

# Chapter 11

## **Multiple Sequence Alignment computation using the T-Coffee regressive algorithm implementation**

Edgar Garriga<sup>1</sup>, Paolo Di Tommaso<sup>1</sup>, Cedrik Magis<sup>1</sup>, Ionas Erb<sup>1</sup>, Leila Mansouri<sup>1</sup>, Athanasios Baltzis<sup>1</sup>,  
Evan Floden<sup>1</sup>, Cedric Notredame<sup>1,2\*</sup>

<sup>1</sup> Centre for Genomic Regulation, The Barcelona Institute of Science and Technology, Barcelona, Spain.

<sup>2</sup> Universitat Pompeu Fabra (UPF), Barcelona, Spain.

\* Correspondence: [cedric.notredame@crg.eu](mailto:cedric.notredame@crg.eu) (C.N.)

***Running Head:*** T-Coffee regressive algorithm implementation

## **Abstract**

Many fields of biology rely on the inference of accurate multiple sequence alignments (MSA) of biological sequences. Unfortunately, the problem of assembling an MSA is NP-complete thus limiting computation to approximate solutions using heuristics solutions. The progressive algorithm is one of the

most popular frameworks for the computation of MSAs. It involves pre-clustering the sequences and aligning them starting with the most similar ones. The scalability of this framework is limited, especially with respect to accuracy. We present here an alternative approach named regressive algorithm. In this framework, sequences are first clustered and then aligned starting with the most distantly related ones. This approach has been shown to greatly improve accuracy during scale-up, especially on datasets featuring 10,000 sequences or more. Another benefit is the possibility to integrate third-party clustering methods and third-party MSA aligners. The regressive algorithm has been tested on up to 1.5 million sequences, its implementation is available in the T-Coffee package.

**Keywords** Sequence alignment, MSA, guide-tree, progressive alignment.

# 1 Introduction

Multiple Sequence Alignment (MSA) is an NP-complete problem whose computation relies on approximate heuristic solutions. The most common solution is the progressive method [1]. This method starts by aligning the most similar sequences following a pre-computed guide-tree, but the accuracy drops when dealing with a large number of sequences.

The regressive method [2] works the other way around and starts by aligning the most diverse sequences going from the root of the guide-tree to the leaves. MSAs are constructed through a divide and conquer process during which smaller MSAs - named sub-MSAs - encompassing the more diverse sequences are gradually expanded until all sequences have been incorporated within the final model. Extensive benchmark analyses carried out on Homfam [3] and Pfam [4] have shown that the regressive algorithm

is both more scalable than regular methods - it was shown to align 1.5 million sequences - and also more accurate, especially when dealing with datasets larger than 10,000 sequences.

An important characteristic of the T-Coffee implementation of this algorithm is its modularity. It allows several third-party methods to be used in order to both estimate the guide tree and to apply the most commonly used alignment algorithms - including the consistency based version of T-Coffee [5] - to perform the sub-MSAs during the divide and conquer stage.

---

## 2 Materials

### 2.1 Equipment setup

- Computer: Any computer running Linux or Mac OSX with access to the internet.
- Software: T-Coffee can be downloaded from <http://tcoffee.org/Packages/Stable/Latest> It is distributed as a set of precompiled binaries for Linux and Mac OSX platforms 32-bit or 64-bit) with a guided install procedure. This is the smoothest and quickest way to install T-Coffee on a local machine, as it comes with all the required components and does not require any special user privileges. It is also possible to download the source code from GitHub or use it from Conda or Docker containers.
- Sequence to align: [www.tcoffee.org/Projects/regressive/datasets/protocols.tar.gz](http://www.tcoffee.org/Projects/regressive/datasets/protocols.tar.gz)

### 2.2 Procedure

T-Coffee: obtaining and installing t-coffee

Install T-Coffee by following one of the following options, some of them are possible to run on both Linux or MacOSX operating systems (OS) and others are specific to each OS.

