CLTree 1.0
User's Manual

Guanghong Zuo
ghzuo@ucas.ac.cn

April 2, 2022

Contents

1 Introduction

CLTree is a tool to annotate and evaluate the phylogenetic tree by lineage (Zuo, 2022). Currently, most studies evaluate a phylogenetic tree by the statistical testing method, e.g. the bootstrap method (Felsenstein, 1985; Sharma and Kumar, 2021; Lemoine et al., 2018; Zuo et al., 2010). As the mean of its name, the bootstrap method evaluates the phylogenetic tree by itself. CLTree provides a way to evaluate the phylogenetic tree by a more independent information, i.e. the the taxonomy system of the biology. The complexity of the algorithm is linear. It is suitable for the studies which content a large number of sequences and taxa. The idea had been used in our previous works (Zuo and Hao, 2015, 2017; Zuo et al., 2013, 2015, 2016, 2018). In the CLTree system, we provide a quantitative method to define the degree of consistency based on the information theory (Shannon, 1948). And the detail differences between phylogenetic tree and lineage is also provided. We believe it will provide a new viewpoint on the study both in phylogeny and taxonomy.

- Please cite CLTree as:

  Guanghong Zuo (2022) CLTree: Annotate Phylogenetic Tree by Lineage and Measure their Consistency based on Shannon Entropy. `in preparation`.

2 Installation and Testing

CLTree is distributed by source code. It can be downloaded from Internet (`https://github.com/ghzuo/Collapse`). There are two ways to compile the source codes of CLTree: compiling the source code by CMake; or using the docker image.

2.1 Normal Unix-like Mode

The program is implemented in the C++ language. The following build tools and libraries are required.

2.1.1 Preparation

- cmake $\geq$ 3.0
- g++ $\geq$ 7.0 or other compiler supporting C++11 standard
- require library: libz, nlohmann-json

2.1.2 Built by CMake

1. unzip the package file and change into it
2. mkdir build and change into it
3. cmake .. and some options you wanted
4. make
5. make install (*option*)

2.1.3 Testing with Example

If this is the first time you use the CLTree package, please go to the "example" folder. Please run the cltree command to get an annotated phylogenetic tree and monophyletic status by:

```
../build/bin/cltree
```

More detail of the command usage can be obtain by '-h' option or read the follow sections.

2.2 Run Collapse in Container

The docker containers make the programs can be performed on both Windows and Linux/MacOS, and transfer the programs easily. To employ the container with Collapse, you should install `docker` at first. You can download docker free and reference from https://docs.docker.com/install/ to how to install it. After installing docker, basic usages for CLTree in the container are shown below:

1. Obtain image: You can build the Collapse docker image based on `Dockerfile` in the source code by command

   ```
   docker build -t="cltree-img" .
   ```

   Here option '-t' set the image name. After build image, you can delete the dangling images for build by `docker image prune`. This will save much hard disk space. You can also download prebuilt Collapse image from internet by command:

   ```
   docker pull ghzuo/cltree .
   ```

   In this step, an image with Collapse programs will obtained.

2. Start container from image: run the follow command in the Collapse directory, i.e. the directory which include the 'example' directory of the Collapse

   ```
   docker run --rm -it -v $PWD/example:/root/data cltree-img
   ```

   In this step, you will enter the Collapse container, and the "example" folder of this project will be find in the "data" folder. Change path to the data folder, and run

   ```
   cltree
   ```

   You will get the result for eight genomes in the `list` file. You can change the path `'$PWD/example'` to your own data directory.

3. Exit and stop container: `exit` in docker terminal.

4. Run Collapse in a temporary container by one command without enter the container:

   ```
   cd <example> or <other data folder>
   docker run --rm -v $PWD:/root/data/ cltree-img cltree
   ```

5. More usage for `docker` can reference https://docs.docker.com/.

3 Workflow and Command-Line Options

3.1 Workflow of CLTree

The workflow of program 'cltree' is shown in Figure 1. Before running the program, users must prepare two objects:

- A phylogenic tree (in newick format, e.g. Tree.nwk).
- Lineage information for the leaves of the phylogenic tree.

