# Caching Needlemann-Wusnch scores to improve progressive sequence alignment optimality

**Furaha Damien, 260754407**[1]

[1] McGill University

Monträal, Quäbec, Canada

Email: :furaha.damien@mail.mcgill.ca

**Abstract — Progressive sequence alignment is a heuristic approach that is used in multiple sequence alignment. The approach builds up a final Multiple sequence alignment by combining pairwise alignments beginning with the most similar pair and progressing to the most distantly related[1].The problem with progressive sequence alignment is that it is a greedy algorithm that makes the most optimal alignment at every step of the alignment. This means that if errors are made in the earlier alignments, they are propagated to the final result. In this paper, I propose a technique for increasing the optimality of the algorithm. Rather than use a guide tree[2] only as most version of the progressive sequence alignments algorithms do, I make us of the vertical information of the tree guides by storing the scores of the respective pairwise sequence alignments. This means that given n sequences $S_1$, $S_2$, ........ $S_n$, I first use the Needlemann-Wusnch algorithm to calculate the pairwise sequence alignment scores.I then create a ranking algorithm obtains a list of all pairwise alignments sorted in decreasing score for every sequence. The scores for each pair are cached. These cached values are then referred to every time we want to align the next sequence to the previous sequences beginning with the highest scoring pairs, keeping in mind the tree guides that I generate using the UGPMA algorithm. The proposed method was applied to randomly generated 10 sets of DNA sequences. The results of these alignments were then compared to those obtained from the traditional Progressive Sequence alignments. Using this approach, I observe that I was able to obtain the most optimal multiple sequence alignment. I also run the algorithm on RNA sequences obtained form Balibase[9] database. Due to complexity and size of the sequence, I fall short of computational resources to run the alignment step of the algorithm on these sequences, something that future research should focus on.**

## 1. Introduction

Multiple sequence alignment is one of the most important tasks in bioinformatics and it is used to achieve some of the most basic Bioinformatic tasks like phylogenetic inference. One of the approached used to achieve this multiple sequence alignments is the Progressive sequence alignment approach. This heuristic method first constructs a guide tree[2] that portrays the phylogenetic relationship between the sequences. The tree is then used in the next step. Here the sequences are aligned iteratively with the highly related sequences(according to the phlylogenetic tree) aligned first and then their alignements aligned to the next highly related sequence. This second step is a greedy approach as it makes the most optimal alignment at that step without taking into consideration the subsequent sequences that form the phylogenetic tree. What this means is that the algorithm highly depends on the evolutionary tree created in the first step. However, this approach does not always work especially if the sequences being aligned are very distantly related or very closely related. For instance, given four sequences $S_1$, $S_2$, $S_3$, $S_4$, if the a pairwise sequence alignment algorithm produces $S_1$ and $S_2$ as being optimally aligned to $S_3$ and $S_4$ with the highest alignment scores, this will create a tree guide such that $S_1$, $S_2$ and $S_3$ and $S_4$ form sub-trees with a shared root. As a result running the traditional progressive sequence alignment algorithm on these sequences produce an alignment such that the alignment of $S_1$, $S_2$ is aligned against that of $S_3$, $S_4$. This will however not be the most optimal alignment. The problem is that when the various paired alignments are grouped, they are seldom consistent from one to another[2]. Thus, if sequence $S_1$ is paired with sequence $S_2$, gaps may appear at various locations, but when either $S_1$ or $S_2$ is aligned with a third sequence, say $S_3$ , the arrangement of gaps may be entirely different from when they are aligned with $S_4$. This means that, if a less optimal alignment is made with regard to the subsequent sequences, the error is propagated to the final result. More or so, this approach yields bad results when the sequences are very distantly related. Since multiple sequence alignment is an important procedure in bioinformatics and progressive sequence alignment is one of the main approaches available to achieve this, there is a need for an improvement to this approach.

Here, I propose a new algorithm that adds an extra step to the traditional progressive sequences alignment algorithm and test it on several DNA sequences. The method has the ability to provide an optimal sequence alignment of distantly related sequences at the same run time as the current methods used by T-COFFEE for example.

## 2. Previous work

Multiple sequence alignment is an important procedure in a biology and bioinformatics. Given how important it is and how costly it use to run the progressive sequence alignment algorithm while not expecting the most optimal result due to the its greedy nature, several projects have been build around improving this algorithm. The suggested algorithms range from those working on optimizing the tree guides[2] to adding new steps in the algorithms[4]. Da-Fei Feng et al[2] proposes an improvement to building the tree guides by adding neutral elements in when aligning sequences which result in gaped alignments. Julie D. Thompson et al[4] uses sequence weighting, position-specific gap penalties and weight matrix choice to improve the algorithm. Both these approaches improve the algorithm but they still short in some areas[2,4].

## 3. Methodology:

The current improvements to the sequence alignment problem still fall short of improving the optimality of the progressive sequence alignment problem with good efficiency. To improve this, I propose a new multiple sequence alignment algorithm that uses the traditional pairwise sequence alignment algorithms as its underlying algorithm. I use the Needlemann-Wusnch pairwise sequence alignment algorithm for this purpose. I use the score from the pairwise alignment to build the tree guides. However, unlike other algorithms, I cache these scores in a data structure and refer to them in the sequential building of the Multiple sequence alignment using a ranking algorithm that I create.