### 2.2.1 Binary

#### - Linux

(i) Download the installer package from <http://tcoffee.org/Packages/Stable/Latest/linux/>

(ii) Grant execution permission to the downloaded file with the following command:

```
chmod +x T-COFFEE_installer_<version_>.bin
```

(iii) Launch the installation wizard with

```
./T-COFFEE_installer_<version_>.bin
```

(iv) Follow the wizard instructions and complete the installation.

(v) Open a new terminal session to be sure that your environment is updated.

(vi) Type the following command to verify that the installation was successful:

```
t_coffee -version
```

#### - MacOSX

(i) Download the installer package from <http://tcoffee.org/Packages/Stable/Latest/macosx/>. With Mac OSX 10.5.x (Leopard) and earlier versions, users have to use the 32-bit installer version; with Mac OSX 10.6.x (Snow Leopard) and above, users have to use the 64-bit installer version.

(ii) Double-click on the DMG file to open it.

(iii) Double-click on the installer icon (within the mounted image) to start the installation.

(iv) Follow the wizard instructions and complete the installation.

(v) Open a new terminal session to be sure that your environment is updated.

(vi) Type the following command to verify that the installation is successful:

```
t_coffee -version
```

### **2.2.2 Compilation from source**

(i) Follow the instructions from the T-Coffee GitHub page: [www.github.com/cberg/tcoffee](http://www.github.com/cberg/tcoffee)

(ii) Go inside the source folder

```
cd t_coffee/src
```

(iii) Compile the package with

```
make t_coffee
```

(iv) add the compile folder in your path.

```
mv t_coffee /bin/
```

### **2.2.3 Docker**

From the command line you can download the docker container with the following command:

```
docker pull cberg/tcoffee_protocols
```

You can use the container with any of the workflow managers, or run it in an iterative mode using the command:

```
docker run -ti --mount type=bind,source=/<path_to_data>/,target=/<container_data_folder>/  
cberg/tcoffee_protocols
```

### **2.2.4 Conda**

To install the conda package you should download from bioconda channel with the following command:

```
conda create --name tcoffee_protocols -c bioconda t-coffee
```

```
conda activate tcoffee_protocols
```

This package includes all the 3rd party software needed for this protocol, but we can always generate an environment combining T-Coffee with other software.

---

## 3 Methods

The T-Coffee regressive algorithm has been developed to allow the computation of ultra-large MSAs. The algorithm's main steps are as follows:

- #1 - computation of a rooted guide tree using any relevant method, including mBed [6] and PartTree [7] (see Note 1).
  
- #2 - label each node with the label of the longest sequence among its progeny (Figure 1), (i.e., the root will be labeled with the longest sequence), Figure 1.
  
- #3 - starting from the root node (parent node), and going one generation at a time, collect N nodes
  - N is a free parameter. In Figure 2, N is set to 3 but in practice, N is set to 1,000. Its value can be changed via the parameter **-reg\_nseq**
  
- #4 - carry out an MSA of the N Sequences that label the N nodes. For instance, in Figure 2, with N=3 this will involve sequences 5,9 and 12 (blue envelope). This MSA is named a parent subMSA and it can be computed using any third-party aligner.

- #5 - run step #3-4 on every node selected in #3 that is not a leaf, the resulting subMSAs will be the children MSAs of the parent subMSA computed one step earlier. For instance, in Figure 3, the children subMSAs will be made of sequences (1,5), (9,3) and (12,2,8). The procedure stops once every leaf node has been incorporated in an MSA.
- #6 - Since every MSA shares the sequence of its parent node with its parent MSA, the children and their parent subMSAs can be combined through these common sequences without the need of an extra alignment step, as shown in Figure 4. Combining the subMSAs merely involves stacking the columns linked by their common sequence. (see Note 2, Note 3)

The regressive algorithm has been shown to have exceptional scalability [2]. One of the reasons for this is the reliance on a strict divide and conquer procedure that never involves aligning more than  $N$  sequences. As a consequence, for  $M$  input sequences, the deployment of any third-party method - regardless of its original complexity - becomes linear in time and memory as it merely involves carrying out  $M/N$  individual MSAs. Moreover, the independence of these MSAs makes their computation an embarrassingly parallel problem.

