An encapsulated Python pipeline application for generating maximum likelihood phylogenetic trees from NCBI accession numbers

Mark Nygard
UCSD Jacobs School of Engineering Department of Computer Science and Engineering
5 June 2020

Introduction

The process of determining the evolutionary timeline of differentiation between related organisms is an important factor in understanding the core similarities and differences between species. At a genetic level, these kinds of analyses allow us to infer the mutations that make a species unique and give us an idea of when those key modifications might have taken place. The most common way to visualize the relationships that are constructed in this process is in the form of phylogenetic trees. Often when a novel species or strain is isolated, the first course of action is to sequence its genome and place it in its respective phylogenic clade in order to confirm and assess its evolutionary relationship to similar organisms. In 2015, Sivian Laviad, et. al published the genome of the novel Leucobacter strain *Leucobacter chironomi*, along with the phylogenetic tree they produced showing its evolutionary relationship to the 15 other strains in the genus and 10 additional closely related strains. In the current study, we aimed to firstly reproduce Figure 1: The Phylogenetic Tree ("Fig. 1") from Laviad's paper entitled "High quality draft genome sequence of *Leucobacter chironomi* … isolated from a *Chironomus* sp. egg mass." Secondly, we attempted to analyze the raw data using alternative available software tools to elucidate what effects, if any, they would have on the resulting tree. We will present two trees resulting from the same dataset with significant differences in their structure based on the tools used for analysis. Ultimately, we sought to build a unary encapsulation of the modular tools that are required to generate and visualize a phylogenetic tree from the most basic raw data of NCBI accession numbers. We deliver an initial working model for a python-contained solution that simplifies the overall workflow into a single interface and allows for combinations of intermediate tools to be utilized in each step of analysis. Applying our implementation to the original study's dataset of accession numbers produced a tree that was >93% similar to the figure presented by Laviad.

Methods

Acquisition of Nucleotide Sequences

Accession numbers for each of the strains included in the phylogenetic tree were available from Fig. 1. Fifteen strains of genus *Leucobacter* were compared with the novel genome of *Leucobacter chironomi*, along with 10 additionally related strains. Sequences for each strain were obtained from NCBI's nucleotide database in fasta format using python's built-in requests library. Some of these sequences had been updated since the publication of Laviad's paper. The contents of each fasta sequence were combined into a master fasta file containing all 26 sequences. Each species' sequence was ~1.5 kilobases (kb) and the combined fasta file was ~39 kb.

Multiple Sequence Alignment

Laviad used the program Clustal W for multiple sequence alignment (MSA) of the 26 sequences to generate a maximum scoring alignment. In order to fulfill our second objective of testing multiple tools throughout each step of the pipeline, we created two separate workflows. We will refer to the first workflow as profile 1, in which we used Clustal (version 2.1; -output clustal) for MSA, and our alignment was formatted as a clustal file. The alignment produced by Clustal was 1,556 bp.

In the second workflow, which we will refer to as profile 2, we implemented mafft (v7.453; --retree 2 –reorder) to perform MSA. This generated a fasta formatted alignment that was 1,560 bp.

Maximum Likelihood Tree Generation

Laviad simply stated that the tree from Fig. 1 was generated using the maximum likelihood (ML) method in MEGA 5 software and bootstrap values were calculated from 1,000 replicates. Currently, we chose not to implement MEGA 5, partly because the version used in the original paper has since been deprecated, but largely because no python wrapper for the command line instance of the software is currently available in BioPython. We will expand upon this choice in the discussion. Instead, for profile 1, we chose to use Clustal's ML tree building capability (--tree -outputtree phylip) for ML tree creation and bootstrapping (-bootstrap 100 -seed 1000 -outputtree phylip). This method generated a guide tree (.dnd) in addition to our ML tree (.ph) and a bootstrapped bipartitioned tree (phb). For profile 2, RAxML was used for ML tree generation and bootstrapping. Implementation of the RAxML wrapper class within BioPython has yet to be completed, so this portion of profile 2 was performed outside of the python pipeline using the command-line interface. Non-standard