The phylogenic tree is obtained in other phylogenic studies. And there are three ways to handle the lineage of the leaves of the phylogenetic tree:

- Use the NCBI taxonomy database dump files directly. It can be downloaded from the NCBI website (https://ftp.ncbi.nlm.nih.gov/pub/taxonomy/taxdump.tar.gz).

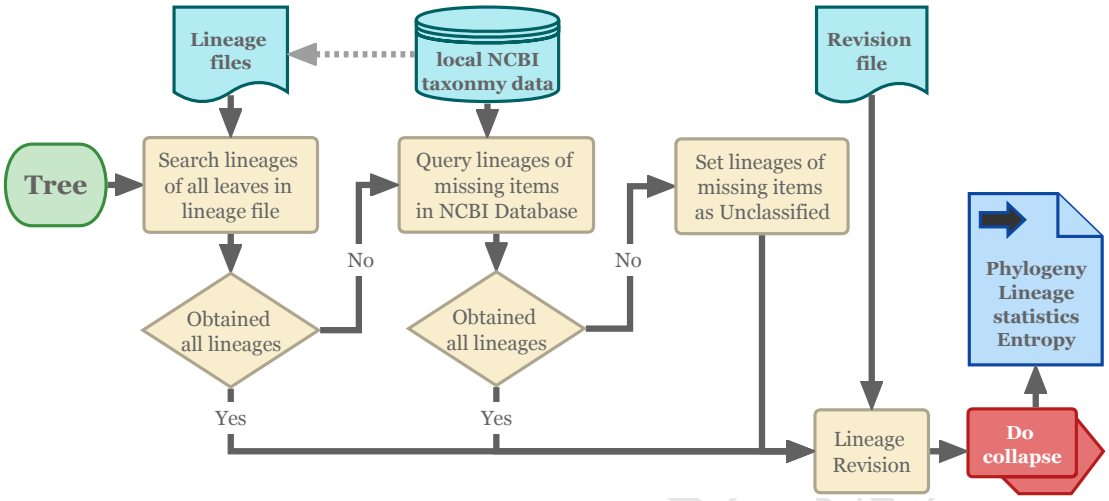

Figure 1: The workflow of CLTree.

- Provide lineage files that include the lineages of phylogenetic tree's leaves. It should be noted that the draft of the lineage files can be generated based on the NCBI taxonomy database dump files, and users only need do revisions on the draft(see below for detail).
- Do lineage revision with revision file, which includes a substitution list, in batch mode. This is an experimental feature and performed based the above two options. It is for the experienced users.

When there is no information for the taxon rank. The taxon name will be set as "Unclassified". In the program, the lineage with "Unclassified" will handled separately.

3.2 Run an Easy Job

The easiest way for preparing lineage information is using the NCBI Taxonomy database dump file package. After downloaded the NCBI taxonomy dump file package, users can obtain the result by the command:

```
cltree -i Tree.nwk -d taxdump.tar.gz
```

In default mode, this command will output four files included annotated tree, lineage statistics, and Shannon entropy between phylogeny and lineage (see Table 1)

To speedup this command, users can make the cache for the NCBI taxonomy database package by using the command:

```
cltree cache -d taxdump.tar.gz
```

Table 1: The default output files of cltree

file name	description
collapsed.entropy	Shannon entropy between lineage and phylogeny for every taxon rank
collapsed.nwk	A newick tree with every node annotated by the common lineage for all sub branches.
collapsed.lineage	Lineages for all genomes in CSV format
collapsed.unit	Statistics for all taxon ranks in the rank order
collapsed.predict (*opt. -P*)	Prediction of unclassified genomes

The command will generate a cache file for the database dump files, named "taxadb.gz". Using the cache file instead of database dump file will speedup the program in future tasks.

3.3 Edit Lineage Information

Except for using the NCBI taxonomy database dump files directly, users can also edit the lineage information manually based on other lineage source. The CLTree system accepted files in CSV format to describe the lineage of species. Users can write the CSV file directly. However, a better solution is generating the CSV format lineage file based on NCBI taxonomy database dump files, and then edit it, i.e.

1. Search the lineage of tree leaves by:

   ```
   cltree search -i Tree.nwk -d taxadb.gz(or taxdump.tar.gz)
   ```

   This command will output a CSV format file named as `Lineage.csv` in default.

