sequence Alignment through Sequence Weighting, Position Specific Gap Penalties and Weight Matrix Choice. Nucleid Acids Research 22, 4673–4680 (1994)
19. Thompson, J.D., Plewniak, F., Poch, O.: A Comprehensive Comparison of Multiple Sequence Alignment Programs. Nucleic. Acids. Res. 27, 2682–2690 (1999)
20. Thompson, J.D., Plewniak, F., Poch, O.: BAliBASE: a Benchmark Alignment Database for the Evaluation of Multiple Alignment Programs. Bioinformatics 15, 87–88 (1999)
21. Wallace, I.M., O'Sullivan, O., Higgins, D.G.: Evaluation of Iterative Alignment Algorithms for Multiple Alignments. Bioinformatics 21, 1408–1414 (2005)
22. Wang, L., Jiang, T.: On the complexity of multiple sequence alignment. J. Comput. Biol. 1, 337–348 (1994)
23. Wheeler, T.J., Kececioglu, J.D.: Multiple alignment by aligning alignments. Bioinformatics 23, 559–568 (2007)
24. Min, Z., Weiwu, F., Junhua, Z., Zhongxian, C.: MSAID: Multiple Sequence Alignment Based on a Measure of Information Discrepancy. Computational Biology and Chemistry 29, 175–181 (2005)
25. Nicholas, K.B., Nicholas, K.B., Deerfield, D.W.: GeneDoc: Analysis and Visualization of Genetic Variation. Embnew News 4 (1997)