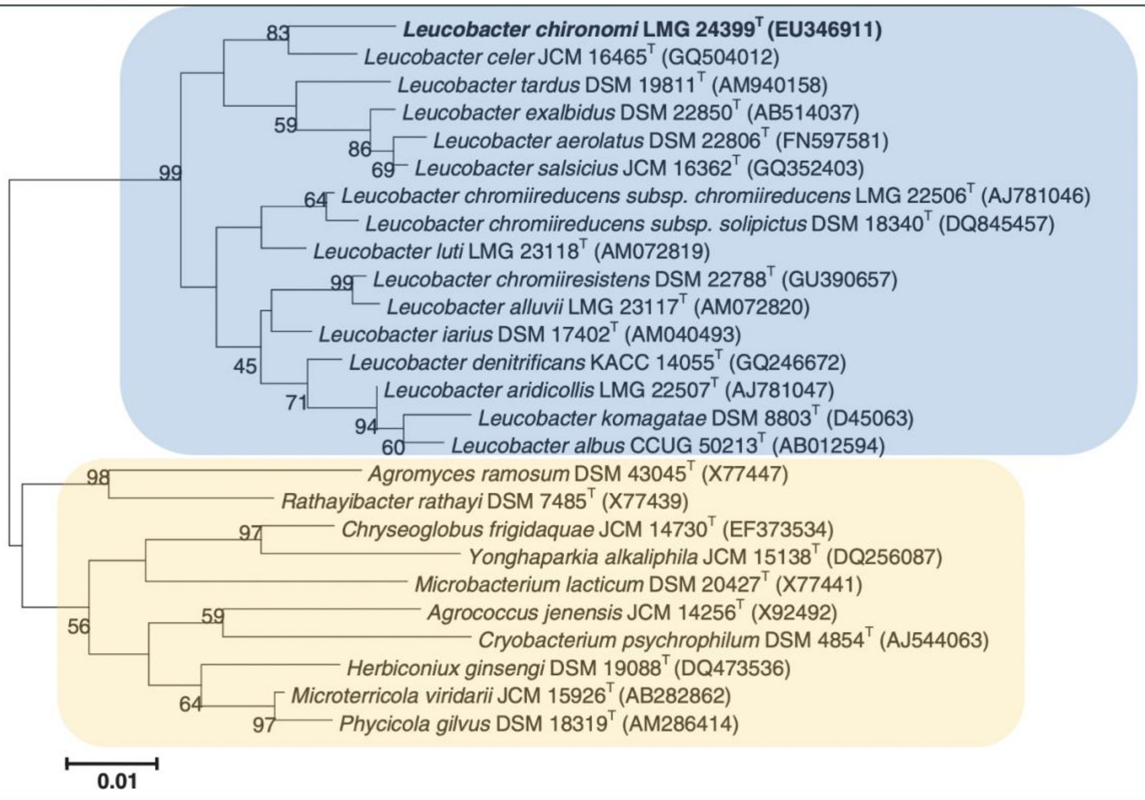
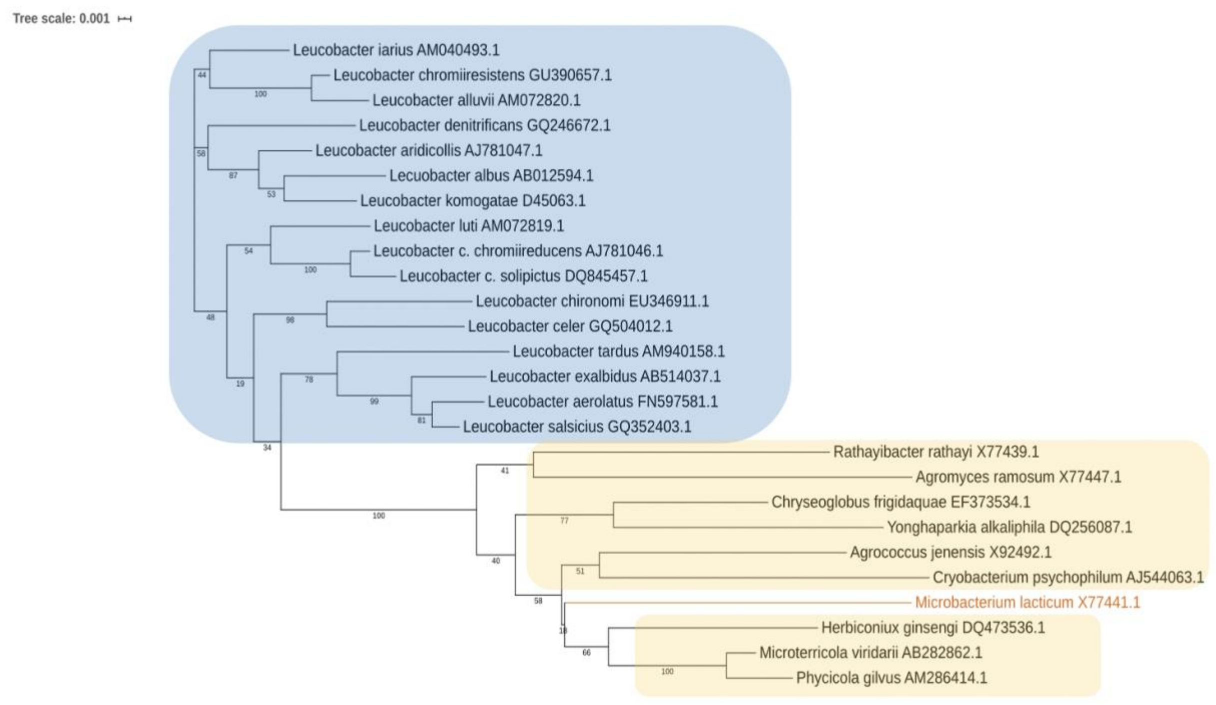

