**Comparative Genomics Practical 3**

Phylogenetic Reconstruction

Group 6 Tianlin He Xueqing Wang

**SUMMARY**

In this practical we undertook all the way from the whole DNA sequences of four prokaryote genomes to the reconstruction of phylogenetic tree. The evolutionary relationship was calculated based on 16S ribosome RNA gene sequences. We firstly extracted the 16S gene fractions by running BLAST using E.coli 16S RNA gene against these genome sequences. All high-scored hitting fractions are recorded in the XML file acquired out of BLAST, with the e-value from high to low. We parsed each XML file got out of different sequences, separately, and obtained the best hitting partitions from each genome, considering them as the 16S RNA gene and applying them as the raw material to build the phylogenetic tree. We aligned the collected sequence fractions and built the phylogenetic tree by belvu with the help of netwick tree viewer. After these we further did bootstrapping as well as sanity check to access the reliability of the tree, taking the impact of recombination into account.

**ACTIVITIES**

**DNA Tree Reconstruction**

1. Format a blast database for your genomes, so you can search locally.

a. What do the parameters mean?

- “makeblastdb”: make a blast database.

- “-in”: input fasta file

- “-dbtype nucl”: type of database is nucleotide

b. Run the program for each nucleotide dataset

makeblastdb -in <inputfile.fa> -dbtype nucl

2. It might be the easiest (but not necessary) to have all your genomes (genes) in one file, so you can make a single database for them (as long as entries in each file are labeled with genome they belong to).

cat genome\_0 genome\_1 ... > genomes\_all

3. Use BLAST to query the database for the 16S rRNA file. Find the best hit in each genome as “actual” 16S rRNA and gather them as entries in a fasta file. Extract 16S sequence from BLAST results that you run against whole genome.

1. What other parameters do you need and what do they mean?

Blastn: it is a nucleotide against nucleotide blasting

-outfmt 5: the output is in XML format

-query: it assigns the query sequence

-db: it assigns the database that we search inside

-out: it assigns the name of the output file. (e.g. output.txt)

b. Where do you find and what you need in the output ?

blastn -outfmt 5 -query <query file.fa> -db <database file.fa> -out <output file>

The outputfile (e.g. output.txt) it is in XML format. In this file, several high-scoring hits obtained from blasting of the query sequence against the database.

The command line we used is:

blastn -outfmt 5 -query 16S\_ecoli.fasta -db 03.fa.txt -out 03-16sRNA.txt

blastn -outfmt 5 -query 16S\_ecoli.fasta -db 08fa.txt -out 08-16sRNA.txt

blastn -outfmt 5 -query 16S\_ecoli.fasta -db 09.fa.txt -out 09-16sRNA.txt

blastn -outfmt 5 -query 16S\_ecoli.fasta -db 18.fa.txt -out 18-16sRNA.txt

**Parsing options A or B**

A. Write a simple biopython blast parser using the ​ NCBIXML module

○ To obtain XML output from BLAST use the -m 5 parameter.

B. If this is not possible, we can provide you with a existing script to parse the blast

output (​ blastResultParser.py​ ). However, you should answer following questions

in addition.

○ How does the script choose the single best blast hit in each genome in the database?

import sys

import re

from Bio.Blast import NCBIXML

blastOutputXMLFile = sys.argv [1] it takes up our XML file from the previous step as input from the command line we typed

blastOutputXMLHandle = open (blastOutputXMLFile)

listOfBlastRecords = NCBIXML.parse (blastOutputXMLHandle)

for aSingleBlastRecord in listOfBlastRecords: it runs a “for” loop inside the XML file

for i in range (len (aSingleBlastRecord.alignments)):

description = aSingleBlastRecord.descriptions [i]

alignment = aSingleBlastRecord.alignments [i]

title = re.compile ("gnl\|BL\_ORD\_ID\|\d\* ").sub ("", description.title) Here, re.compile search words with the specific format as gnl|BL\_ORD\_ID|, so that the name of hit after this format can be extracted as use as title

print ">" + title

print alignment.hsps [0].sbjct

In a NCBIXML file, hit sequences (HSPs) are listed in the order of ascending E-value (i.e. sequences with higher score in the front of the list). In the biopython script we are provided, it writes the sequence with highest score (i.e. lowest E-value) from the input file to output file, by using the code:

sequence = alignment.hsps[0].sbjct

Here hsps[0] means the first sequence in the list of high-scoring pairs (HSPs).