2. Edit the `Lineage.csv` file manually.

3. Obtained the result by

   ```
   cltree -i Tree.nwk -l Lineage.csv.
   ```

3.4 Select Output Taxon Ranks

In the textbooks of taxonomy, the standard taxon rank is: domain(superkingdom), kingdom, phylum, class, order, family, genus, species. And this is the default output taxon rank of CLTree system. In some case, user want some other taxon rank, for example: subspecies, subgenus, and etc. When a `lineage.csv` file is used, the output taxon rank is according to the header of `lineage.csv`. In the CLTree system, users can also select the output taxon rank the option `-r`, for example,

```
cltree search -i Tree.nwk -r PCOFGS -d taxadb.gz
```

Here the string `PCOFGS` is the abbreviations of phylum, class, order, family, genus, and species. We noted that the abbreviations of the taxon rank should be sort with the taxon rank. And the abbreviations of the common taxon rank are shown in Table 2:

Table 2: The abbreviation of taxon ranks.

taxon rank	abbreviation	taxon rank	abbreviation
superkingdom	D	superorder	W
kingdom	K	order	O
subkingdom	k	suborder	o
superphylum	Q	family	F
phylum	P	subfamily	f
subphylum	p	genus	G
superclass	L	subgenus	g
class	C	species	S
subclass	c	subspecies	s

In CLTree system, more taxon ranks and abbreviations are also used. Users can obtained them by `cltree rank -a`. Users can also defined their own taxon ranks and abbreviations with a two columns file, and input them with the option `-R`.

3.5 Advance Lineage Tools

There are three formats to describe the lineage information: CSV (Comma Separated Value), JSON (JavaScript Object Notation), and LNS (Lineage Notation String). Here the CSV format is the default format for most case, and the JSON is the interface for Internet options. The LNS format is work format, which is first used in our previous work (Zuo and Hao, 2015). In the lineage notation string, a taxon rank and name pair is `<X>Taxon_name`, here 'X' is the abbreviation of the taxon rank. For example, the full lineage notation string for the species *Escherichia coli* is:

```
<D>Bacteria<K>Bacteria<P>Proteobacteria<C>Gammaproteobacteria
<O>Enterobacterales<F>Enterobacteriaceae<G>Escherichia
<S>Escherichia_coli
```

We noted that the full lineage string is very long string without breaks, we wrapped it into three line here. And the LNS format can be set by option `-F` or output file suffix, i.e.

```
cltree search -o lineage.lns
cltree search -o lineage.txt -F lns
```

A part of the LNS format lineage file is shown below (Some words are omitted and marked with ellipsis). There are two columns in the file. The first column is the name of the leaf, and the second column is it's lineage.

```
Thermofilum_pendens_Hrk_5... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
Vulcanisaeta_distributa_DSM_14429... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
Caldivirga_maquilingensis_IC_167... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
Thermoproteus_tenax_Kra_1... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
Thermoproteus_uzoniensis_768_20... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
Pyrobaculum_islandicum_DSM_4184... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
Thermogladius_calderae_1633... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
Thermosphaera_aggregans_DSM_11486... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
Desulfurococcus_mucosus_DSM_2162... <D>Archaea<P>Crenarchaeota<C>Thermoprotei...
```

The batch revision on lineage is also handle by the lineage notation string. A example of the substitution list is shown below (see also the file `example/Revision.txt`). The hashtag ('#') is the mark for comments, the content after the hashtag will be omitted by program. And the regex can also be used in the first columns. It is implemented by the C++ Template Standard Library (C++ STL). The regex grammar can reference https://en.cppreference.com/w/cpp/regex/ecmascript .

```
# Phylum Ignavibacteriae IJSEM 2014, 64: 8-10
<P>Chlorobi<C>Ignavibacteria <P>Ignavibacteriae<C>Ignavibacteriae

# Phylum B13 Firmicutes
<F>Erysipelotrichaceae<G>Eubacterium <F>Erysipelotrichaceae<G>Erysipelothrix   # CVTree
<P>Firmicutes<C>Erysipelotrichia <P>Tenericutes<C>Erysipelotrichia  # CVTree
<F>Unclassified<G>Exiguobacterium <F>Bacillaceae<G>Exiguobacterium   # CVTree
<F>Planococcaceae<G>Solibacillus <F>Bacillaceae<G>Solibacillus  #   NCBI taxonomy -> LPSN

# Phylum B14 Bacteroidetes
<O>Actinomycetales<F>Nakamurellaceae <O>Nakamurellales<F>Nakamurellaceae  # IJSEM 2014, 64, 3821-3832
<O>Actinomycetales<F>Frankiaceae <O>Frankiales<F>Frankiaceae              # IJSEM 2014, 64, 3821-3832
```