Aside from its algorithmic properties, the regressive implementation of the T-Coffee algorithm also brings many added benefits through the seamless integration of a large number of third-party clustering and alignment methods. Overall, 5 clustering methods are supported along with 5 multiple sequence aligners. The package comes along with an extensive documentation allowing non-supported alignment methods to be incorporated via simple configuration files.

In the next section we explore various combinations of clustering methods and alignment algorithms that allow users to explore different trade-offs between accuracy and efficiency. For instance, it is possible to very rapidly estimate ultra-large models by combining the fastest clustering method (PartTree) with the fastest MSA method (MAFFT default). The same framework makes it possible to combine a slower but more accurate tree method (like mBed) with a very accurate MSA method (like MAFFT-ginsi) that only allows aligning a few hundred sequences but can be massively scaled-up by the regressive framework.

### 3.1 Validated Method Combinations

The following combinations of pre-clustering and alignment methods were validated in the original paper for their relative speed and accuracy. They are recommended for large scale analysis

#### **3.1.1 Fast and accurate**

This mode offers the best trade-off between speed and accuracy. It relies on the ClustalO mBed trees that were found to yield the highest accuracy on large datasets while the combination of these guide trees with the ClustalO aligner results in alignments of reasonable accuracy.

```
t_coffee -reg -seq gluts.fasta -reg_nseq 1000 -reg_tree mbed -reg_method clustalo_msa -outfile gluts.aln  
-outtree gluts.mbed
```

#### **3.1.2 Slower and more accurate**

As discussed earlier, the regressive algorithm framework can be used to deploy methods that would be prohibitive on any dataset larger than 1,000 sequences. In the example below, we show how the MAFFT-ginsi method can be deployed on large datasets. On the HomFam dataset, this protocol required about 4.7 times more CPU time (as compared with the fast approximate mode using fftns1), but resulted in a 21 % improvement in the number of correctly aligned columns.

```
t_coffee -reg -seq gluts.fasta -reg_nseq 1000 -reg_tree mbed -reg_method mafftginsi_msa -outfile gluts.aln  
-outtree gluts.mbed
```

### **3.1.3 Very Fast and approximate**

On the other end of the spectrum, the combination of the fastest aligner with the fastest clustering method provides the most efficient alignment method currently available. This combination is about 3 times faster than the fast and accurate ClustalO combination.

```
t_coffee -reg -seq gluts.fasta -reg_nseq 1000 -reg_tree parttree -reg_method mafftfftnsi_msa -outfile gluts.aln  
-outtree gluts.parttree
```

### **3.1.4 Further Method Combinations**

A major strength of the regressive algorithm is its capacity to support any method combination of interest to the user. All these combos have not been validated so far, but they are nonetheless supported and available for exploration.

One of the main limitations of both the progressive and the regressive procedures is the generation of the guide-tree because not all the clustering methods are able to handle a large number of sequences.

The Regressive method has the advantage that it allows to use any clustering method from which a tree can be obtained, making it possible to use algorithms that work well with big data.

T-Coffee offers some built-in options for building trees from a range of clustering algorithms, and they can be used with the **-reg\_tree** flag.