○ To what does a blast record correspond?

A blast record collects a number of high-scoring hits.

It corresponds to all the blast that we have done, given that we have one single XML file which contains all hits records obtained from blasting the query sequence against all 4 genomes 03, 08, 09,18.

○ What do we assume about the BLAST XML output?

We assume that XML output always list the hits in ascending order of E-value. Therefore, hsps[0] gives us the best hit with lowest E-value.

○ What does this script output?

This script outputs the best hit for each genome, which is the 16S RNA in those 4 prokaryotes.

**Align Your Sequences**

4. You can use KALIGN on the resulting sequence file to make a multiple alignment

of the homologs identified in the previous step.

a. What parameters for gap penalties exist, and would any of them make

sense to apply?

kalign <infile> <outfile>

-The parameters existing for gap penalties are as follows:

-s, which adds gap open penalty;

-e, which adds gap extension penalty

-t, which adds terminal gap penalties

-a, which adds parameter increasing gap penalties depending on the number of existing gaps

We didn’t specify these parameters in our practice, since the program would detect the type of our input sequences and assign relatively reasonable gap penalty numbers, shown as follows:

217.00000000 gap open penalty

39.40000153 gap extension

292.60000610 terminal gap penalty

283.00000000 bonus

Before we do kalign we didn’t know exactly how the alignment would be, so it is a better idea to follow the default outcomes.

**Perform Tree Reconstruction**

5. Reconstruct using more than one method, ie distance based and parsimony.

a. What are the advantages of each? What influenced your choice?

-Distance based method transforms sequence alignment data into pairwise distance matrix, and then use the values in matrix during to reconstruct the tree instead of comparing the alignments. Thus, distance-based methods are more accurate and can be used to calculate out the tree even if the sequence similarity is not very high.

-Parsimony method, or maximum parsimony, refers to the criterion that the simplest (or shortest) tree describing the evolutionary relationship of the species is selected. In other words, the best tree would minimize the number of evolutionary events undergoing during the whole process. To achieve the simplest phylogenetic tree, all the possibilities need to be tested and the one with the least evolutionary procedures would be chosen. For a small number of taxa it is possible to do detailed searching, with every possible tree is scored and the best one selected. For more taxa, it would be better to use heuristicsearching. Besides, it would be better to fulfill this method when the sequences are quite similar to each other.

-Based on above considerations, the number of taxa we have, the extent of similarity among them, the calculation power we have, the accuracy and running time we expect and so on all influence the final choice.

6. Learn about the various options of Belvu and how to create a tree from an

alignment.

a. Which distance correction method does Belvu use?

If not specified, Belvu uses Scoredist distance correction by default.

b. How would you request a different distance correction.

Instead of directly using “belvu -o tree kalign0510.txt”, add some additional parameters, such that:

“belvu -T j -o tree kalign0510.txt” uses Jukes-Cantor distance correction.

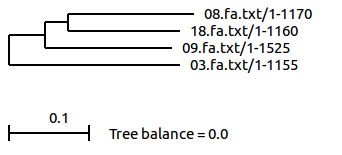
“belvu -T s -o tree kalign0510.txt” uses Storm & Sonnhammer distance correction

“belvu -T k -o tree kalign0510.txt” uses Kimura distance correction.

7. An alternative way to look at the tree is to use tree newick viewer

http://etetoolkit.org/treeview/

The tree we got out of command “/common/courses/comparative\_genomics/belvu\_Ubuntu\_12.04.3\_64bit kalign0510.txt” is as follows:



Sequence Bootstrapping

8. Your result is might merely be an artifact of sampling? To get a significance measure for the tree you can use bootstrapping. Explain what bootstrapping is.