### *3.1. Algorithm*

Given a set of n sequences that need to be aligned using multiple sequence alignment, I first do a pairwise alignment of all the sequences and store the pairwise sequence alignment scores in a n by n matrix. After aligning the scores, for each of the sequences, I store a list of the scores obtained by aligning this sequence against all the other n-1 sequences. I then sort the list in decreasing order. This allows to know which sequence is closely or distantly related to each of the sequences. This information is used in the final step of the proposed algorithm. In the next step, just like several other sequence alignment algorithms, I build a tree guide that represent the most probable evolutionary relationship between the sequences. This phylogenetic tree is build using the efficient clustering UPGMA[5] algorithm.

I use the tree guides and the cached pairwise scores to build our final multiple sequence alignment. Unlike the traditional progressive sequence alignment approach that looks at any of the two leaves that are part of the same subtree, in this algorithm I refer to our score table generated by the Needlemann-Wusnch pairwise sequence alignment. From the table, we obtain the highest scoring pair as our starting pair since the tree guides will have made these as leaves of the same binary tree. We do this for all leaves of a direct binary sub-tree. In the next step obatin the highest scoring pair(alignment) as presented by both the tree guide and the pairwise sequence alignment. From the cached list of the pairwise sequence alignment scores of each of the sequences in the pair, I look at their next highest scoring pairwise alignment. If the sequences that comes next in the list are the same for both of the sequences, we obtain the alignment made by the sub-tree from which this sequence(s) originates. We align this alignment against the previously obtained alignment. If the two sequences have different next highest scoring sub-sequences, then we obtain both of their alignments in their sub-trees and align them to the alignment. We do this for all the sub-alignments until we obtain the final Multiples Sequence Alignment with all the sequences.

### *3.2. Sudo code*

**Algorithm 1** pairwise sequence alignment

```
procedure PAIRWISE ALIGNMENT
    $Sequences \leftarrow S_1, S_2 ... S_n$
    initialize matrix D[n][n]
loop:
    for i = 0 to n do:
        for j = 0 to n do:
            if i != j then:
                //Needlemann-Wusnch Algorithm
                $D[i][j] \leftarrow score$
loop:
            for i = 0 to n do:
                initialize iScores[n]
                for j = 0 to n do:
                    if i == j then:
                        $iScore[j] \leftarrow infinity$
                    else:
                        $iScore[j] \leftarrow D[i][j]$
                    $iScore$.sort()
```

The algorithm has two phase:

1. The pairwise alignment phase that uses the Needlemann-Wusnch. It creates a an n by n matrix and n lists of size n. This phase has two double loops, both running through each sequence twice. the second double loop has an extra step that sorts the lists, which we do using quick sort

Running time

**Algorithm 2** Progressive sequence alignment

```
procedure PAIRWISE ALIGNMENT
    $treeT \leftarrow UPGMA$
loop:
    for all binary leaves do:
        align maxScore pairs
        for all binary alignments do:
            for each leaf do:
                $maxSequence \leftarrow max(iScore)$
                align maxScore sub-tree to this subtree
Return Alignment
```

first double for loop : $O(n^2.L^2)$

second double for loop : $O(n^2) + O(nlog(n)$

space complexity:

first double for loop : = $O(n^2)$

second double for loop : $O(n^2) + O(n^2)$

2. The second phase of the algorithm finds the multiple sequence alignment. It builds the tree using UPGMA and then sequentially builds MSA. At every alginment step, it gets the highest scoring pair whcih was computed in the first phase.

Running time: $O(n^2) + O(n^2) * L = O(n^2) * L$

Space Complexity : = $O(n) * 2L$

where L is the length of the longest sequence

### *3.3. Experimental set up*

The algorithm is divided into two parts[section 3.2]. In the first part, I used the Needlmann-Wusnch algorithm to obtain pairwise sequence alignment and a list of scores indicating how each of the sequence is related to the rest. I sort the respective lists in decreasing order. The second part uses both score matrix and the obtained lists and builds the tree guides from which we build our MSA as explained[section 3.1]. To test the effectiveness of our approach, we used a two-phase approach. In the first phase, we used simple sequences.

$s_1 = CGCAGGCAACAGTGGCTTCG$

$s_1 = GAACGGTAGAGGGTGACCAC$

$$s_1 = CAGTCCCGTCCCGACAGCAG$$
$$s_1 = CTTAGCGGAGAGGCCCGCCA$$
$$s_1 = ACCTGGCGGTTAGCGCTGCG$$
$$s_1 = CGACGGCGACGCTCCTCACC$$

We used the used the Needlemn-Wusnch to align them and UPGMA to generate tree guides.

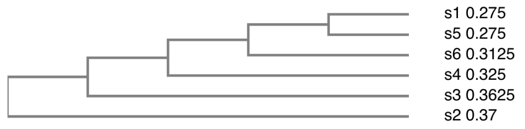

Figure 1 : Tree generated from $S_1 ... S_6$ by UPGMA.