- mbed : use mBed mode of ClustalO - Default
- cwdnd : use the quicktree mode of ClustalW
- parttree : parttree method of MAFFT - fastest option. Does not support sequences less than 6 AA long
- dpparttree : MAFFT fast clustering method
- fastparttree: MAFFT fast clustering method
- mafftdnd : default MAFFT NJ tree - slower than the parttree modes
- fftns1dnd : Tree produced after the first iteration MAFFT fftns mode
- fftns2dnd : Tree produced after the second iteration MAFFT fftns mode
- upgma : upgma tree - warning cubic time computation
- nj : Neighbour Joining tree
- #<command> : Runs comamnd <seq> > <tree>.
- filename : Any file in newick format. The seq file and the tree file must match

Thanks to the possibility to freely combine guide-trees and alignment methods, the regressive algorithm allows the usage of highly accurate methods (limited to a small set of sequences) or less accurate but faster methods. Users can create and explore their own combinations via the flag **-reg\_method** that makes T-Coffee use a set of built-in aligners.

```
ktup_msa
blastp_msa
clustalo_msa
clustaloNF_msa
clustalw2_msa
clustalw_msa
```

```
uppNF_msa
upp_msa
msa_msa
dca_msa
mafftsparecore_msa
maffttest_msa
mafft_msa
...
```

The **-reg\_nseq** flag is the only free parameter. This parameter defines the maximum number of sequences in the subMSAs. It allows to use more accurate methods that can only handle a limited number of sequences. There is also a tradeoff between the size of the subMSAs and the CPU time. Based on results in [3] we have defined this size to 1.000 sequences as a default value.

The optimal value may change somewhat depending on the guide-tree and the alignment methods as well as the type of sequences to be aligned.

---

## 4 Notes

1. One of the possible issues of this method occurs in step #1, where the guide tree computation is required. Some of the classic methods are not able to handle large amounts of sequences and this fails at delivering a guide tree. Yet, provided a guide tree is available, most methods can be deployed using the regressive mode of T-Coffee.
2. An important contribution to scalability results from the way the final MSA is assembled. Because it results from the combination of sub-MSAs containing a common sequence, the gaps do not need to be stored in memory and they can be kept as counters and eventually written on disc when the computation is finished. This allows without any swapping the computation of models larger than the available RAM.
3. It is worth mentioning that the regressive implementation of T-Coffee explicitly avoids aligning non-homologous indels (i.e. indels having occurred independently according to the guide tree). These indels are concatenated rather than aligned. This process has two consequences: it can result in much larger MSAs and it means that given two alternative guide trees (i.e. mBed and PartTree), the one producing the MSA containing the smallest number of gaps is probably the most accurate.

