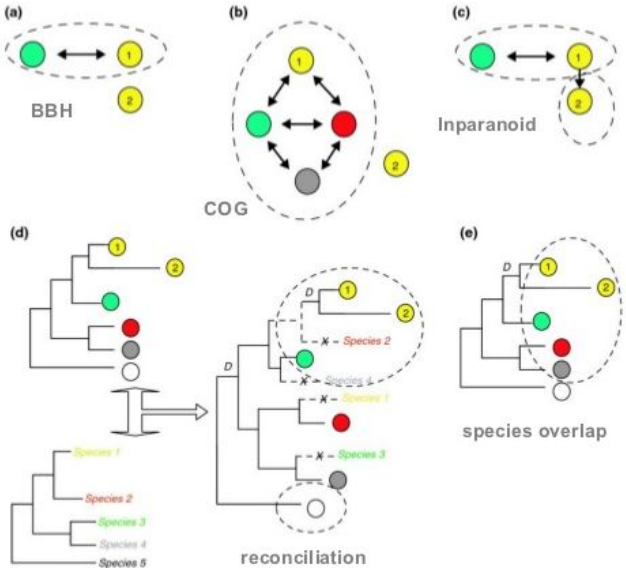
Orthology and paralogy prediction lab

Overview methodologies



Gabaldon, 2008

Approaches we will see to predict orthology and paralogy:

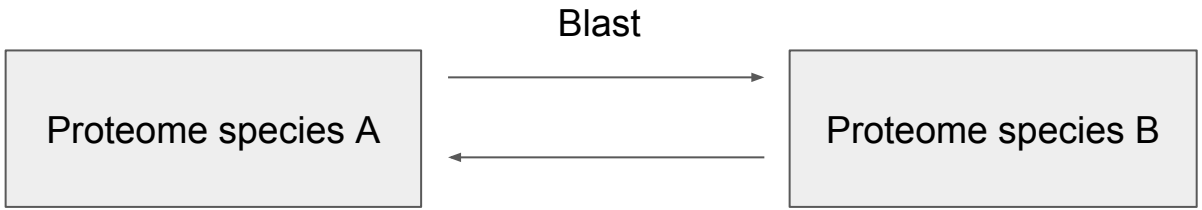
- Best Reciprocal Hits
- Inparanoid
- MCL
- PorthoMCL
- Phylogenetic based orthology / paralogy predictions (ETE3)
- Website pre-calculated predictions

Best Reciprocal Hits:

It’s the most simple of the methods and is based on the idea that if two sequences from different species are each others best hit then they are likely orthologs.

Advantages: It’s fast and simple to understand.

Disadvantages: Will ignore complex evolutionary scenarios. Will be confused by differential gene loss.



How to run a local blast

Fasta file with the queries (can be one or many)

Fasta file with the targets

↓ makeblastdb -in target_file.fasta -dbtype prot

Format the targets file into a blast database

↓ blastp -query query_file.fasta -db target_file.fasta -evalue 1e-5 -outfmt 6 -out results.blast

Execute the blast search and output it into a file

How does the blast results file look like:

```
# BLASTP 2.6.0+
# Query: PENCH_0037_10162
# Database: ASPCL.fasta
# Fields: query acc.ver, subject acc.ver, % identity, alignment length, mismatches, gap opens, q. start, q. end, s. start, s. end, evalue, bit score
# 4 hits found
PENCH_0037_10162        ASPCL_0077_01917        59.836  122     45      3       1       121     1       119     1.69e-45        138
PENCH_0037_10162        ASPCL_0089_08902        57.258  124     38      4       1       121     1       112     8.06e-39        121
PENCH_0037_10162        ASPCL_0081_03871        52.041  98      38      2       20      117     39      127     1.88e-30        100
PENCH_0037_10162        ASPCL_0089_08749        40.816  98      47      3       19      116     390     476     4.45e-17        69.7
```

(this is with option -outfmt 7, -outfmt 6 is the same but without the headers)

Exercice 1: Get the Best Reciprocal Hits between two subsets of proteins of *Aspergillus clavatus* and *Penicillium chrysogenum*.

A.- Download the orthology_lab.tar.gz file from the course website

B.- Uncompress it in your Desktop (tar -zxvf orthology_lab.tar.gz)

C.- Go to the main folder and within this to the exercise 1 folder (cd orthology_lab/exercise1)

D.- Build the two blast databases that we need, one for ASPCL.fasta and another one for PENCH.fasta

E.- Now run a blastp search using ASPCL as query and PENCH as target. Be sure to name the results files in such a way that you know which was the query and which was the target in each case. (i.e. ASPCL_2_PENCH.blast)

F.- Repeat step E but the other way around. (suggested output name: PENCH_2_ASPCL.blast)

G.- Run the python script get_BRH.py using the options: -s1 ASPCL.fasta -s2 PENCH.fasta -h1 ASPCL_2_PENCH.blast -h2 PENCH_2_ASPCL.blast -t results

The result will be two files, one named results.BRH.txt and another one named results.unpaired.txt

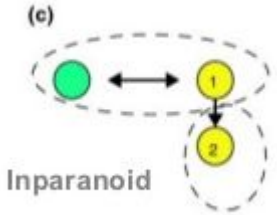
The first one contains all the best reciprocal hits found between the two species whereas the second contains a list of proteins that were left unpairs.

H.- Now take one of the pairs in the file results.BRH.txt and go back to the blast results files. Using grep, check out that the chosen pair is indeed the best reciprocal hit of each other.

I.- Protein ASPCL_0080_03602 does not have a best reciprocal hit. Looking at the blast files, can you figure out why not?

Inparanoid:

Inparanoid adds a layer of complexity to the best reciprocal hits approach. In BRH it is assumed that all proteins that have a blast hit against another protein but it is not reciprocated are paralogs. Inparanoid makes the distinction between inparalogs and outparalogs. Given one member of a pair of orthologs, if there is a sequence in the same species that has a higher blast score than the corresponding ortholog, inparanoid assumes that this sequence is an inparalog. Additionally, inparanoid allows the use of a outgroup that will help distinguish between orthologs, inparalogs and outparalogs.



How to run Inparanoid

Inparanoid is an old program and as such it is a bit difficult to get it ready to run. Follow the next steps:

A.- Go to the file exercise2, there you will see three files with sequences.

B.- Download inparanoid from this link: genome.crg.es/~mmarcet/inparanoid_4.1.tar.gz and move it to your working folder.

C.- Uncompress it: tar -zxvf inparanoid_4.1.tar.gz

D.- mv the inparanoid folder out of the tmp folder: mv tmp/tmp3Ol0q8/inparanoid_4.1/ .