Leucobacter chironomi LMG 24399T (EU346911)
Leucobacter celer JCM 16465T (GQ504012)
Leucobacter tardus DSM 19811T (AM940158)
Leucobacter exalbidus DSM 22850T (AB514037)
Leucobacter aerolatus DSM 22806T (FN597581)
Leucobacter salsicius JCM 16362T (GQ352403)
Leucobacter chromiireducens subsp. chromiireducens LMG 22506T (AJ781046)
Leucobacter chromiireducens subsp. solipictus DSM 18340T (DQ845457)
Leucobacter luti LMG 23118T (AM072819)
Leucobacter chromiiresistens DSM 22788T (GU390657)
Leucobacter alluvii LMG 23117T (AM072820)
Leucobacter iarius DSM 17402T (AM040493)
Leucobacter denitrificans KACC 14055T (GQ246672)
Leucobacter aridicollis LMG 22507T (AJ781047)
Leucobacter komagatae DSM 8803T (D45063)
Leucobacter albus CCUG 50213T (AB012594)
Agromyces ramosum DSM 43045T (X77447)
Rathayibacter rathayi DSM 7485T (X77439)
Chryseoglobus frigidaquae JCM 14730T (EF373534)
Yonghaparkia alkaliphila JCM 15138T (DQ256087)
Microbacterium lacticum DSM 20427T (X77441)
Agrococcus jenensis JCM 14256T (X92492)
Cryobacterium psychrophilum DSM 4854T (AJ544063)
Herbiconiux ginsengi DSM 19088T (DQ473536)
Microterricola viridarii JCM 15926T (AB282862)
Phycicola gilvus DSM 18319T (AM286414)
83
59
86
69
99
64
99
45
71
94
60
98
97
56
59
64
97
0.01

Figure 1a. *Figure 1 from High quality draft genome sequence of* Leucobacter chironomi *… isolated from a* Chironomus *sp. egg mass," Laviad, et. al 2015. Genus* Leucobacter *is highlighted in blue, others are in yellow.*

RAxML (v8.2.12) options for ML tree generation were -PTHREADS-SSE3 -m GTRGAMMA -p 12345 -# 100 -T 6. GTRGAMMA method was used as the documentation specifies that GTRCAT is only to be used on datasets with >50 taxa. RAxML was then used to bootstrap with 100 replicates (-m GTRGAMMA -p 12345 -b 12345 -# 100) and bipartition the tree (-m GTRCAT -p 12345 -f b).