Bootstrapping can be interpreted as an examining method to test the reliability of the splits shown in the constructed tree. It starts by generating a given number of pseudoreplicate datasets, which are acquired by repeating the tree construction on the same dataset with different sampling every time. The resulting tree we get then become a “consensus” outcome, showing the agreement among the samplings, with the presentation of percentage getting the same split within these samplings on the diagram. The higher the percentage is, the more pseudoreplicate datasets getting into the same split, the more reliable the split is. However, a high bootstrapping value does not necessarily mean that the built branch is correct.

9. Construct a consensus tree with bootstrap support values from your alignment.

a. What does N mean?

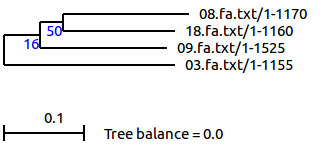
-The “N” after -b refers to the number of bootstrap samples.

b. What N do you choose and what consequences do that choice have?

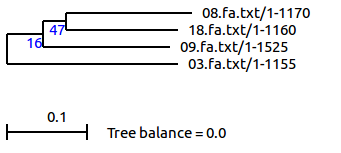
belvu -b N

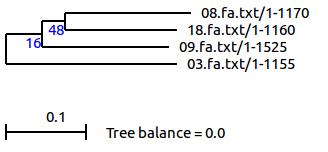
-I chose several bootstrap sample numbers, the outcome phylogenetic trees remain the same, just that the values for each branches are different. Basically, the higher bootstrap scores we have, the more qualified (reliable) the splits in the tree are.

The tree we got out of bootstrapping with 100 samples is as follows:



The tree we got out of bootstrapping with 1000 samples is as follows:

The tree we got out of bootstrapping with 10000 samples is as follows:



**Sanity Check**

10. To test the effects of poor sampling (or database errors, or recombination), make a copy of your original alignment.

11. For each sequence, paste a copy of itself after it (ie ATTCGT->ATTCGTATTCGT).

12. For one or more pairs of sequences, swap the second half of the sequence.

a. This would match the situation where half of the sequence has been mislabelled, or where half of the sequence has evolved in a way that involves recombination.

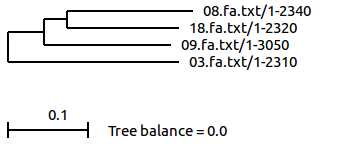
13. Perform tree reconstruction as before on these sequences.

a. What tree do you get?

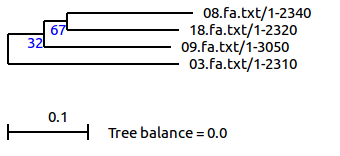
14. Perform bootstrap-based analysis.

a. Can you deduce from the bootstrap scores which taxa you shuffled the second half of the sequence for?

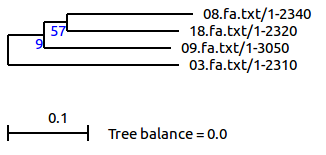
-The tree I got out of doubling every sequence and pasting after themselves is as follows:



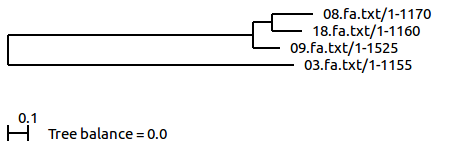
-The tree I got out of doubling every sequence, pasting after themselves and bootstrapping with 100 samples is as follows:

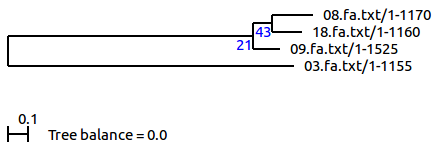


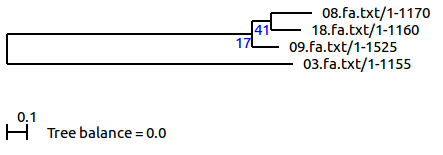
-The tree I got out of doubling every sequence, pasting after themselves and bootstrapping with 1000 samples is as follows:



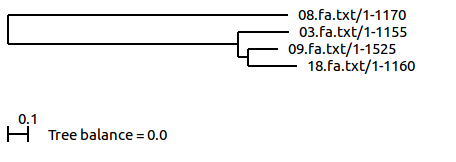
-The tree got out of swapping the first and second half of sequence 1 (from 03.fa.txt) is as follows:

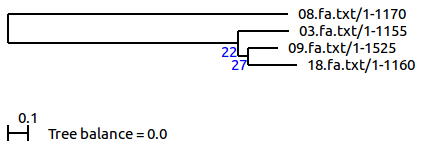
-The tree got out of swapping the first and second half of sequence 1 (from 03.fa.txt) with bootstrap number 100 is as follows:

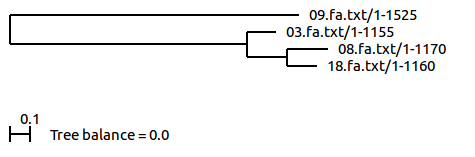
-The tree got out of swapping the first and second half of sequence 1 (from 03.fa.txt) with bootstrap number 1000 is as follows:



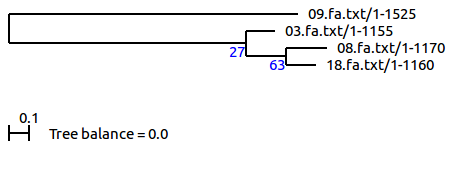
-The tree got out of swapping the first and second half of sequence 2 (from 08.fa.txt) is as follows:

-The tree got out of swapping the first and second half of sequence 2 (from 08.fa.txt) with bootstrap number 100 is as follows:

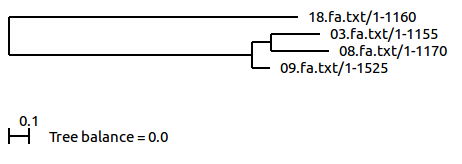
-The tree got out of swapping the first and second half of sequence 3 (from 09.fa.txt) is as follows:



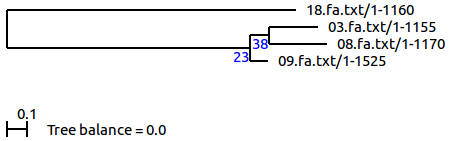
-The tree got out of swapping the first and second half of sequence 3 (from 09.fa.txt) with bootstrap number 100 is as follows:



-The tree got out of swapping the first and second half of sequence 4 (from 18.fa.txt) is as follows:



-The tree got out of swapping the first and second half of sequence 4 (from 18.fa.txt) with bootstrap number 100 is as follows:



After double the sequence alignment, the phylogenetic tree remains the same, but the bootstrapping score increases; after shuffling the second half of the sequence, the tree alters dramatically. Basically, the one which is shuffled branches out separately in the tree; when the species 03, 08 or 09 is shuffled, the remaining ones show similar evolutional relations as before; when species 18 is shuffled, the remaining ones show different relationships compared to former situations. Perhaps we can speculate that species 18 sequences may have recombination.

**DISCUSSION**

We followed the normal procedures to explore the evolutional relationship among four prokaryotes, E.coli, Synechocystis, D.turgidum, and B.thetaiotaomicron, as identified in the previous practical, and represented this relationship by phylogenetic tree. The tree was established by the alignment of 16S RNA gene, due to the their relatively slow evolution rates. To access the reliability of the tree, we performed bootstrapping, and further counted for the recombination effect in sanity check.

After bootstrapping we get roughly similar trees, which is because that we apply the same tree reconstructing methods (all correspond to the setting of belvu by default) when generating the pseudoreplicate groups; we don’t change the tree-building principles and then built consensus tree, which would more probably give out different results (i.e., some splits vanish while others appear). The difference before and after fulfilling bootstrapping is just that the tested result (the later one) presenting out the supporting percentage of each split in all the samplings in value, giving us an intuitive idea of the reliability of these splits. After a larger amount of sampling, the splits in the outcome tree still have low score (less than 60%). However, the hits we got are actually accurate: all the e-values of BLAST tests are less than 10^(-140); 16S RNA gene is also an ideal material for prokaryote phylogenetic tree building. The low bootstrapping value may due to the distance measuring method or the weak relation among these four species.

We also fulfilled the sanity check to examine whether there are poor sampling influence that affect the phylogenetic tree building. After shuffling the second half of the sequences separately, one of the species introduced dramatically alteration for the tree, suggesting that it might have recombination circumstances.