E.- Go to the inparanoid folder and open the file called inparanoid.pl (cd inparanoid_4.1/ followed by pluma inparanoid.pl)

inparanoid.pl

```
38
39 ###############################################################################
40 # Set following variables:                                                    #
41 ###############################################################################
42
43 # What do you want the program to do?                                        #
44 $run_blast = 1;   # Set to 1 if you don't have the 4 BLAST output files      #
45                   # Requires 'blastall', 'formatdb' (NCBI BLAST2)            #
46                   # and parser 'blast_parser.pl'                             #
47 $blast_two_passes = 1;  # Set to 1 to run 2-pass strategy                     #
48                   # (strongly recommended, but slower)                       #
49 $run_inparanoid = 1;
50 $use_bootstrap = 1;# Use bootstrapping to estimate the confidence of orthologs#
51                    # Needs additional programs 'seqstat' and 'blast2faa.pl'
52 $use_outgroup = 0; # Use proteins from the third genome as an outgroup       #
53                    # Reject best-best hit if outgroup sequence is MORE      #
54                    # similar to one of the sequences                        #
55                    # (by more than $outgroup_cutoff bits)                   #
56
57 # Define location of files and programs:
58 #$blastall = "blastall -VT"; #Remove -VT for blast version 2.2.12 or earlier
59 $blastall = "blastall";  #Add -aN to use N processors
60 $formatdb = "formatdb";
61 $seqstat = "inparanoid_4.1/seqstat";
62 $blastParser = "inparanoid_4.1/blast_parser.pl";
63
64 $matrix = "BLOSUM62"; # Reasonable default for comparison of eukaryotes.
65 #$matrix = "BLOSUM45"; #(for prokaryotes),
66 #$matrix = "BLOSUM80"; #(orthologs within metazoa),
67 #$matrix = "PAM70";
68 #$matrix = "PAM30";
69
70 # Output options:                                                            #
71 $output = 1;       # table_stats-format output                             #
72 $table = 1;        # Print tab-delimited table of orthologs to file "table.txt" #
73                    # Each orthologous group with all inparalogs is on one line #
74 $mysql_table = 1; # Print out sql tables for the web server                  #
75                    # Each inparalog is on separate line                     #
76 $html = 1;         # HTML-format output                                     #
```

How to run inparanoid.

Inparanoid is quite old and still depends on the old blast program, so make sure you have formatdb and blastall installed in your computer.

F.- We will need to adjust the following parameters:

In line 61 and 62 change the paths to the inparanoid_4.1 folder as seen in the image.

Once done, save the file and close it.
G.- Copy the BLOSUM62 file to your working folder (cp BLOSUM62 ../) and return to your working folder (cd ../)

H.- Now we're ready to run inparanoid. To execute it simply type.

```
perl inparanoid_4.1/inparanoid.pl ASPCL.fasta PENCH.fasta
```

This will result in numerous files. The main tables of interest to us are table.ASPCL.fasta-PENCH.fasta which is a list of the orthologous groups while the Output.ASPCL.fasta-PENCH.fasta gives a more detailed overview. Have a look at them and see if you see any superficial difference between these results and those obtained from the Best Reciprocal Hits analysis.

I.- Move the main results to a different folder so that they are not overwritten:

mkdir no_outgroup
mv -t no_outgroup/ table.ASPCL.fasta-PENCH.fasta orthologs.ASPCL.fasta-PENCH.fasta.html Output.ASPCL.fasta-PENCH.fasta

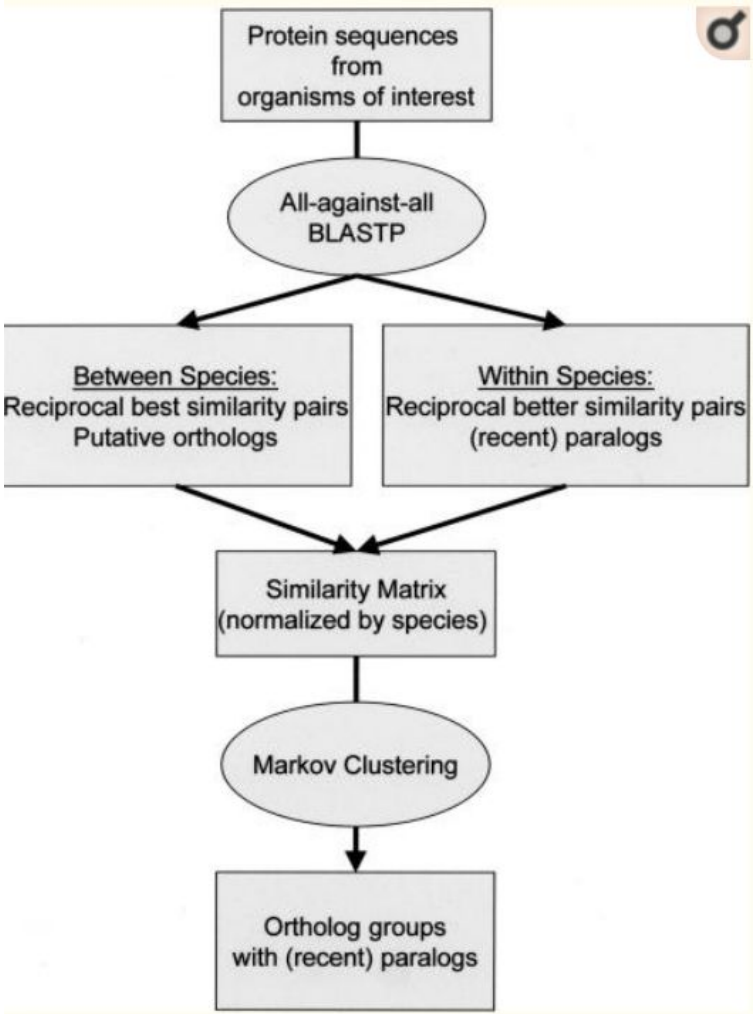
J.- Run inparanoid with an outgroup (PENMQ.fasta), you will need to modify the parameters in the script first so that in line 52 there's a 0 instead of a 1.

K.- Check out the new results and compare them with the previous ones, do you see any changes?

L.- Check out the file rejected_sequence.PENMQ.fasta

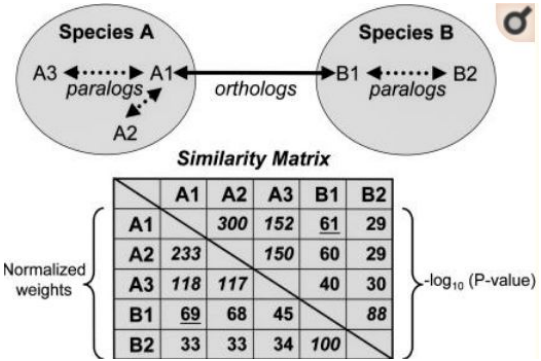
M.- Do you think running with an outgroup is better or worse?

N.- Search for protein ASPCL_0080_03602. Can you find it in any of the orthologous groups?