Tree Visualization

Laviad did not provide documentation regarding the method of tree visualization, although it can be accomplished with MEGA 5. For the purpose of this study and to provide a figure that could be easily compared to Fig. 1, we have used iTol (Interactive Tree of Life; https://itol.embl.de/) to visualize the trees from both profiles 1 and 2. This method currently requires that the text from the bootstrapped tree be copied to the iTol site independently of the python pipeline; however, the functionality to load the contents of the tree file directly to iTol within the pipeline can be trivially added using HTTP requests. Importantly, once the tree files have been created, our application uses the fasta files that were downloaded



Tree scale: 0.001
Leucobacter iarius AM040493.1
Leucobacter chromiiresistens GU390657.1
Leucobacter alluvii AM072820.1
Leucobacter denitrificans GQ246672.1
Leucobacter aridicollis AJ781047.1
Lecuobacter albus AB012594.1
Leucobacter komogatae D45063.1
Leucobacter luti AM072819.1
Leucobacter c. chromiireducens AJ781046.1
Leucobacter c. solipictus DQ845457.1
Leucobacter chironomi EU346911.1
Leucobacter celer GQ504012.1
Leucobacter tardus AM940158.1
Leucobacter exalbidus AB514037.1
Leucobacter aerolatus FN597581.1
Leucobacter salsicius GQ352403.1
Rathayibacter rathayi X77439.1
Agromyces ramosum X77447.1
Chryseoglobus frigidaquae EF373534.1
Yonghaparkia alkaliphila DQ256087.1
Agrococcus jenensis X92492.1
Cryobacterium psychophilum AJ544063.1
Microbacterium lacticum X77441.1
Herbiconiux ginsengi DQ473536.1
Microterricola viridarii AB282862.1
Phycicola gilvus AM286414.1

Figure 1b. *Profile 1 phylogenetic tree reproduction of Fig. 1a. Clustal W2 was used for MSA as well as ML tree generation and bootstrapping. Genus* Leucobacter *is highlighted in blue, others are in yellow.*

from the accession numbers to re-associate the species name with the accession number in the tree. This step uses a dictionary of accession numbers and parses the combined fasta file from the acquisition of nucleotide sequences to assign species names to each number, allowing the tree to be readable without manually entering species information.

Results

Tree Comparisons

The paper by Laviad used *de novo* sequencing by Illumina NGS to determine the sequence of *Leucobacter chironomi* and compared the results to known sequences from related strains from GenBank. Figure 1a shows the resulting tree. Our analysis of the sequences in profile 1, using Clustal for both MSA and tree generation can be seen in Figure 1b. With the species of genus *Leucobacter* highlighted in blue, we can see that the phylogeny within the genus was very similarly identified. We used a general method for determining percentage similarity by assigning points for each