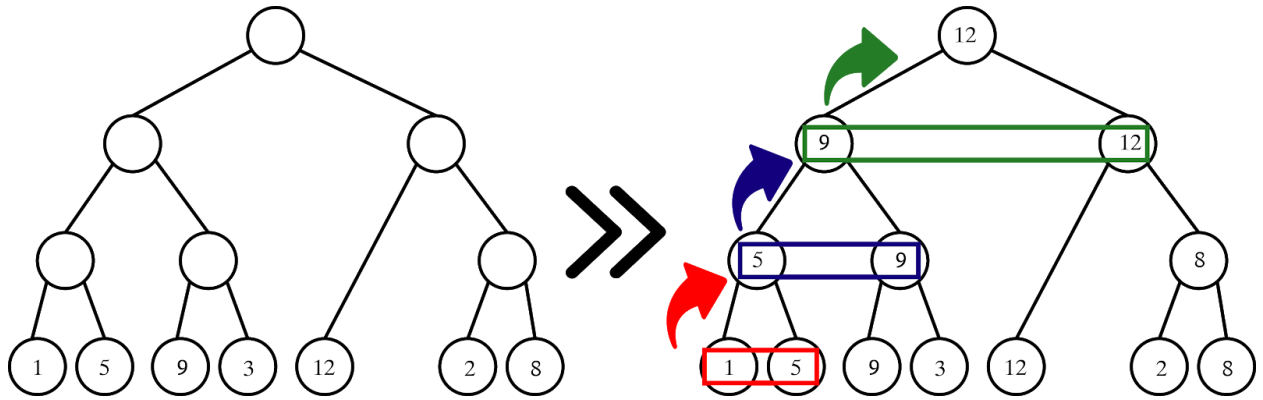
---

**Acknowledgment.** We acknowledge Des Higgins and Olivier Gascuel for useful discussions and feedback.

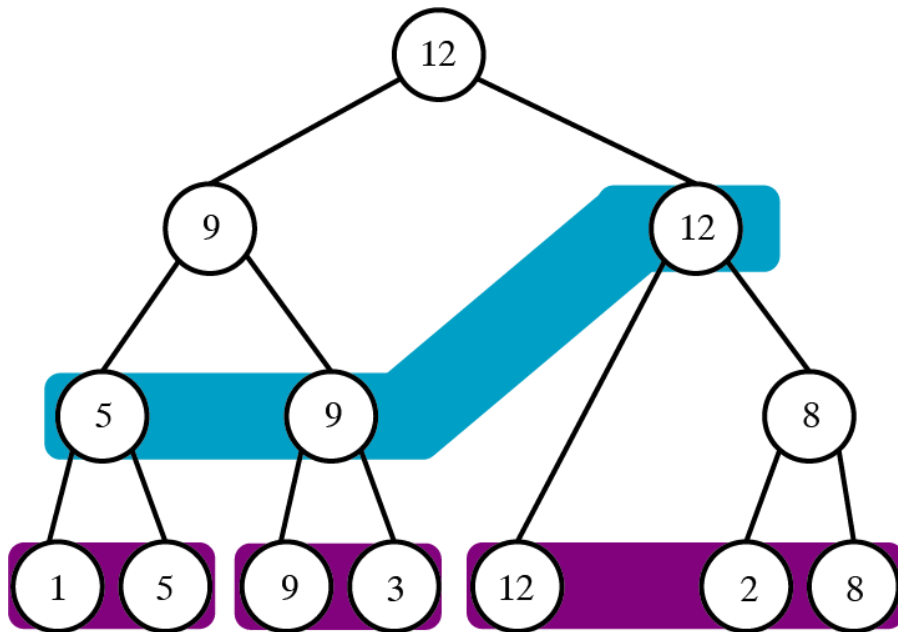
## References

1. Hogeweg P, Hesper B (1984) The alignment of sets of sequences and the construction of phyletic trees: an integrated method. *J Mol Evol* 20:175–186. <https://doi.org/10.1007/bf02257378>
2. Garriga E, Di Tommaso P, Magis C, et al Large multiple sequence alignments with a root-to-leaf regressive method. *Nat Biotechnol* (in press)
3. Sievers F, Wilm A, Dineen D, et al (2011) Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Mol Syst Biol* 7:539. <https://doi.org/10.1038/msb.2011.75>
4. Finn RD, Bateman A, Clements J, et al (2014) Pfam: the protein families database. *Nucleic Acids Res* 42:D222–30. <https://doi.org/10.1093/nar/gkt1223>
5. Notredame C, Higgins DG, Heringa J (2000) T-Coffee: A novel method for fast and accurate multiple sequence alignment. *J Mol Biol* 302:205–217. <https://doi.org/10.1006/jmbi.2000.4042>
6. Blackshields G, Sievers F, Shi W, et al (2010) Sequence embedding for fast construction of guide trees for multiple sequence alignment. *Algorithms Mol Biol* 5:21. <https://doi.org/10.1186/1748-7188-5-21>
7. Katoh K, Toh H (2007) PartTree: an algorithm to build an approximate tree from a large number of unaligned sequences. *Bioinformatics* 23:372–374. <https://doi.org/10.1093/bioinformatics/btl592>