3.6 Handle Tree for iToL

iTOL (https://itol.embl.de/) is an online tool for the display, annotation and management of phylogenetic and other trees (Letunic and Bork, 2021). The output files for handling tree on iTOL can be obtained by option `-I`, e.g.

Table 3: The output files for iTOL with option `-I`

file name	description
collapsed-iToL_tree.nwk	iToL tree file (annotated by id)
collapsed-iToL_nodelist.txt	Lineage and type of all nodes (list by id)
collapsed-iToL_label.txt	iTOL annotate file for node name
collapsed-iToL_popup.txt	iTOL annotate file for popup info
collapsed-iToL_symbol.txt	iTOL annotate file for symbol on clade nodes
collapsed-iToL_strap.txt	iTOL annotate file for strap and color

An example of handing tree on iTOL is shown in figure 2. On the iTOL website, users can handle the nodes of tree to obtain a meaningful tree and output pretty figure for publication. The four annotate files which outputted by `cltree -I` provided basic annotate options on the tree. The `iTOL_tree` is annotated by IDs. And the `iToL_nodelist` file records the type and lineage of every node by ID. Thus users can generate more complex options for iTOL system based on the `iToL_nodelist` file. More info on how to generate annotate files please refer https://itol.embl.de/help.cgi.

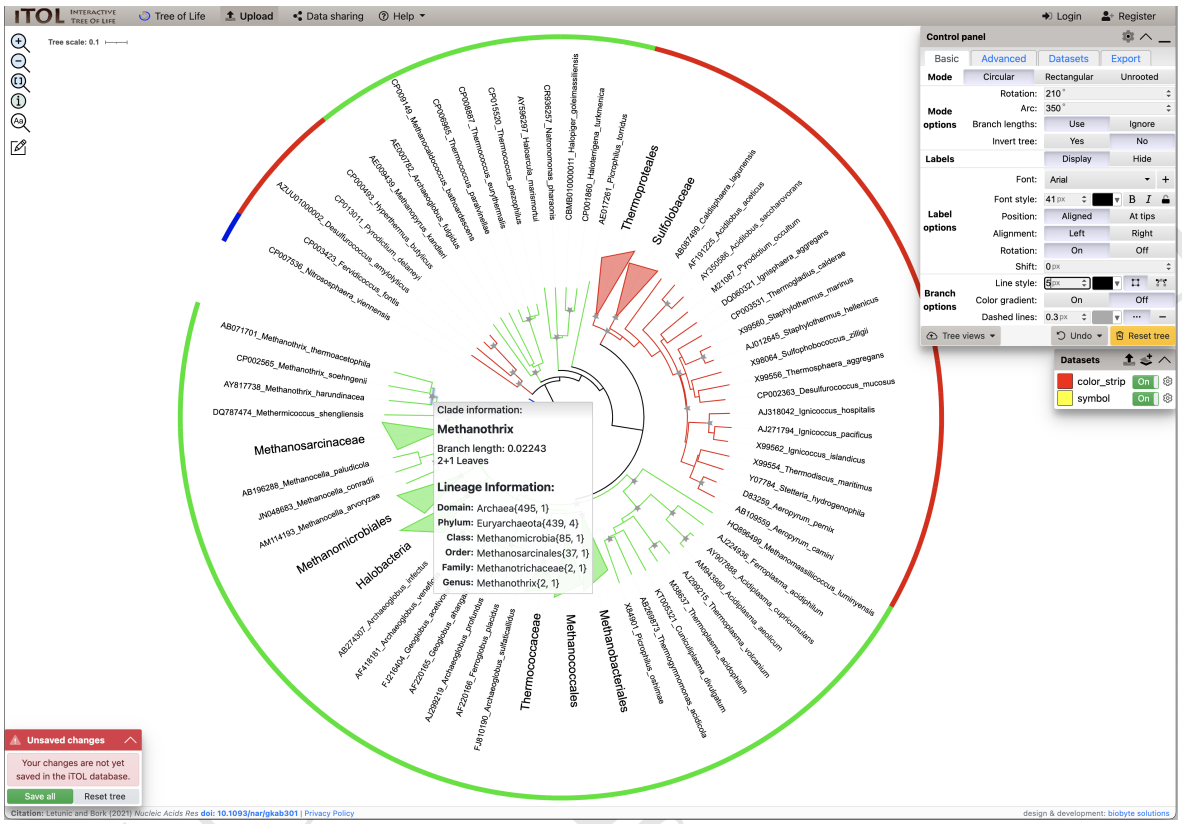

Figure 2: Snapshot of handling tree on iToL website. The stars indicate the clades, and the popup window shows basic info of clade and lineage. On the popup window, the `2 + 1 leaves` means there are 2 leaves with full lineage and 1 leaf with "Unclassified" lineage in this clade. The numbers after every taxon indicate the numbers of genomes and clades of the taxon on the tree.