In this first phase, I did not write the code UPGMA algorithm code but rather run it on the European Bioinformatics institute[8] servers. I then used the proposed algorithm to rank into decreasing order the most closely related sequences. Using this ranking, I then used the scores of the respective sequences in relations to each other to iterative build the multiple sequence alignment. In the second phase of the algorithm I used sequence data obtained from Balibase[9]. I obtained 11 RNA sequences with each being at least 400 pb in length. I run our algorithm on these RNA sequences.

For both data sets that we run our experiment on, we run the sequences on the DNA Alignment software[8].

## 4. Results:

I run the algorithm on the two data-sets that I had. The simple data set that contained six sequences each with ten nucleotide. After running the algorithm, for each of the sequence, I obtained the following ranking in relations to the other sequences using the Needlemann-Wusnch algorithm.

$s_1 : s_5 > s_6 > s_2 > s_4 > s_3$
$s_2 : s_4 > s_6 > s_1 > s_5 > s_3$
$s_3 : s_5 > s_6 > s_1 > s_4 > s_2$
$s_4 : s_2 > s_2 > s_5 > s_1 > s_3$
$s_5 : s_1 > s_4 > s_3 > s_6 > s_2$
$s_6 : s_4 > s_1 > s_3 > s_2 > s_5$

From the phylogenetic tree guide obtained after running the UPGMA algorithm from the European Bioinformatics Institute, and using the sequence relationship ranking that I obtained from using Needlemann-Wusnch, I obtained a Multiple sequence alignment[Table 1:] that was similar what I expected. This first step was used as a control for the development of my algorithm as running it on short sequences would help me to easily pick out any instances where it did not work.

| | |
|---|---|
| s3 | - - - - - - C A G T C C C G T C C C G A C A G C A G |
| s1 | C G C A G G C A A C A G T G G C T T C G - - - - - - |
| s5 | A C C T G G C G G T T A G C G C T G C G - - - - - - |
| s2 | - - - G A A C G G T A G A G G G T G A C C A C - - - |
| s4 | - - C T T A G C G G A G A G G C C C G C C A - - - - |
| s6 | - - - - C G A C G G C G A C G C T C C T C A C C - - |

Table 1 : MSA generated

In the second phase of the experiment, I run the algorithm on the long sequence data sets obtained from the Balibase[9]. The sequences were long, about 400 base pairs each. Running the algorithm on them, I obtained a similarity ranking and used this alongside the phylogenetic tree guide also obtained using the European Bioinformatics institute servers. However, I was unable to generate a multiple sequence aligmnet with this data because I realized that I had to build my own UPGMA algorithm to be able to feed the tree that I would obtain in the the alignment algorithm.

## 5. Discussion and future work

The construction of multiple sequence alignments is among the most important techniques to perform biological sequence analysis, with important applications to many areas of computational biology. The progressive sequence alignment algorithm is one of the key techniques used to achieve this multiple sequence alignment of biological sequences. While this algorithm achieves optimal result in most cases, its greedy nature means that most of the times errors incurred in aligning sequences earlier on in the process are propagated to the final Multiple Sequence Alignment.

I have proposed an improvement to the progressive sequence alignment algorithm that makes maximum use of the vertical information to improve the optimality of the final Multiple Sequence alignment. By caching the scores of every pairwise alignment of the sequences obtained from Needlemann-Wusnch Algorithm and referring to the scores when building the final alignment, I iteratively build the alignment depending on the subsequent highest scoring pairwise alignments based on the ranking algorith than I create.

The proposed algorithm worked well on the short sequence data set that we tested it on. However, due to the size of the long sequence data set, I did not run the last step of the algorithm that involves building the alignment due to shortage of compute power on my machine. In future work, the algorithm needs to be run on long complex data sets like the one obtained from the Balibase[9]

## References

[1] Feng DF, Doolittle RF (1987). ”Progressive sequence alignment as a prerequisitetto correct phylogenetic trees”. J Mol Evol. 25 (4): 351„360. doi:10.1007/BF02603120. PMID 3118049.

[2] Zhan, Q., Ye, Y., Lam, T. W., Yiu, S. M., Wang, Y., Ting, H. F. (2015). Improving multiple sequence alignment by using better guide trees. BMC bioinformatics, 16 Suppl 5(Suppl 5), S4. doi:10.1186/1471-2105-16-S5-S4

[3]

[4] Julie D. Thompson, Desmond G. Higgins, Toby 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 Research, Volume 22, Issue 22, 11 November 1994, Pages 4673„4680, https://doi.org/10.1093/nar/22.22.4673

[5] Sokal, Michener (1958). ”A statistical method for evaluating systematic relationships”. University of Kansas Science Bulletin. 38: 1409„1438.

[6]

[7] http://www.fluxus-engineering.com/align.htm

[8] https://www.ebi.ac.uk/Tools/services/web/toolresult.ebi?jobId=simple$_phylogeny-I20191130-220442-0393-47447392-p1manalysis=tree$