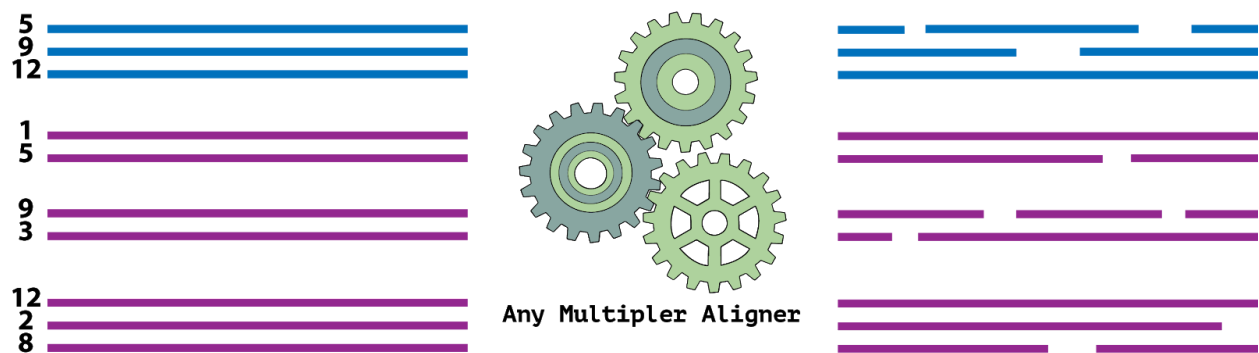
**Figures**



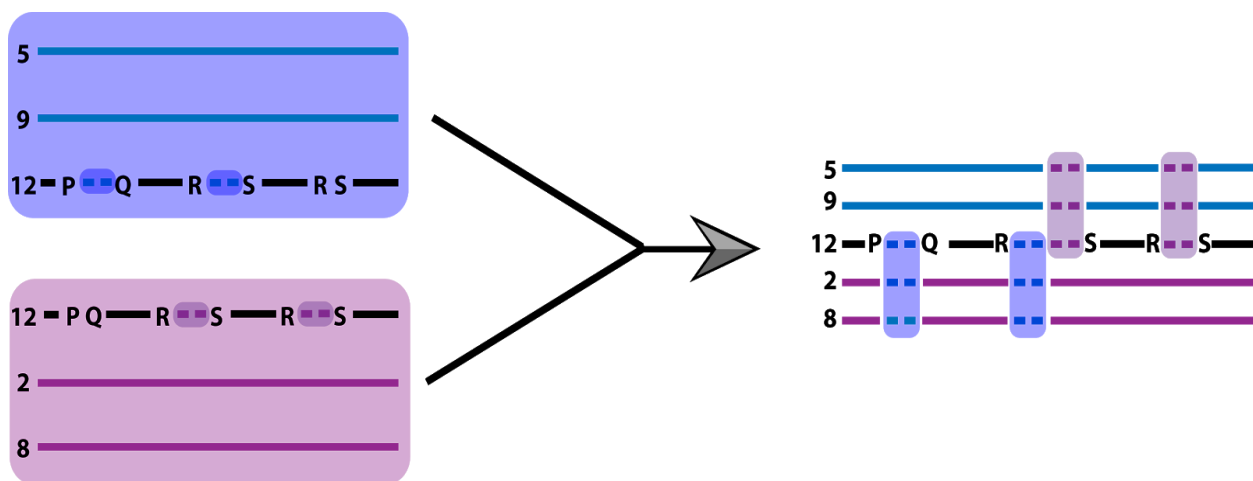
**Fig1.** From the naked guide-tree, the algorithm starts from the leaf until the root labeling the internal nodes with the longest sequence of the children.



**Fig2.** From the root of the guide-tree,  $N$  sequences are collected., ( $N=3$  in this example.) This is repeated recursively on the remaining nodes until all the leaf nodes are included in one of the subMSAs.



**Fig3.** Once all the small groups are defined, it is possible to generate the subMSAs in a parallel way using any third-party alignment software.



**Fig4.** Parent and children subMSAs are merged using the common sequence (12) projecting indels from parent to child and from child to parent.