4 Command-Line Usage

4.1 Tokens and Input/Output Files

The command line of the CLTree is

```
cltree token [options]
```

Here `cltree` is the main command. Users select different tokens for different tasks. Figure 3 shows the scheme of the CLTree, include the tokens (red blocks) and input/output files. The two tokes `search` and `run` perform composite tasks, i.e. it will perform the upstream tasks when the input information is not ready.

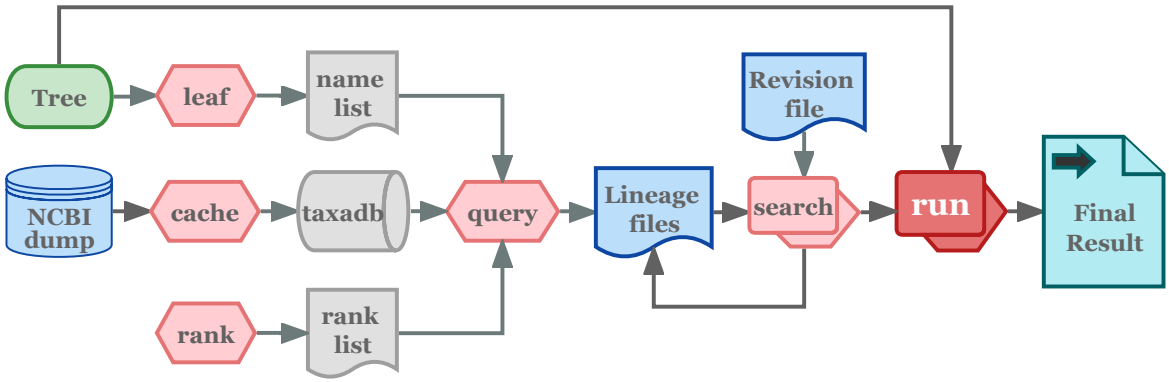

Figure 3: The scheme of CLTree. The red tokens are in red color. The green, blue, cyan and gray blocks indicate phylogeny, lineage, final result and temporary files respectively. And the arrow-lines indicate the I/O of these tokens.

The basic usage of tokens can be obtained by the "help" token, as `cltree help` or `cltree -h`. The output of the help information is shown in below. And users can type `cltree help Task` to get more information about other tasks.

```
cltree Token [options]
 Available Tokens:
   run        All-in-one command: search lineage of leavies,
              annotate phylogenetic tree, and do comparation.
   leaf       Obtain species name list of phylogenetic tree
   cache      Make NCBI database cache from taxdump.tar.gz
   rank       Output taxon rank names and abbreviations
   query      Query lineage from local NCBI taxonomy database
   search     Search lineage from lineage files and NCBI taxonomy
              database, and revised by the revision file
   help       Provide the help information for <Token>
 [ -h ]       Display this information
```