OrthoMCL

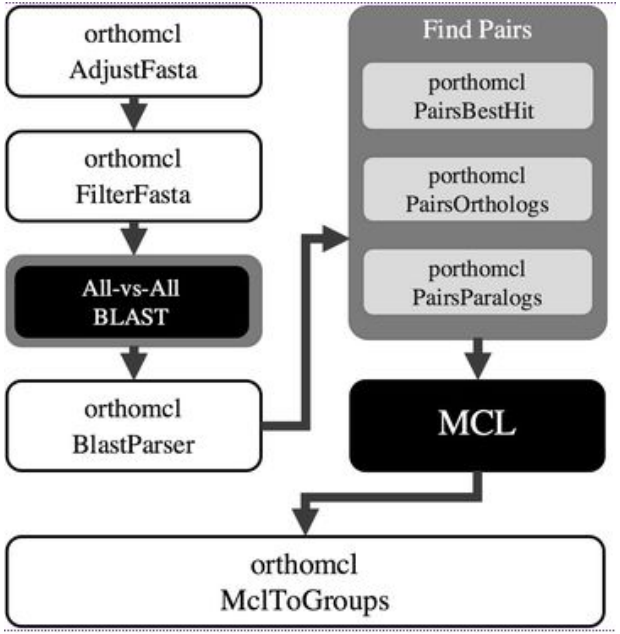
Another way to detect orthology and paralogy relationships, this time one a larger set of proteomes, is to use clustering based methods. The idea is to group sequences that are similar together and make those groups in such a way that they will only contain orthologs and sometimes inparalogs.



	A1	A2	A3	B1	B2
A1		300	152	61	29
A2	233		150	60	29
A3	118	117		40	30
B1	69	68	45		88
B2	33	33	34	100	

PorthoMCL

One of the most used of such methods is orthoMCL. PorthoMCL is just a simpler implementation of orthoMCL that runs without the need of a mysql database. It also has a protocol for running it with the use of a computation cluster which allows a user to run it on a large dataset.



How to run PorthoMCL:

A.- Go to the folder called exercise3. You will see that this time we have five proteome files.

B.- Create a folder called results that will be the container folder (mkdir results)

C.- Inside the container folder create a second folder called 0.input_faa (mkdir results/0.input_faa). Note that the name of this folder needs to be 0.input_faa because the automatic run of PorthoMCL will be expecting it.

D.- Move all the fasta sequences to this folder (mv *fasta results/0.input_faa)

E.- Finally execute PorthoMCL: /home/phylogenomics/software/PorthoMCL/porthomcl.sh -t 2 results/

The results of PorthoMCL are now found in the container folder (i.e. results/). Go there and have a look at the files.

F.- Search for the orthologs of protein ASPCL_0080_03602, is it congruent with previous results?

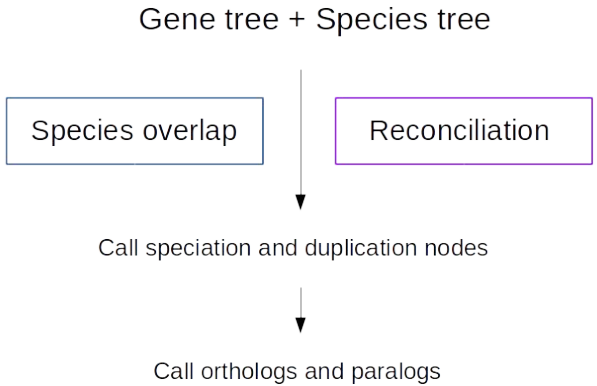
G.- What do you think are the drawbacks of PorthoMCL?

Orthology and paralogy predictions based on trees.

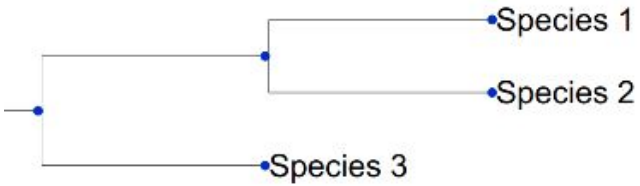
Orthology and paralogy predictions can be obtained directly from phylogenetic trees. The first step is to go through the tree and at each node decide whether we are in front of a duplication node or a speciation node. This can be done in several ways. The most used ones are:

1.- Reconcilliation: You need to have a species tree. It assumes that any inconsistency found between the gene tree and the species tree is explained by a duplication and loss scenario.

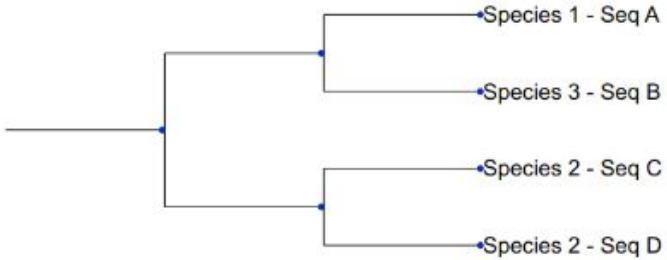
2.- Species overlap: It's based uniquely on the gene tree. It assumes a duplication is present when at either side of the node there is at least a pair of sequences belonging to the same species.



