Algorithms in Bioinformatics
Project 4 - Tree Comparison

Ditte Torlyn (201408508)
Rune Wind (201608439)
Astrid Christiansen (201404423)

April 2020

Introduction

In this project, we compare evolutionary trees constructed using Neighbor Joining (NJ) methods on different datasets. We have implemented an efficient algorithm for computing the Robinson-Foulds Distance (RF-distance) between two trees and we have used this implementation in an experiment. For the experiment, we have constructed trees from 2×395 sequences by first using the alignment methods Clustal Omega, Kalign and MUSCLE to make multiple alignments of the sequences and then we have used QuickTree and RapidNJ to construct evolutionary trees from the alignments.

Implementation

In out implementation, we read the trees in newick format using the BioPython Phylo package. We have implemented Day's Algorithm for computing the RF-distance, as explained in class. Our program is called `rfdist` and works as we expect on the cases we have used to test our program. We have tested out program on the two trees with RF-distance 8 in `testdata.zip` and also on the two trees T_1 and T_2 with RF-distance 3 from the slides `Tree_Comparison.pdf`.

How to run our program

Our program file is called `rfdist.py`. Running this program will output the RF-distance of two input trees. To run our programs from the command line, type:

```
python rfdist.py tree1.new tree2.new
```

where `tree1.new` and `tree2.new` are files specifying the two trees we want to compare in newick format.

The output will then look like this:

```
The RF-distance of tree1.new and tree2.new is X
```

Where to find our alignments and trees

Our alignments in STOCKHOLM format can be found via this link:

https://github.com/RuneWind/AiB/tree/master/Project4/stockholm

There is a folder containing the alignments of the non-permuted sequences and a folder containing the alignments of the permuted sequences.

Our trees in newick format can be found via this link:

https://github.com/RuneWind/AiB/tree/master/Project4/trees

Again, there is a folder containing the trees constructed from the non-permuted sequences and a folder containing the trees constructed from the permuted sequences.

Experiments

Experiment 1

We make 3 multiple alignments of the 395 sequences found in `patbase_aibtas.fasta` using the alignment methods Clustal Omega, Kalign and MUSCLE. We then build NJ-trees using QuickTree and RapidNJ on these alignments. This gives us 6 trees. We annotate the trees by the following indices:

	Quick Tree	RapidNJ
Clustal Omega	1	4
Kalign	2	5
MUSCLE	3	6

On each pair of these 6 trees, we use our program `rfdist` to calculate the RF-distance between them. Our results are shown here, where the numbers in the first row and first column are the tree indices, as indicated above:

	1	**2**	**3**	**4**	**5**	**6**
1		202	194	250	246	256
2			220	262	172	284
3				296	270	200
4					266	242
5						264
6						

We notice that the smallest numbers in the table seem to be the result of calculating the RF-distance between two trees that have something in common, i.e. the alignment method or the tree construction method. This makes sense and would be expected.

Experiment 2

We now make multiple alignments of the 395 permuted sequences found in `patbase_aibtas_permuted.fasta`, using the same alignments methods as before. We redo Experiment 1 with these alignments. We annotate the trees in the same manner as before and get the following RF-distances:

	1	**2**	**3**	**4**	**5**	**6**
1		184	174	258	254	233
2			232	290	152	266
3				292	268	214
4					284	252
5						284
6						

Again, it seems that the smallest numbers in the table are the RF-distances between trees with similar alignment methods or tree construction methods.

Experiment 3

Now we compare tree number i from Experiment 1 with tree number i from Experiment 2 for $i = 1, ..., 6$, by calculating the RF-distance between them using our program `rfdist`. The results are:

Tree indices	RF-dist
1	100
2	148
3	162
4	180
5	196
6	228

Based on these results, it seems that QuickTree makes more similar trees when we run it on the alignments of the non-permuted and the permuted sequences compared to RapidNJ.