4.2 Usage of Task Tokens

- `run` – The all-in-one command. It search the lineage of all leaves, annotate phylogenetic tree by lineage, and measuring their differences by Shannon entropy. Due to this is the

default task of the command, the token `run` can be omitted except for the "-h" option.

```
cltree run
 [ -D ./ ]              The work directory, default: ./
 [ -i Tree.nwk ]        Input newick tree, default: Tree.nwk
 [ -o collapsed ]       Set prefix name of output files,
                        default: collapsed
 [ -m <Revision.txt> ]  Lineage revision file for batch edit,
                        default: None
 [ -l Lineage.lns ]     Input lineage file for leaves of tree,
                        default: Lineage.lns or Lineage.csv
 [ -d taxadb.gz ]       Taxonomy data file or directory,
                        default: taxadb.gz or taxdump.tar.gz
 [ -R <None> ]          List of rank names and abbreviations,
                        default: set by program
 [ -r DKPCOFGS ]        Abbreviations of output taxon rank,
                        default: set by program
 [ -O <Outgroup> ]      Set the outgroup for the unroot tree.
                        default: None, rearranged by taxonomy
 [ -I ]                 Output newick and annotate files for itol
                        default: No
 [ -P ]                 Output prediction for undefined leafs
 [ -q ]                 Run command in quiet mode
 [ -h ]                 Display this information
```

- `leaf` – Obtain the name list of all leaves of phylogenetic tree

```
cltree leaf
 [ -i Tree.nwk ]      Input tree file,
                      default: Tree.nwk
 [ -o namelist.txt ]  Output name list file,
                      default: namelist.txt
 [ -q ]               Run command in quiet mode
 [ -h ]               Display this information
```

- `search` – Search the lineage of the genome in the name list file from the lineage files and NCBI taxonomy database dump file, and do the revision by the revision file.

```
cltree search
```

```
  [ -i namelist.txt ]  Input name list, ':N' after the file name
                       select the N column of the file
                       default: first column of namelist.txt
  [ -o Lineage.csv ]   Output lineage file, default: lineage.csv
  [ -F CSV ]           Set the output lineage file format
                       default: CSV
  [ -m Revision.txt ]  Lineage revise file for batch edit,
                       default: None
  [ -l Lineage.lns ]   Lineage file for leafs of tree,
                       default: Lineage.lns or Lineage.csv
  [ -d taxadb.gz ]     Taxa database file or directory,
                       default: taxadb.gz or taxdump.tar.gz
  [ -R <None> ]        List file for rank names
                       and abbreviations,
                       default: set by program
  [ -r DKPCOFGS ]      Set output taxon rank by abbreviations,
                       default: set by program
  [ -q ]               Run command in quiet mode
  [ -h ]               Display this information
```

- `cache` – Package the dump files of NCBI taxonomy database as cache to speedup the process of querying lineage from database.

```
cltree cache
  [ -d taxdump.tar.gz ]  NCBI taxon dump,
                         default: taxdump.tar.gz
  [ -o taxadb.gz ]       Packaged taxon database,
                         default: taxadb.gz
  [ -q ]                 Run command in quiet mode
  [ -h ]                 Display this information
```

- `rank` – Output an example for the ordered list of the taxon rank names and abbreviations.

```
cltree rank
  [ -o ranklist.txt ]  Output name list file,
                       default: ranklist.txt
  [ -a ]               Output all inbuilt ranks
  [ -q ]               Run command in quiet mode
  [ -h ]               Display this information
```

- `query` – Query the lineage of the genome ONLY from NCBI Taxonomy database dump files or the database cache, and output the lineage in the lineage string format.

```
cltree query
 [ -I <Taxon ID> ]      Query a taxon id
 [ -N <Taxon Name> ]    Query a taxon name
 [ -i namelist.txt ]    The query list file,
                        default: namelist.txt
 [ -d taxadb.gz ]       The NCBI taxonomy data file
                        default: taxadb.gz or taxdump.tar.gz
 [ -o Lineage.txt ]     Output lineage file,
                        default: Lineage.txt
 [ -R <None> ]          List file for rank names and
                        abbreviations,
                        default: set by program
 [ -r DKPCOFGS ]        Set output taxon rank by
                        abbreviations,
                        default: set by program
 [ -H ]                 Don't output missing items
 [ -q ]                 Run command in quiet mode
 [ -h ]                 Display this information
```

5 Algorithm

5.1 A Recursive Algorithm To Annotate the Phylogeny Tree

5.1.1 For Rooted Phylogenetic Tree

After obtain the lineage of all leaves of the phylogenetic tree, a rooted tree can be annotated by a recursive algorithm, which named as "Collapse" algorithm. The pseudo-code was show below:

5.1.2 For Unrooted Phylogenetic Tree

The collapse algorithm can only annotate the rooted tree, due to the taxonomy of biology is a hierarchic classified system. In the CLTree system, there are two methods to annotate a unrooted phylogenetic tree:

- Set a leaf as the out-group for the phylogenetic tree manually.