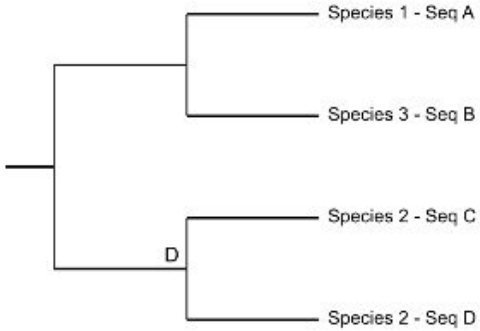
Species tree



Gene tree

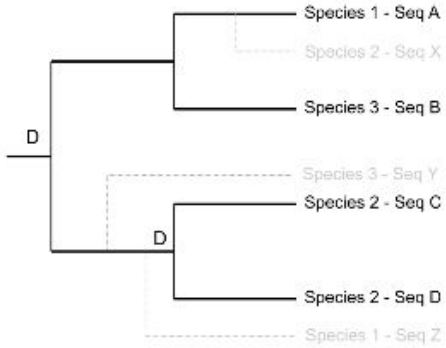


Species overlap

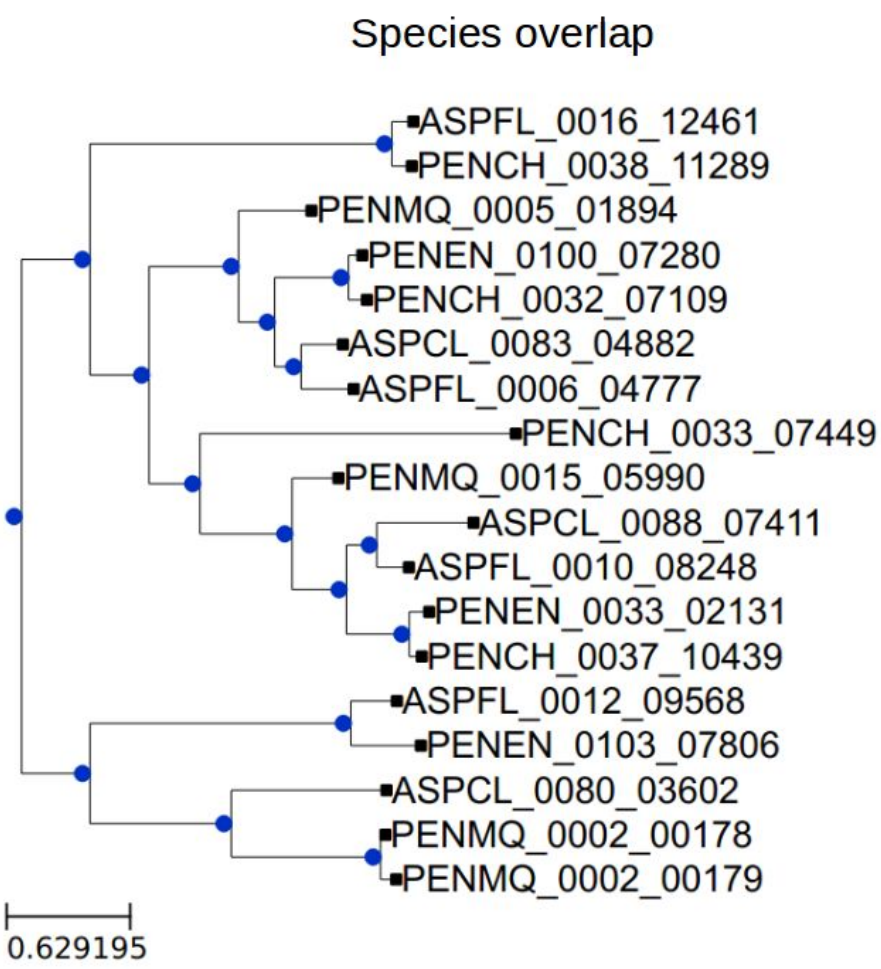


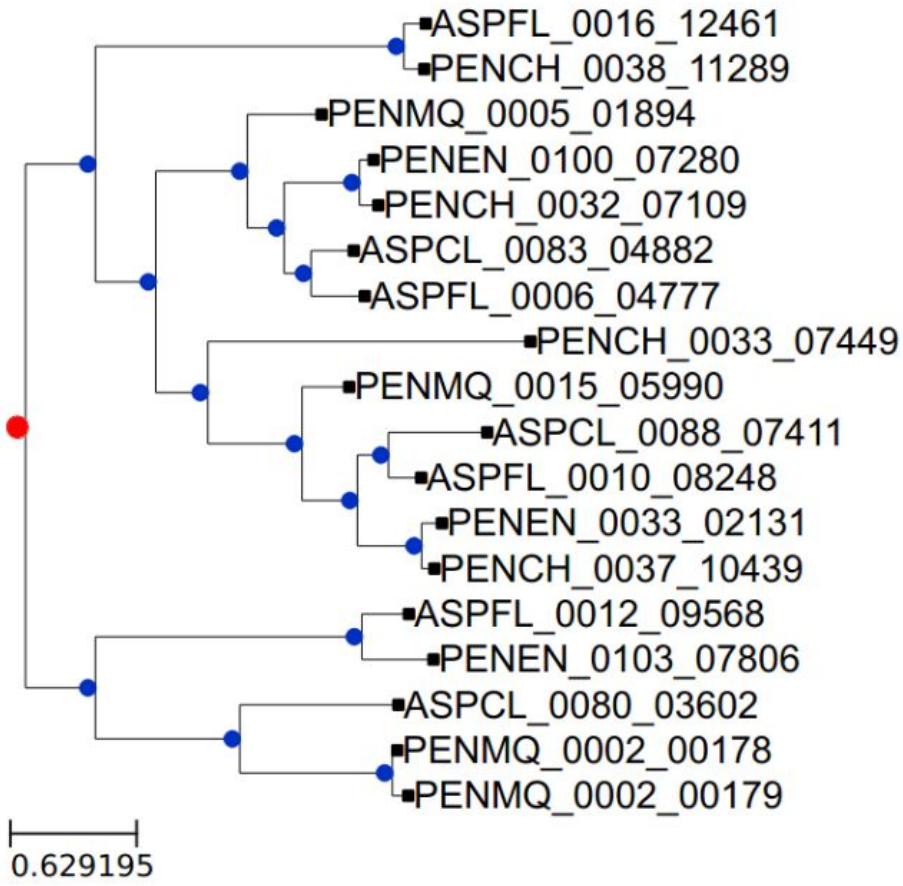
1 duplication and 0 losses

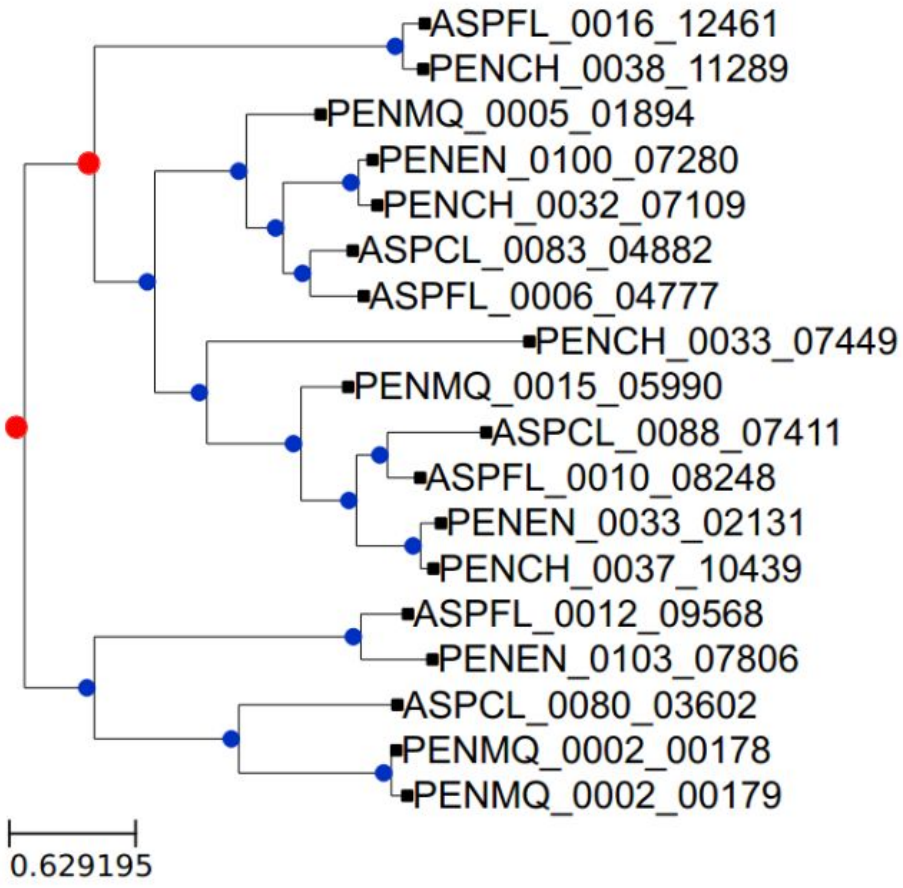
Reconciliation