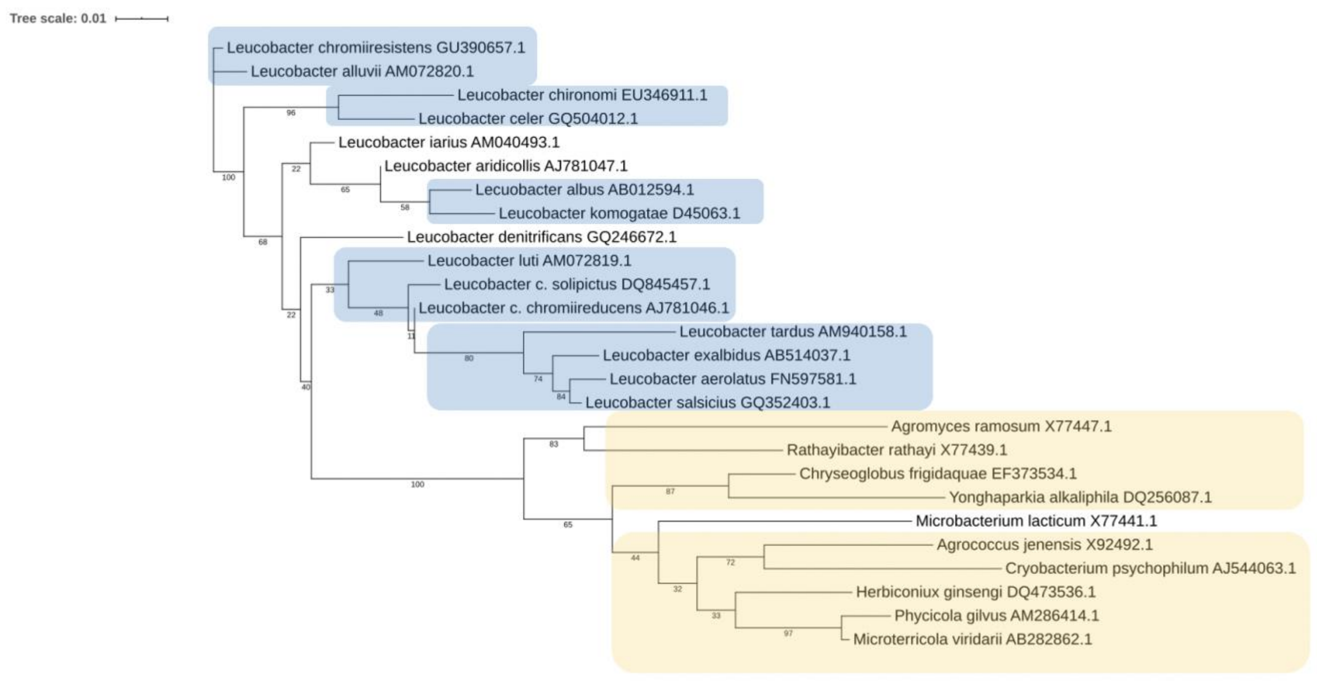

Tree scale: 0.01
Leucobacter chromiiresistens GU390657.1
Leucobacter alluvii AM072820.1
Leucobacter chironomi EU346911.1
Leucobacter celer GQ504012.1
Leucobacter iarius AM040493.1
Leucobacter aridicollis AJ781047.1
Lecuobacter albus AB012594.1
Leucobacter komogatae D45063.1
Leucobacter denitrificans GQ246672.1
Leucobacter luti AM072819.1
Leucobacter c. solipictus DQ845457.1
Leucobacter c. chromiireducens AJ781046.1
Leucobacter tardus AM940158.1
Leucobacter exalbidus AB514037.1
Leucobacter aerolatus FN597581.1
Leucobacter salsicius GQ352403.1
Agromyces ramosum X77447.1
Rathayibacter rathayi X77439.1
Chryseoglobus frigidaquae EF373534.1
Yonghaparkia alkaliphila DQ256087.1
Microbacterium lacticum X77441.1
Agrococcus jenensis X92492.1
Cryobacterium psychophilum AJ544063.1
Herbiconiux ginsengi DQ473536.1
Phycicola gilvus AM286414.1
Microterricola viridarii AB282862.1
96
100
22
65
58
68
33
48
11
22
80
74
84
40
83
100
87
65
44
72
32
33
97

Figure 1c. *Profile 1 phylogenetic tree reproduction of Fig. 1a. Clustal W2 was used for MSA as well as ML tree generation and bootstrapping. Genus* Leucobacter *is highlighted in blue, others are in yellow.*

similarly placed node, assigning 1 point when a node's nearest relation was accurately matched based on the original figure, and a second point if the next nearest neighbor was accurately assigned, for a maximum of 52 points in the reproduced figure. We found that profile 1 scored a similarity of >94% (49 points of 52 possible). Profile 2 is highlighted in corresponding colors in Figure 1c. The combination of using maffta for MSA and RAxML for tree generation introduced more variation to the relative phylogeny within genus *Leucobacter* compared to the original figure. However, the genus is still clearly defined in this profile. We found that the similarity score for profile 2 was >88% (46 points of 52 possible). †

Discussion

The process of creating a phylogenetic tree from a given set of nucleotide sequences can provide a powerful tool for visualizing and analyzing the evolutionary relationship between organisms. The pipeline for doing so is well established and many tools for MSA and ML tree generation exist. While these tools often produce very similar results, we have shown how different approaches can yield varying trees.

† As the format of these figures does not allow them to be compared side by side in the text, a side by side comparison can be found in the appendix.