Algorithm 1 Collapse Algorithm for Rooted Tree

Input: Root Node of Unannotated Tree
Output: Root Node of Annotated Tree

```
Initialization: get lineages of leaves
for node ∈ root.children do
    ANNOTATE(node)
end for

function ANNOTATE(node)
    if nd.children.size ≠ 0 then
        for nd ∈ node.children do
            ANNOTATE(nd)
        end for
        node.lineage ← common lineage of note.children
    end if
end function
```

- Select a branch as the root of the phylogenetic tree to obtain the best consistency between the phylogenetic tree and taxonomy (default).

For the second method, the degree of consistency is measured by the entropy reduction ratio in the CLTree system (see the next section). And for an input unrooted phylogenetic tree, the process to handle the annotation is that:

1. Random select a leaf as the out-group, and annotate the tree.
2. Obtain the root candidates by Algorithm 2.
3. Select a candidates as the root, rearrange the tree, and annotate the new subtree.
4. Compare the lineage of sub-branches of the root, find the optimal out-group, and annotate the new nodes.

It should be noted that the algorithm 2 may find more than one candidate. All of them process the same entropy reduction ratio. And after the annotation, the unrooted tree will become a rooted tree.

5.2 Entropy Reduction Ratio

It is obvious that all leaves of the phylogenetic tree was be divided into many groups at every taxon rank, and every leaf belong to one and only one group. As the information theory, the

Algorithm 2 A Recursive Algorithm To Find Root Candidates for Unrooted Tree

Input: Pre-annotated Tree
Output: Root Candidates

```
Initialization: candidates
topRank ← the rank of root
EXAMINE(root, topRank, candidates)

function EXAMINE(node, topRank, candidates)
    if rank(node) = topRank then
        if ∩rank(node.children) < topRank then
            add node to Candidates
        else
            for nd ∈ node.children do
                EXAMINE(nd, topRank, candidates)
            end for
        end if
    end if
end function
```

Shannon entropy of a taxon rank is defined as:

$$H_{taxa} = -\sum_i \frac{N_i}{N_{leaf}} \log_2 \frac{N_i}{N_{leaf}} \tag{1}$$

Here N_i is the number of leaves of a group (taxonomic unit), and the N_{leaf} is the total number of leaves in the phylogenetic tree.

To obtain the Shannon entropy of a taxon rank for a rooted annotated tree, we defined the taxon branches on the phylogenetic tree. On the annotated tree, when the taxon rank of a branches is lower than its parent branches, the node is defined as a taxon branches of the taxon rank. These taxon branches of a taxon rank also divide all leaves of the phylogenetic tree into many groups, and every leaf belong to one and only one group. Thus the Shannon entropy of a taxon rank for the phylogenetic tree is defined as:

$$H_{tree} = -\sum_i \frac{M_i}{N_{leaf}} \log_2 \frac{M_i}{N_{leaf}} \tag{2}$$

Here M_i is the number of the leaf on a branch.

It is obvious that $H_{taxa} \leq H_{tree} \leq H_{max}$, and $H_{max} = -\sum_i \frac{1}{N_{leaf}} \log_2 \frac{1}{N_{leaf}}$ for the case that every leaf belong to different taxonomic unit with their sibling. Thus in the CLTree system, the degree of consistency between the phylogenetic tree and taxonomy at a taxon rank is measured by the entropy reduction ratio

$$\widetilde{H} = \frac{H_{max} - H_{tree}}{H_{max} - H_{taxa}} \tag{3}$$

Here $0 \leq \widetilde{H} \leq 1$, 0 for the worst case, in which every leaf belong to different taxonomic unit with their sibling; and 1 for all taxonomic units at the the taxon rank are monophyletic on the phylogenetic tree (Zuo, 2021).