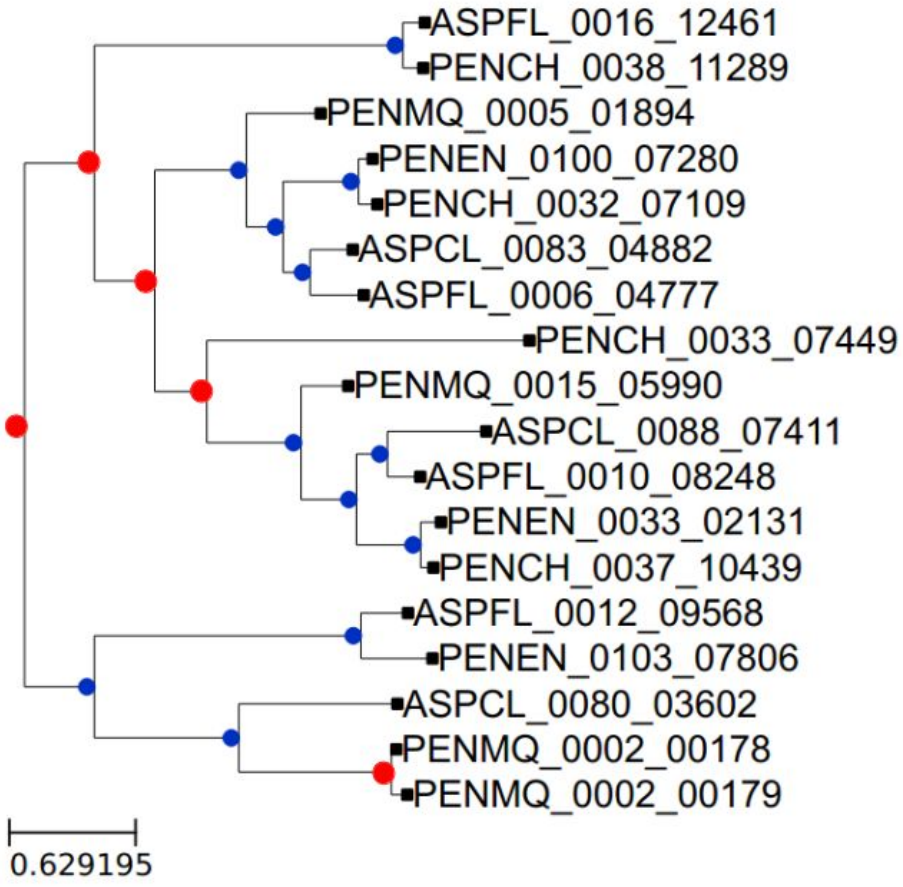


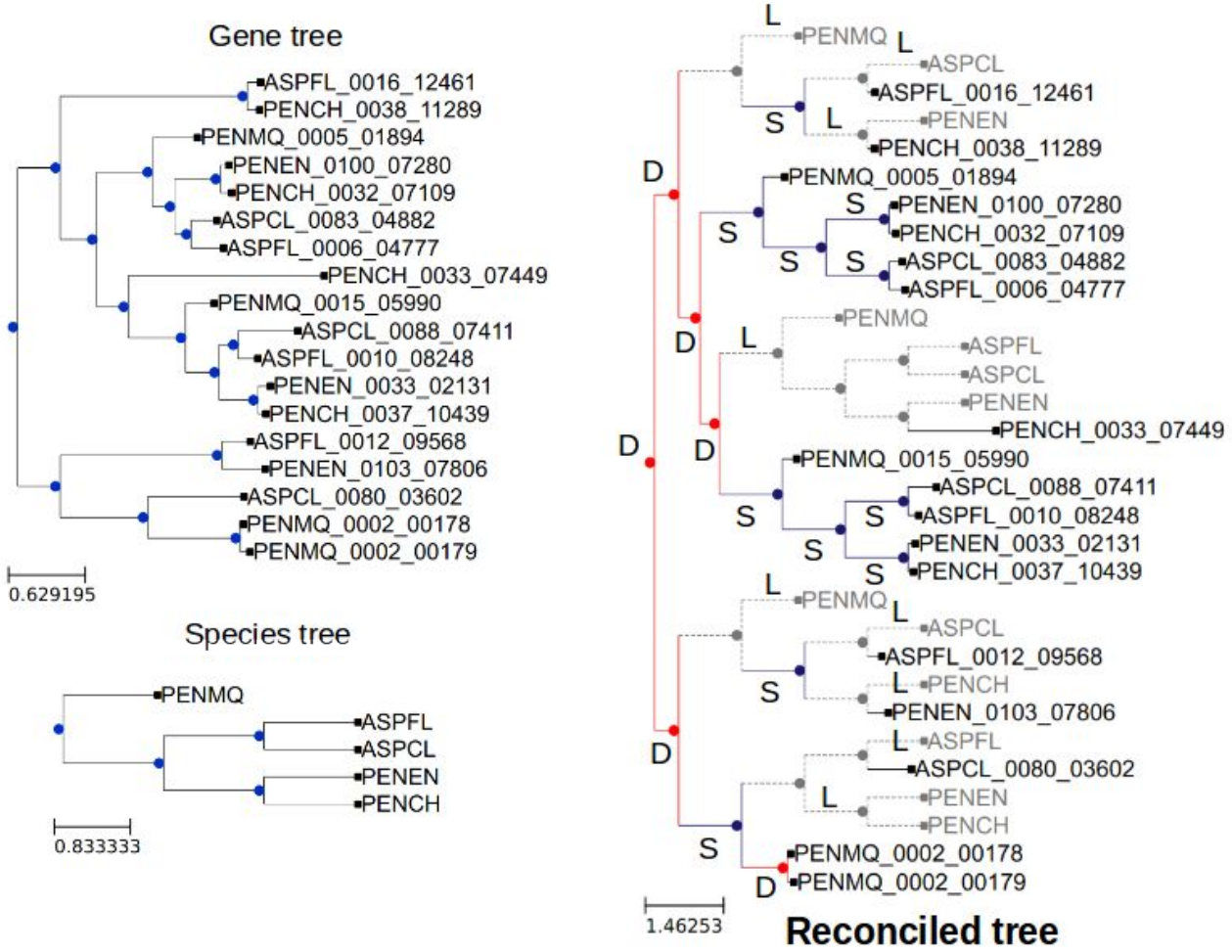
2 duplications and 3 losses

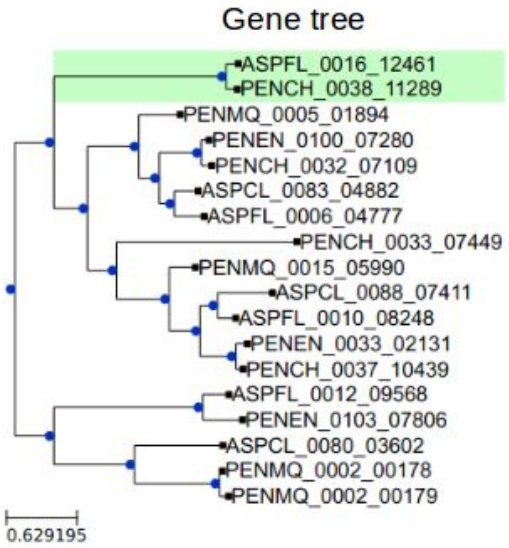

Species overlap
ASPFL_0016_12461
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
ASPFL_0012_09568
PENEN_0103_07806
ASPCL_0080_03602
PENMQ_0002_00178
PENMQ_0002_00179
0.629195

