




Taxonomy Resolver: A Python package for building and filtering taxonomy trees

Fábio Madeira ¹✉, Nandana Madhusoodanan ¹, Joonheung Lee ¹, Alberto Eusebi ¹, Ania Niewielska ¹, and Sarah Butcher ¹

¹ European Molecular Biology Laboratory, European Bioinformatics Institute (EMBL-EBI), Wellcome Trust Genome Campus, Hinxton, Cambridge CB10 1SD, UK ✉ Corresponding author

DOI: [10.21105/joss.07604](https://doi.org/10.21105/joss.07604)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Charlotte Soneson](#) 

Reviewers:

- [@ryneches](#)
- [@tbrittoborges](#)

Submitted: 15 October 2024

Published: 13 February 2025

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

Taxonomy classification provides an important source of information for studying biological systems. It is a key component for many areas of biological sciences research, particularly genetics, evolutionary biology, biodiversity and conservation ([Sandall et al., 2023](#)). Common ancestry, homology and conservation of sequence and structure are all central ideas in biology that are directly related to the evolutionary history of any group of organisms ([Ochoterena et al., 2019](#)). The National Center for Biotechnology Information (NCBI) Taxonomy ([Schoch et al., 2020](#)) provides a curated classification and nomenclature for all the organisms in the public sequence databases, across the taxonomic ranks (i.e. Domain, Kingdom, Phylum, Class, Order, Family, Genus and Species).

Here we describe Taxonomy Resolver, a Python module and command-line interface (CLI) application for building and filtering taxonomy trees based on the NCBI Taxonomy. Taxonomy Resolver streamlines the process of manipulating trees, enabling fast tree traversal, searching and filtering.

Statement of need

The NCBI Taxonomy Database ([Schoch et al., 2020](#)) provides a hierarchically arranged list of organisms across all domains of life found in the sequence databases. Tree filtering, i.e. generation of tree subsets, referred to as subtrees, has various applications for sequence analysis, particularly for reducing the search space of sequence similarity searching algorithms. A sequence dataset composed of sequences from diverse taxa can be more quickly searched if only a subset of sequences which belong to taxonomies of interest are selected.

The NCBI BLAST+ suite is the most widely used toolset in bioinformatics for performing sequence similarity search ([Camacho et al., 2009](#)). The suite provides a Bash script (`get_species_taxids.sh`) to convert NCBI Taxonomy identifiers (TaxIDs) or text into TaxIDs suitable for filtering sequence searches. While this is a useful utility, it only works with sequences submitted to GenBank or other NCBI-hosted databases, and more importantly, it relies on making API calls via Entrez Direct (EDirect) ([Kans, 2024](#)). EDirect requires an internet connection and does not scale well when working with large sequence datasets. Other general-purpose tree libraries exist for Python (e.g. `anytree` ([Cofe Code and contributors, 2024](#)) and `bigtree` ([Kay Jan W. and contributors, 2024](#))) and R (e.g. `ggtree` ([Yu et al., 2017](#))), but they do not support the core features provided by Taxonomy Resolver or focus mainly on tree visualisation. The development of Taxonomy Resolver started in 2020 and aims to provide user-friendly interfaces for working directly with the NCBI Taxonomy hierarchical dataset.

Features

Taxonomy Resolver has been developed with simplicity in mind and it can be used both as a standard Python module or as a CLI application. The main tasks performed by Taxonomy Resolver are:

- **downloading** the NCBI Taxonomy classification hierarchy “dump” from the NCBI FTP server
- **building** complete taxonomy tree data structures or partial trees, i.e. subtrees
- **searching** particular TaxIDs at any level of the taxonomy hierarchy, performing fast tree traversal
- **validating** TaxIDs against the NCBI Taxonomy or any given subtree
- **generating** taxonomy lists that compose any subtree, at any level of the taxonomy hierarchy
- **filtering** a tree based on the inclusion and/or exclusion of certain TaxIDs
- **writing and loading** tree data structures using Python’s object serialisation
- **generating** partial and complete tress in NEWICK format

Implementation

A taxonomy tree is a hierarchical structure that can be seen as a collection of deeply nested containers - nodes connected by edges, following the hierarchy, from the parent node - the root, down to the children nodes - the leaves. An object-oriented programming (OOP) tree implementation based on recursion typically scales poorly for large trees, such as the NCBI Taxonomy, which is composed of >2.6 million nodes. To improve performance, Taxonomy Resolver represents the tree structure following the Nested Set Model, which is a technique developed to represent hierarchical data in relational databases lacking recursion capabilities. This allows for efficient and inexpensive querying of parent-child relationships. The full tree is traversed following the Modified Preorder Tree Traversal (MPTT) strategy (Celko, 2004), in which each node in the tree is visited twice. In a preorder traversal, the root node is visited first, then recursively a preorder traversal of the left subtree, followed by a recursive preorder traversal of the right subtree, in order, until every node has been visited. The modified strategy allows capturing the ‘left’ and ‘right’ (*lft* and *rgt*, respectively) boundaries of each subtree, which are stored as two additional attributes. Finding a subtree is as simple as searching for the nodes of interest where $lft > node's\ lft$ and $rgt < node's\ rgt$. Likewise, finding the full path to a node is as simple as searching for the nodes where $lft < node's\ lft$ and $rgt > node's\ rgt$. Traversal attributes, depth and node indexes are captured for each tree node and are stored as a pandas DataFrame (The pandas development team, 2024).