References

G. H. Zuo. CLTree: Annotate phylogenetic tree by lineage and measure their differences by shannon entropy. *In Preparation*, 2022.

J. Felsenstein. Confidence limits on phylogenies: An approach using the bootstrap. *Evolution*, 39(4):783–791, 1985. doi: 10.1111/j.1558-5646.1985.tb00420.x. URL https://www.ncbi.nlm.nih.gov/pubmed/28561359.

S. Sharma and S. Kumar. Fast and accurate bootstrap confidence limits on genome-scale phylogenies using little bootstraps. *Nat Comput Sci*, 1(9):573–577, 2021. doi: 10.1038/s43588-021-00129-5. URL https://www.ncbi.nlm.nih.gov/pubmed/34734192.

F. Lemoine, J. B. Domelevo Entfellner, E. Wilkinson, D. Correia, M. Davila Felipe, T. De Oliveira, and O. Gascuel. Renewing felsenstein's phylogenetic bootstrap in the era of big data. *Nature*, 556(7702):452–456, 2018. doi: 10.1038/s41586-018-0043-0. URL https://www.ncbi.nlm.nih.gov/pubmed/29670290.

G. H. Zuo, Z. Xu, H. J. Yu, and B. L. Hao. Jackknife and bootstrap tests of the composition vector trees. *Genomics Proteomics Bioinformatics*, 8(4):262–7, 2010. doi: 10.1016/S1672-0229(10)60028-9. URL https://www.ncbi.nlm.nih.gov/pubmed/21382595.

G. H. Zuo and B. H. Hao. CVTree3 web server for whole-genome-based and alignment-free prokaryotic phylogeny and taxonomy. *Genomics Proteomics and Bioinformatics*, 13 (5):321–31, 2015. doi: 10.1016/j.gpb.2015.08.004. URL https://www.ncbi.nlm.nih.gov/pubmed/26563468.

G. H. Zuo and B. L. Hao. *Whole-Genome-Based Phylogeny and Taxonomy for Prokaryotes*, book section Chapter 6. 2017. ISBN 978-953-51-3499-2 978-953-51-3502-9. doi: 10.5772/intechopen.68563.

G. H. Zuo, Z. Xu, and B. L. Hao. Shigella strains are not clones of Escherichia coli but sister species in the genus Escherichia. *Genomics Proteomics Bioinformatics*, 11:61–65, 2013.

G. H. Zuo, Z. Xu, and B. L. Hao. Phylogeny and taxonomy of archaea: A comparison of the whole-genome-based cvtree approach with 16s rrna sequence analysis. *Life (Basel)*, 5 (1):949–68, 2015. doi: 10.3390/life5010949. URL https://www.ncbi.nlm.nih.gov/pubmed/25789552.

G. H. Zuo, X. Y. Zhi, Z. Xu, and B. L. Hao. Lvtree viewer: An interactive display for the all-species living tree incorporating automatic comparison with prokaryotic systematics. *Genomics Proteomics Bioinformatics*, 14(2):94–102, 2016. doi: 10.1016/j.gpb.2015.12.002. URL https://www.ncbi.nlm.nih.gov/pubmed/27018315.

G. H. Zuo, J. Qi, and B. L. Hao. Polyphyly in 16s rrna-based lvtree versus monophyly in whole-genome-based cvtree. *Genomics Proteomics Bioinformatics*, 16(5):310–319, 2018. doi: 10.1016/j.gpb.2018.06.005. URL https://www.ncbi.nlm.nih.gov/pubmed/30550857.

Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948. doi: 10.1145/584091.584093. URL https://dl.acm.org/doi/pdf/10.1145/584091.584093.

I. Letunic and P. Bork. Interactive tree of life (itol) v5: an online tool for phylogenetic tree display and annotation. *Nucleic Acids Res*, 49(W1):W293–W296, 2021. doi: 10.1093/nar/gkab301. URL https://www.ncbi.nlm.nih.gov/pubmed/33885785.

G. H. Zuo. Cvtree: A parallel alignment-free phylogeny and taxonomy tool based on composition vectors of genomes. *Genomics Proteomics Bioinformatics*, 19:1–6, 2021. doi: 10.1016/j.gpb.2021.03.006. URL https://www.ncbi.nlm.nih.gov/pubmed/34119695.