A primary factor we identify as a contributor to the differences in our reproduction compared to the original figure is the range of possible combinations of algorithmic options available within each different software, as well as “frequent code and data structure changes” in the underlying computational models of each tool (Stamatakis 6). These changes can drastically affect how the data is analyzed. Variations can also be attributed to updates to the sequences being analyzed. Many of the accession numbers used in this study have been updated between the publish data of the original in 2015 and present. Thus, while it can be useful to attempt to reproduce previously generated trees as we did to begin this study, here we posit that differential analysis of multiple possible trees from a given set of sequences can provide an additional level of insight. However, the modular nature of using different software tools for analysis at each step in the workflow can make performing multiple runs time consuming and prone to error.

To increase the efficiency of the workflow from sequence acquisition to tree visualization, we sought to create a single interface pipeline that is entirely contained within Python. We present an initial working model for achieving such an application. The current application uses native Python modules as well as the BioPython library to incorporate multiple tools for MSA and tree generation into a streamlined workflow. Given a list of NCBI accession numbers, we can generate a single combined fasta file containing each sequence for analysis. Using BioPython’s Applications class, which provides wrappers for command line tools such as Clustalw and mafft, the combined fastas can be subjected to multiple sequence alignment either by mafft or Clustalw. Currently, only the specific non-standard options that we used for our study have been implemented for these tools, but the framework we have designed is easily scalable to include all possible options. For tree generation and bootstrapping, our design currently only allows for Clustalw to be used. However, BioPython does provide a wrapper class for the RAxML command line utility, and future builds of the application will aim to include the option for RAxML tree generation and bootstrapping. This was a challenge to implement simply because documentation for the RAxML wrapper is minimally available and RAxML by nature has a more complex set of options and underlying structure. As mentioned previously, we did not use the MEGA 5 software in our tree generation for either profile 1 or 2 largely because BioPython does not provide a wrapper for the command line version of MEGA 5. Ultimately our goal currently is to provide a single interface solution, and therefore we

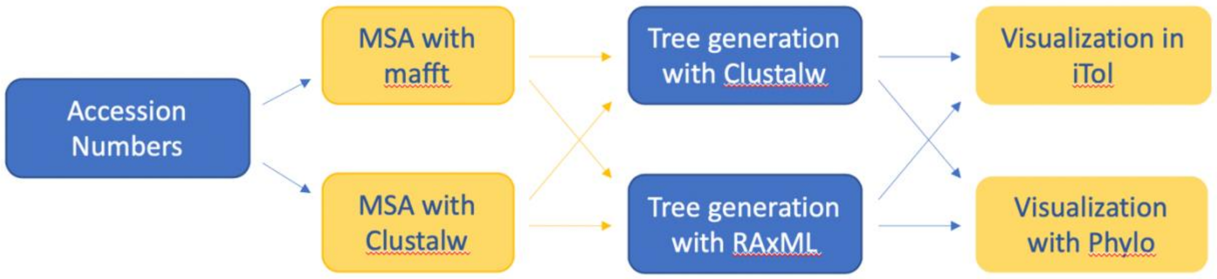


Figure 2. *Overview of the proposed pipeline as a flowchart of the possible routes for analysis and visualization*

did not include MEGA 5 in our analysis. It would be possible, however, to create a BioPython wrapper for MEGA 5 if it is deemed that our pipeline sufficiently increases efficiency and that MEGA 5 is an important option to have available within the pipeline.

Visualization of our trees was accomplished using iTol. In future iterations, iTol can be integrated into the Python workflow using HTTP requests, as iTol simply requires a POST request with the tree representation in the POST form data to grant access to the visualization of the tree. A subsequent GET request sent to the URL returned by the POST response header will display the tree to the user. However, we believe a more pythonic method is to utilize the Phylo package for visualizing the tree within BioPython, which is integrated with MatPlotLib (Talevich, et. al). This method is currently functional within our application, but further development is needed to make the displayed trees visually effective.