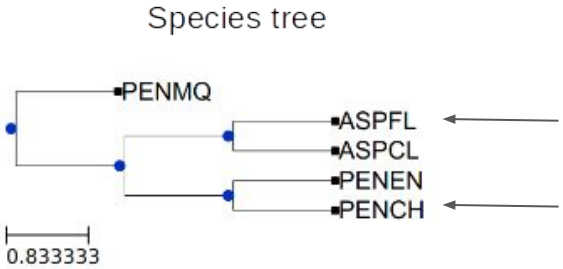

ASPFL_0016_12461
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
ASPFL_0012_09568
PENEN_0103_07806
ASPCL_0080_03602
PENMQ_0002_00178
PENMQ_0002_00179
0.629195

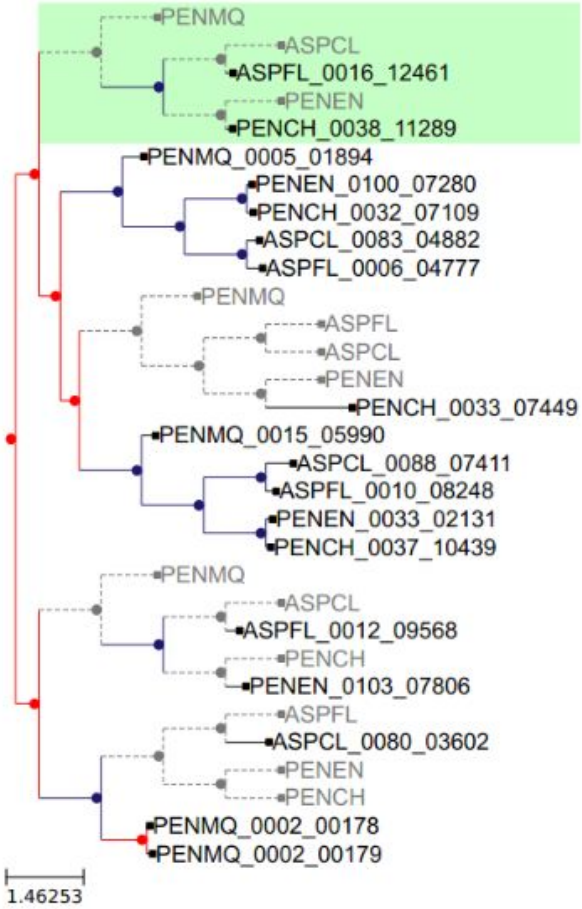

ASPFL_0016_12461
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
ASPFL_0012_09568
PENEN_0103_07806
ASPCL_0080_03602
PENMQ_0002_00178
PENMQ_0002_00179
0.629195

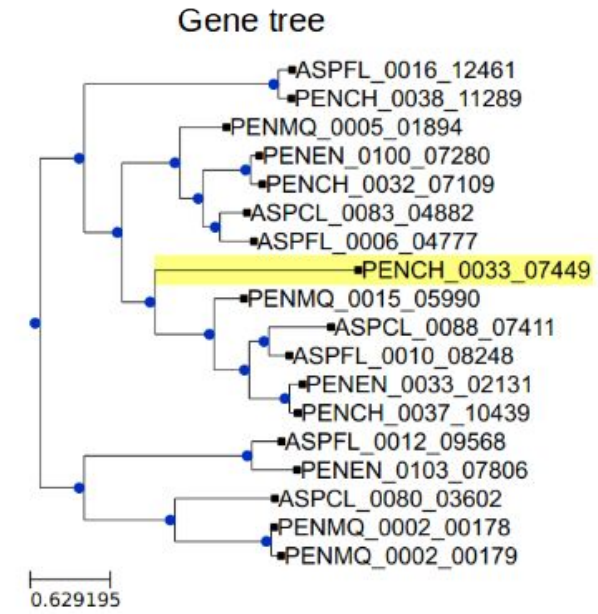

ASPFL_0016_12461
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
ASPFL_0012_09568
PENEN_0103_07806
ASPCL_0080_03602
PENMQ_0002_00178
PENMQ_0002_00179
0.629195

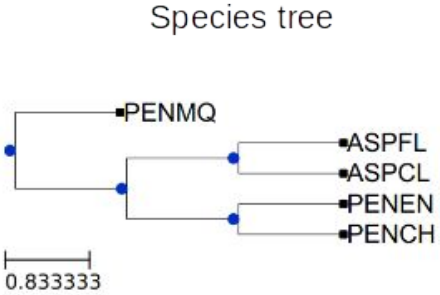

Gene tree
ASPFL_0016_12461
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
ASPFL_0012_09568
PENEN_0103_07806
ASPCL_0080_03602
PENMQ_0002_00178
PENMQ_0002_00179
0.629195
Species tree
PENMQ
ASPFL
ASPCL
PENEN
PENCH
0.833333
L
PENMQ
L
ASPCL
ASPFL_0016_12461
S
L
PENEN
PENCH_0038_11289
D
PENMQ_0005_01894
S
PENEN_0100_07280
S
PENCH_0032_07109
S
S
ASPCL_0083_04882
ASPFL_0006_04777
D
PENMQ
L
ASPFL
ASPCL
PENEN
PENCH_0033_07449
D
D
PENMQ_0015_05990
S
ASPCL_0088_07411
S
ASPFL_0010_08248
S
PENEN_0033_02131
S
PENCH_0037_10439
L
PENMQ
L
ASPCL
ASPFL_0012_09568
S
L
PENCH
PENEN_0103_07806
D
L
ASPFL
ASPCL_0080_03602
L
PENEN
PENCH
S
PENMQ_0002_00178
D
PENMQ_0002_00179
1.46253
Reconciled tree

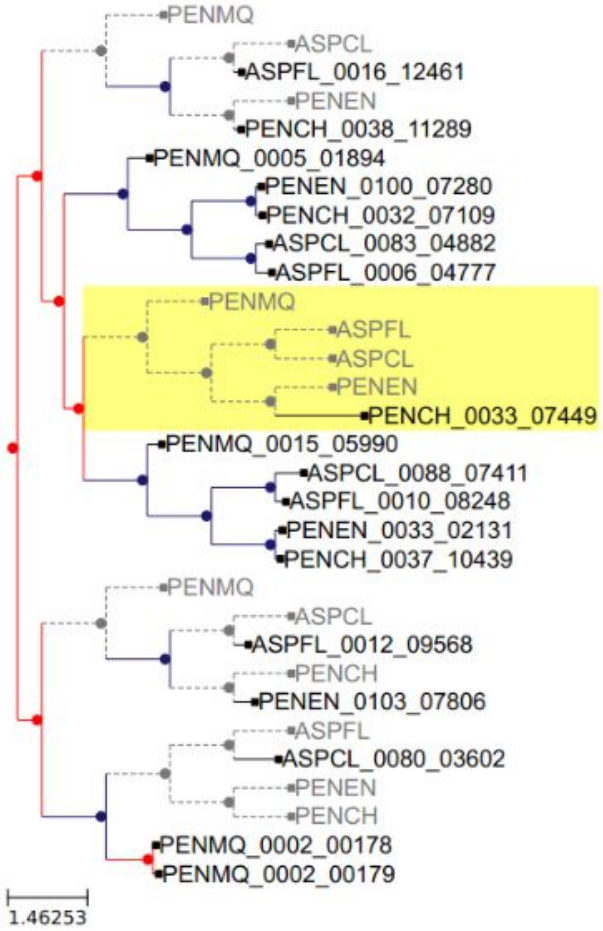

Gene tree
ASPFL_0016_12461
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
ASPFL_0012_09568
PENEN_0103_07806
ASPCL_0080_03602
PENMQ_0002_00178
PENMQ_0002_00179
0.629195