Taxonomy Resolver has been developed to take advantage of the Nested Set Model tree structure, so it can perform fast validation and create lists of taxa that compose a particular subtree. Inclusion and exclusion lists can also be seamlessly used to produce subset trees with wide applications, particularly for sequence similarity search. Taxonomy Resolver has been used in production since 2020, serving thousands of users every month. It enables taxonomy filtering features for NCBI BLAST+ provided by the popular EMBL-EBI Job Dispatcher service, available from <https://www.ebi.ac.uk/jdispatcher/sss/ncbiblast> (Madeira et al., 2024).

Acknowledgements

We would like to thank current and past members of the EMBL-EBI for their continued support. We would like to also thank EMBL and its funders.

References

- Camacho, C., Coulouris, G., Avagyan, V., Ma, N., Papadopoulos, J., Bealer, K., & Madden, T. L. (2009). BLAST+: Architecture and applications. *BMC Bioinformatics*, 10, 421. <https://doi.org/10.1186/1471-2105-10-421>
- Celko, J. (2004). Chapter 4 - Nested Set Model of Hierarchies. In J. Celko (Ed.), *Joe Celko's Trees and Hierarchies in SQL for Smarties* (pp. 45–99). Morgan Kaufmann. <https://doi.org/10.1016/B978-155860920-4/50005-2>
- Cofe Code and contributors. (2024). Anytree: Python tree data library. In *GitHub repository*. GitHub. <https://github.com/c0fec0de/anytree>
- Kans, J. (2024). Entrez Direct: E-utilities on the Unix Command Line. In *Entrez Programming Utilities Help [Internet]*. National Center for Biotechnology Information (US). <https://www.ncbi.nlm.nih.gov/books/NBK179288/>
- Kay Jan W. and contributors. (2024). BigTree: Tree implementation and methods for python, integrated with list, dictionary, pandas and polars DataFrame. In *GitHub repository*. GitHub. <https://github.com/kayjan/bigtree>
- Madeira, F., Madhusoodanan, N., Lee, J., Eusebi, A., Niewielska, A., Tivey, A. R. N., Lopez, R., & Butcher, S. (2024). The EMBL-EBI Job Dispatcher sequence analysis tools framework in 2024. *Nucleic Acids Research*, 52(W1), W521–W525. <https://doi.org/10.1093/nar/gkae241>
- Ochoterena, H., Vrijdaghs, A., Smets, E., & Claßen-Bockhoff, R. (2019). The Search for Common Origin: Homology Revisited. *Systematic Biology*, 68(5), 767–780. <https://doi.org/10.1093/sysbio/syz013>
- Sandall, E. L., Maureaud, A. A., Guralnick, R., McGeoch, M. A., Sica, Y. V., Rogan, M. S., Booher, D. B., Edwards, R., Franz, N., Ingenloff, K., Lucas, M., Marsh, C. J., McGowan, J., Pinkert, S., Ranipeta, A., Uetz, P., Wieczorek, J., & Jetz, W. (2023). A globally integrated structure of taxonomy to support biodiversity science and conservation. *Trends in Ecology & Evolution*, 38(12), 1143–1153. <https://doi.org/10.1016/j.tree.2023.08.004>
- Schoch, C. L., Ciufo, S., Domrachev, M., Hotton, C. L., Kannan, S., Khovanskaya, R., Leipe, D., Mcveigh, R., O'Neill, K., Robbertse, B., Sharma, S., Soussov, V., Sullivan, J. P., Sun, L., Turner, S., & Karsch-Mizrachi, I. (2020). NCBI Taxonomy: A comprehensive update on curation, resources and tools. *Database: The Journal of Biological Databases and Curation*, 2020, baaa062. <https://doi.org/10.1093/database/baaa062>
- The pandas development team. (2024). *Pandas-dev/pandas: Pandas*. Zenodo. <https://doi.org/10.5281/zenodo.13819579>
- Yu, G., Smith, D. K., Zhu, H., Guan, Y., & Lam, T. T.-Y. (2017). ggtree: An R package for visualization and annotation of phylogenetic trees with their covariates and other associated data. *Methods in Ecology and Evolution*, 8(1), 28–36. <https://doi.org/10.1111/2041-210X.12628>