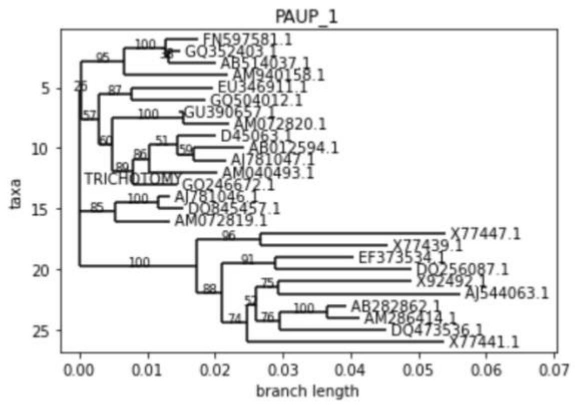


Figure 3. *Visualization of the profile 1 tree using the BioPython Phylo package draw function. Matplotlib integration allows for fine-tuning of the output, to be developed in future versions.*

Conclusions

The development of a single-interfaced pipeline capable of performing sequence acquisition, MSA, ML tree generation and visualization has the potential to increase the efficiency of these tasks immensely. Currently we present a framework for such an application that can be used to accurately reproduce previously published phylogenetic trees. Future development of the interface will seek to address the limited functionality while maintaining a user-friendly and efficient workflow.

Data Availability

The accession numbers used to generate both profiles are available from NCBI nucleotide database as EU346911, GQ504012, AM940158, AB514037, FN597581, GQ352403, AJ781046, DQ845457, AM072819, GU390657, AM072820, AM040493, GQ246672, AJ781047, D45063, AB012594, X77447, X77439, EF373534, DQ256087, X77441, X92492, AJ544063, DQ473536, AB282862, and AM286414. See the supplemental data for Python code.

References

Laviad et al. Standards in Genomic Sciences (2015) 10:21 High quality draft genome sequence of Leucobacter chironomi strain MM2LBT (DSM 19883T) isolated from a Chironomus sp. egg mass.

Halpern M, Shakéd T, Pukall R, Schumann P. Leucobacter chironomi sp. nov., a chromate-resistant bacterium isolated from a chironomid egg mass. *Int J Syst Evol Microbiol*. 2009;59(Pt 4):665-670. doi:10.1099/ijs.0.004663-0

Stamatakis, Alexandros. The RAxML v8.2.X Manual. *Heidelberg Institute for Theoretical Studies.* July 20, 2016. https://cme.h-its.org/exelixis/resource/download/NewManual.pdf

Talevich, E., Invergo, B.M., Cock, P.J. *et al.* Bio.Phylo: A unified toolkit for processing, analyzing and visualizing phylogenetic trees in Biopython. *BMC Bioinformatics* **13,** 209 (2012). https://doi.org/10.1186/1471-2105-13-209