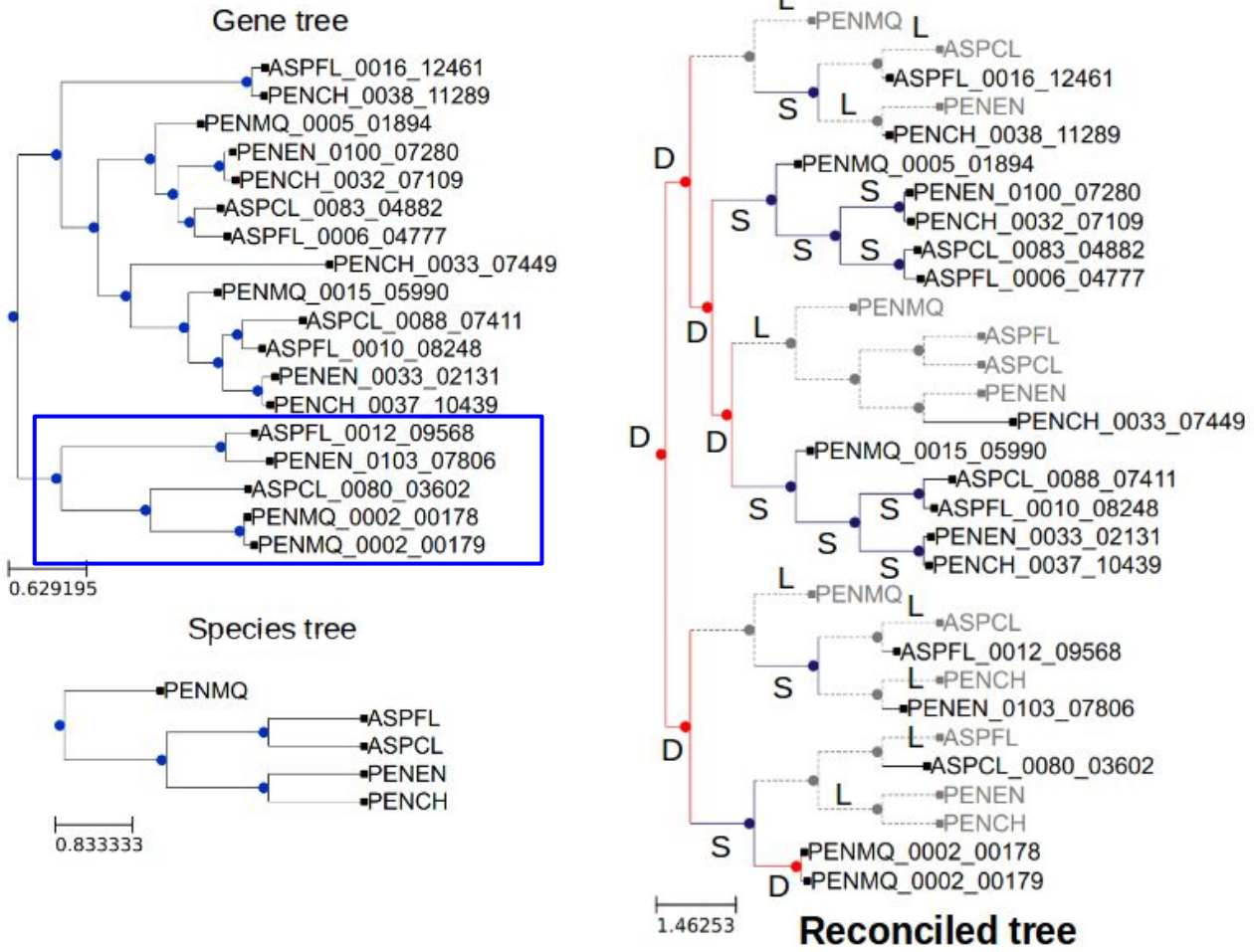

Species tree
PENMQ
ASPFL
ASPCL
PENEN
PENCH
0.833333


PENMQ
ASPCL
ASPFL_0016_12461
PENEN
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENMQ
ASPFL
ASPCL
PENEN
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
PENMQ
ASPCL
ASPFL_0012_09568
PENCH
PENEN_0103_07806
ASPFL
ASPCL_0080_03602
PENEN
PENCH
PENMQ_0002_00178
PENMQ_0002_00179
1.46253

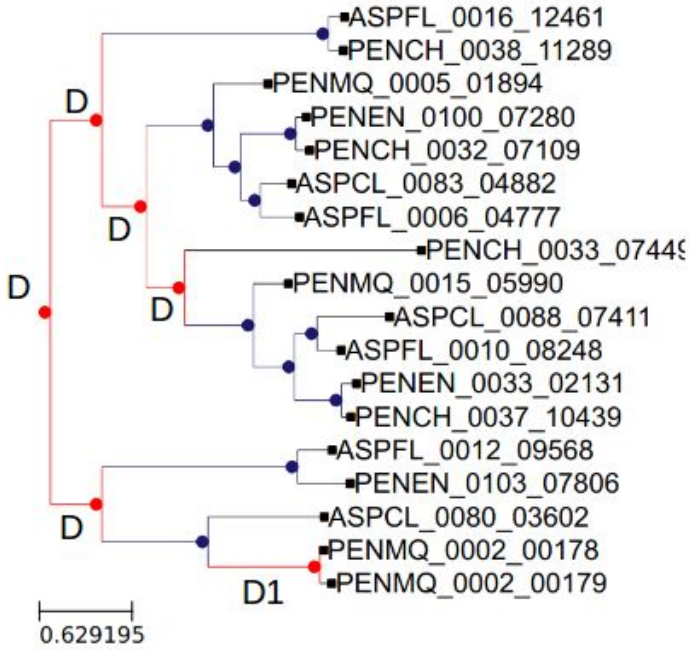

Gene tree
ASPFL_0016_12461
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
ASPFL_0012_09568
PENEN_0103_07806
ASPCL_0080_03602
PENMQ_0002_00178
PENMQ_0002_00179
0.629195


Species tree
PENMQ
ASPFL
ASPCL
PENEN
PENCH
0.833333


PENMQ
ASPCL
ASPFL_0016_12461
PENEN
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENMQ
ASPFL
ASPCL
PENEN
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
PENMQ
ASPCL
ASPFL_0012_09568
PENCH
PENEN_0103_07806
ASPFL
ASPCL_0080_03602
PENEN
PENCH
PENMQ_0002_00178
PENMQ_0002_00179
1.46253