Appendix

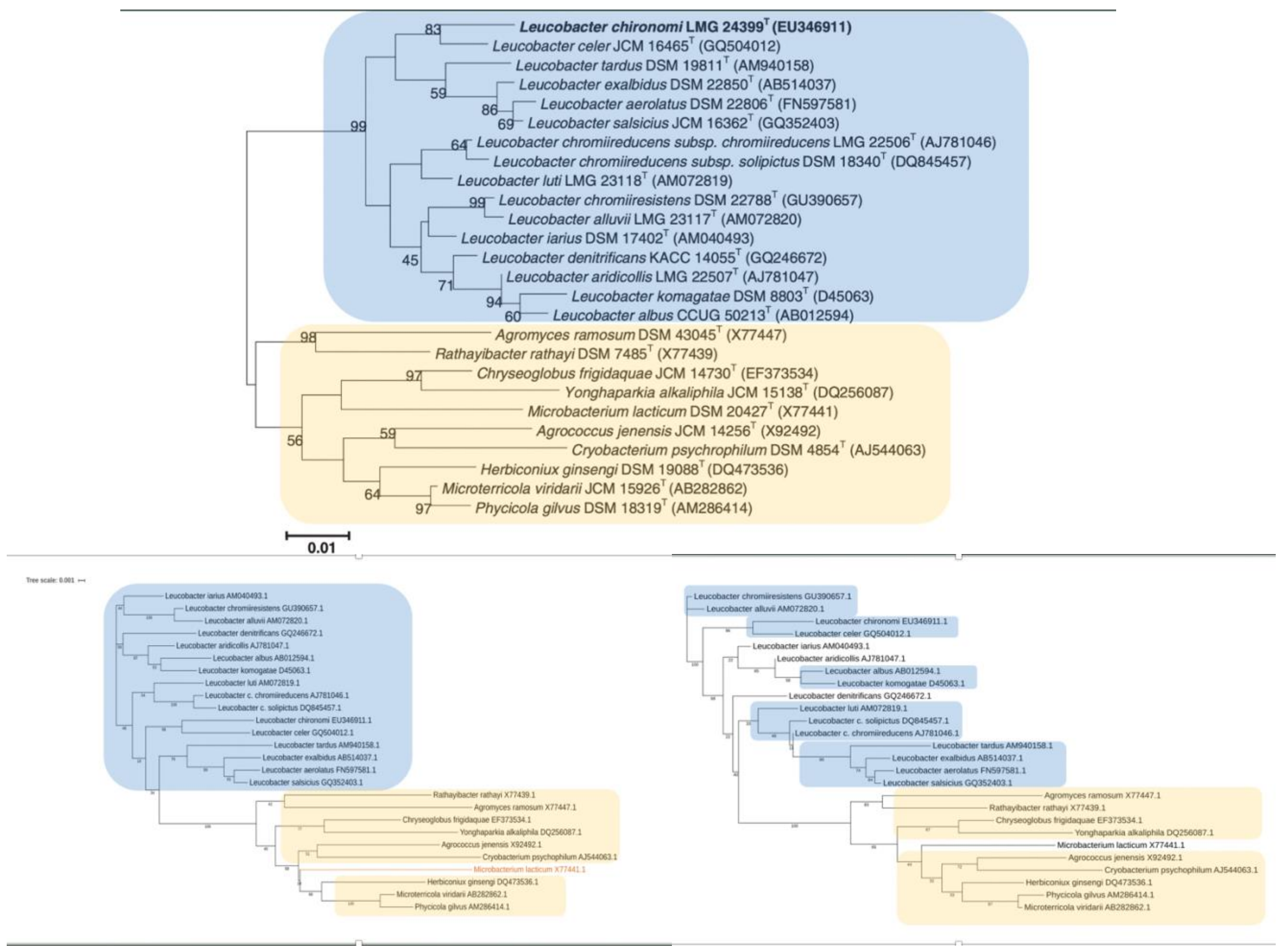

Leucobacter chironomi LMG 24399^T (EU346911)
Leucobacter celer JCM 16465^T (GQ504012)
Leucobacter tardus DSM 19811^T (AM940158)
Leucobacter exalbidus DSM 22850^T (AB514037)
Leucobacter aerolatus DSM 22806^T (FN597581)
Leucobacter salsicius JCM 16362^T (GQ352403)
Leucobacter chromiireducens subsp. chromiireducens LMG 22506^T (AJ781046)
Leucobacter chromiireducens subsp. solipictus DSM 18340^T (DQ845457)
Leucobacter luti LMG 23118^T (AM072819)
Leucobacter chromiiresistens DSM 22788^T (GU390657)
Leucobacter alluvii LMG 23117^T (AM072820)
Leucobacter iarius DSM 17402^T (AM040493)
Leucobacter denitrificans KACC 14055^T (GQ246672)
Leucobacter aridicollis LMG 22507^T (AJ781047)
Leucobacter komagatae DSM 8803^T (D45063)
Leucobacter albus CCUG 50213^T (AB012594)
Agromyces ramosum DSM 43045^T (X77447)
Rathayibacter rathayi DSM 7485^T (X77439)
Chryseoglobus frigidaquae JCM 14730^T (EF373534)
Yonghaparkia alkaliphila JCM 15138^T (DQ256087)
Microbacterium lacticum DSM 20427^T (X77441)
Agrococcus jenensis JCM 14256^T (X92492)
Cryobacterium psychrophilum DSM 4854^T (AJ544063)
Herbiconiux ginsengi DSM 19088^T (DQ473536)
Microterricola viridarii JCM 15926^T (AB282862)
Phycicola gilvus DSM 18319^T (AM286414)
0.01
Tree scale: 0.001

Figure A1. *Original Fig. 1 phylogenetic tree from Laviad, et. al. (top). Profile 1 reproduction (bottom left), and Profile 2 reproduction of the original figure.*