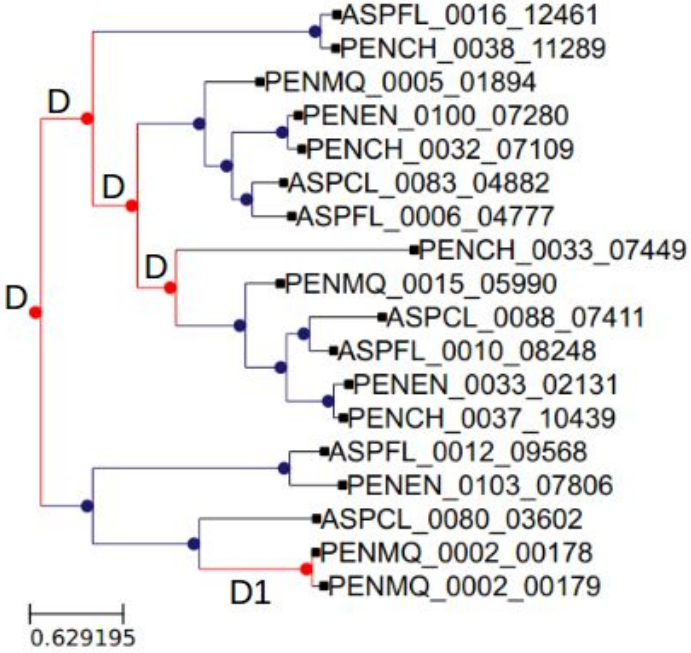

Gene tree
ASPFL_0016_12461
PENCH_0038_11289
PENMQ_0005_01894
PENEN_0100_07280
PENCH_0032_07109
ASPCL_0083_04882
ASPFL_0006_04777
PENCH_0033_07449
PENMQ_0015_05990
ASPCL_0088_07411
ASPFL_0010_08248
PENEN_0033_02131
PENCH_0037_10439
ASPFL_0012_09568
PENEN_0103_07806
ASPCL_0080_03602
PENMQ_0002_00178
PENMQ_0002_00179
0.629195
Species tree
PENMQ
ASPFL
ASPCL
PENEN
PENCH
0.833333
L
PENMQ
L
ASPCL
ASPFL_0016_12461
S
L
PENEN
PENCH_0038_11289
D
PENMQ_0005_01894
S
PENEN_0100_07280
S
PENCH_0032_07109
S
S
ASPCL_0083_04882
ASPFL_0006_04777
PENMQ
D
L
ASPFL
ASPCL
PENEN
PENCH_0033_07449
D
D
PENMQ_0015_05990
S
ASPCL_0088_07411
S
ASPFL_0010_08248
S
PENEN_0033_02131
S
PENCH_0037_10439
L
PENMQ
L
ASPCL
ASPFL_0012_09568
S
L
PENCH
PENEN_0103_07806
D
L
ASPFL
ASPCL_0080_03602
L
PENEN
PENCH
S
PENMQ_0002_00178
D
PENMQ_0002_00179
1.46253
Reconciled tree

Reconciliation



Five duplications at the base of the tree and one species specific duplication.

Species overlap



Four duplications at the base of the tree and one species specific duplication.

How to use ETE3 to predict orthology and paralogy relationships.

```
import ete3

def get_species_tag(node):
      return node.split("_")[0]

tree=ete3.PhyloTree("ASPCL_0083_04882.tree.txt",sp_naming_function=get_species_tag)

t.show()
```

Now we will use the species overlap algorithm:

```
tree.get_descendant_evol_events(
)

tree.show()
```

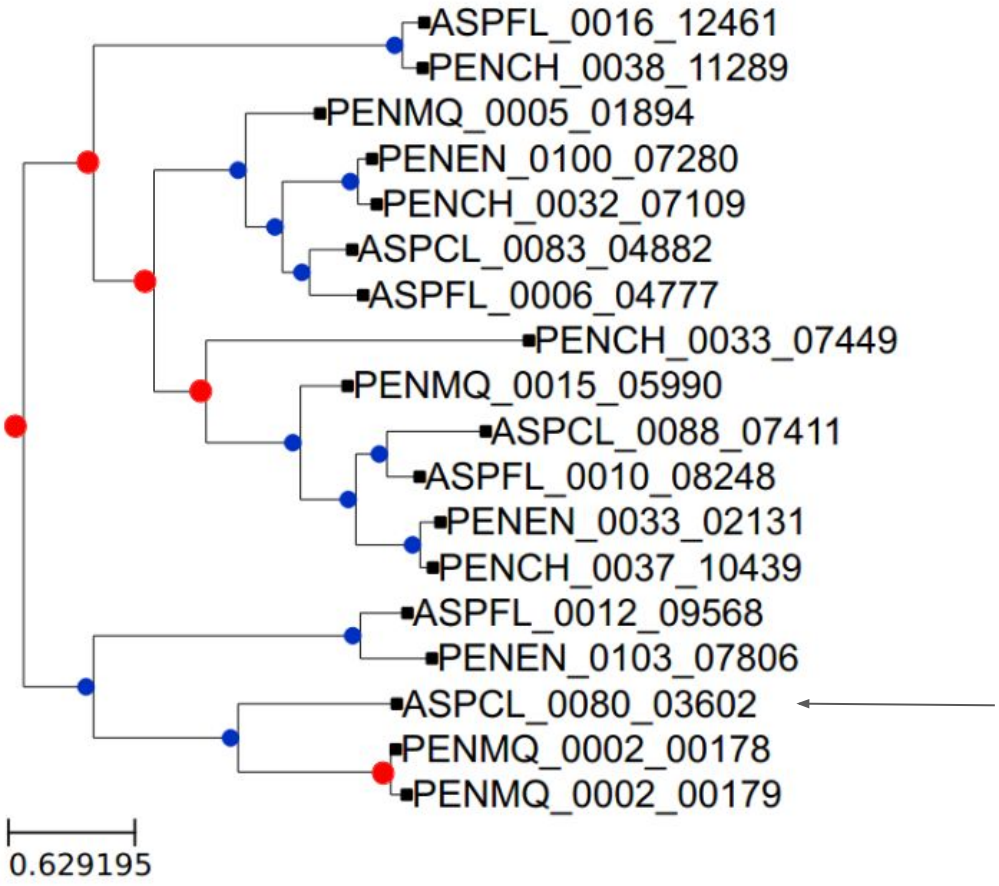
To perform the reconciliation we need to first upload the species tree.

```
def get_whole_name(node):
      return node

spTree=ete3.PhyloTree(“ASPCL_0083_04882.tree.txt”,sp_naming_function=get_whole_name
)

t.reconcile(spTree)

t.show()
```



With the help of the phylogenetic trees, what can you now say about the evolution of protein ASPCL_0080_03602?

How to get pre-calculated orthology predictions

There are websites that will provide pre-calculated orthology predictions. Spend now some time searching for the orthologs of Human TP53 in Mouse, Ciona intestinalis and Drosophila in several on-line databases. You can search TP53 by name or by Blast searchers in the different databases.
Let us try the following databases, explore the different structures and type of information that they provide:

Try to fill the table with the kind of orthology relation with human p53	MOUSE	CIONA	FLY
EggNog (http://eggnogdb.embl.de/#/app/home)			
Ensembl (https://www.ensembl.org/index.html)			
PhylomeDB (http://phylomedb.org/)	One-to-one		
MetaPhOrs (http://betaorthology.phylomedb.org/)			
OMA (https://omabrowser.org/oma/home/)			

Note that some of the databases may not cover all species.
More databases can be found here: https://questfororthologs.org/orthology_databases