BSc Bioinformatics

Integrating ARISE fungal ITS data in the MycoDiversity Database for biodiversity analysis

Daniël Zee (s2063131)

Supervisors:
Fons Verbeek, Irene Martorelli & Barbara Gravendeel

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

15/01/2025

Abstract

This thesis investigates the integration of ARISE fungal ITS data into the MycoDiversity Database (MDDB) to facilitate large-scale biodiversity and phylogenetic analyses. The ARISE project provides high-quality fungal ITS data generated using Illumina sequencing, which offers advantages in sequence quality compared Roche 454 sequencing. The current truncation value of 250bp used for sequences in MDDB is therefore not strictly necessary for this data. This research analyses the effects of sequence truncation on biodiversity metrics and phylogenetic diversity on the ARISE data, with the goal of making a final decision for the truncation value for this data, and subsequently integrating it in MDDB. Utilizing pipelines designed for MDDB for processing the sequence data, assigning taxonomies and performing phylogenetic placement, the study demonstrates that truncation to 250bp does not significantly compromise biodiversity insights while ensuring compatibility with existing MDDB data. The study also includes the implementation of a server-side tool to enable user-friendly phylogenetic placement analysis directly on the MDDB server, which can be used on both the ARISE data and samples from other studies currently present in MDDB.

Contents

1 Introduction

This bachelor thesis is written as part of the bioinformatics program at Leiden Institute of Advanced Computer Science (LIACS), at Leiden University.

The aim of this research was to integrate the ARISE fungal ITS data from Naturalis in the Mycodiversity database, and implement tools in the database that allow for phylogenetic diversity analysis on this data and on data previously present in the database. In order to motivate this research, we will first provide an introduction to fungal biodiversity analysis, the MycoDiversity database and phylogenetics.

1.1 Fungal biodiversity

Fungi are one of the largest groups in the domain of the eukaryotes, and therefore play a big role in Earth's ecosystems and biodiversity. They are most abundantly present in soil, where they decompose organic matter in the environment to provide nutrients that can be used by other organisms [FHBJ18]. The diversity and activity of fungi in soil is regulated by other organism, such as plants and other fungi, together with factors such as soil pH, moisture, salinity and temperature. The abundance of fungal species in soil is therefore an important observation for determining soil health and fertility. While recent studies estimate the size of the fungal kingdom between 2 and 6 million species, only 2-8% are believed to be discovered [HL17][THM+14]. There is therefore much room for discovery and identification of fungal species in soil samples around the world.

1.1.1 Metabarcoding

In the past, morphological traits like sporocaps where used to identify fungal species in soil samples. These days however, a relatively new technique called DNA metabarcoding has become the main technique to achieve this. Metabarcoding identifies species in a sample based on the presence of a barcode gene. This type of analysis has become possible because of the introduction of next generation sequencing (NGS) methods, allowing for high-throughput and cost effective sequencing of soil samples. An effective barcode gene should be a standardized short DNA sequence that can easily be generated and characterized for all species. These genes are therefore chosen to have as few intraspecific and as much interspecific variation as possible and contain highly conserves flanking sites in order to easily extract the gene from the sample using PCR primers [KE08]. Barcode genes can be used for DNA barcoding to identity a specific species in a sample, but when we aim to identify whole communities of species at a time, do we call this metabarcoding [PH20]. Figure 1 shows an overview of the general workflow of an DNA metabarcoding study. The DNA is extracted from the environment samples, amplified with PCR using primers specifically designed to target the barcode gene, and finally sequenced using the many available NGS methods.

Not all groups of organisms can be easily identified using the same barcodes. A region of the mitochondrial gene, CO1, is used as a barcode for animals, but proved to be difficult to amplify in fungi. For fungi, the internal transcribed spacer (ITS) region, located on the nuclear ribisomal RNA cistron, proved to have the highest probability of successful identification of a broad range of fungi [SSH+12]. Figure 2 shows an overview of the ITS region. The ITS region consists of two

spacer regions, ITS1 and ITS2. For fungal metabarcoding studies, only one of the spacer regions is normally used. It has been shown that both to a large extent yield similar results when used as barcodes for fungi [BKN+13].

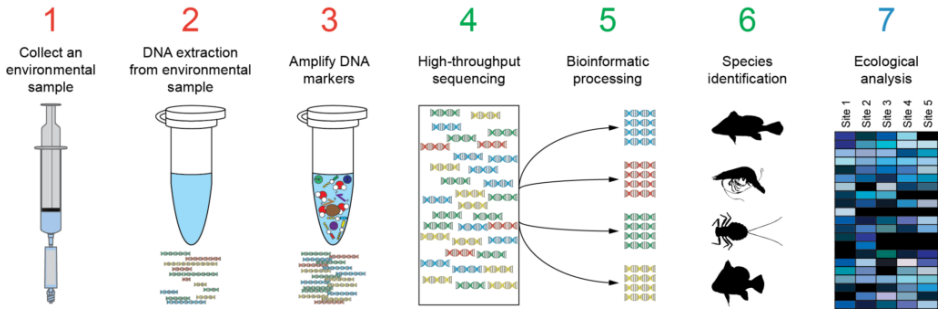

Figure 1: General process of a DNA metabarcoding study [Source: Nature Metrics]

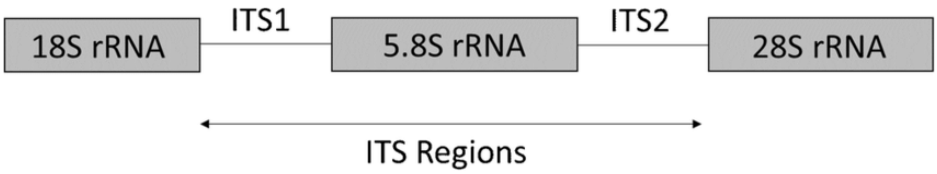

Figure 2: Picture of the ITS region as spacers between the ribosomal subunit sequences [Source: Genohub]

Since many studies have generated ITS data of Fungi, they have been made accessible in large online databases. The UNITE database [1] is regarded as the main reference database for ITS sequences. It clusters similar sequences together to form operational taxonomic units (OTUs), which UNITE calls species hypotheses (SHs) [AHNL+10]. Each SH is mapped to a taxonomy, for all taxonomic ranks from phylum to species. Is is however possible for a SH to have undefined taxonomic information at one or more ranks, in which case they are classified as *Incertae sedis*.

1.1.2 MycoDiversity DataBase

While the UNITE database provides a reference for assigning taxonomic information to ITS sequences, is does not provide the means for analysing the global distribution of fungal species. The MycoDiversity DataBase (MDDB) [2] is a joint project between Naturalis and Leiden Institute of Advanced Computer Science (LIACS), with the goal of allowing the study of biodiversity patterns of fungi in space and time. In MDDB, information from 25 publicly available fungal metabarcoding studies are included, allowing the data from the different studies to be directly comparable. MDDB achieves this by uniformly processing the raw DNA sequencing data from the individual studies, in addition with descriptive information of the samples and location data. This data is available in sequence read archives, a big example being the National Center for Biotechnology Information (NCBI) Sequence Read Archive (SRA) [SCS09]. The pipelines created for MDDB can extract this

[1] https://unite.ut.ee/
[2] https://mycodiversity.liacs.nl/

data automatically from SRA, and perform uniform curation before integration in the database. Figure 3 shows the UML data model of MDDB.

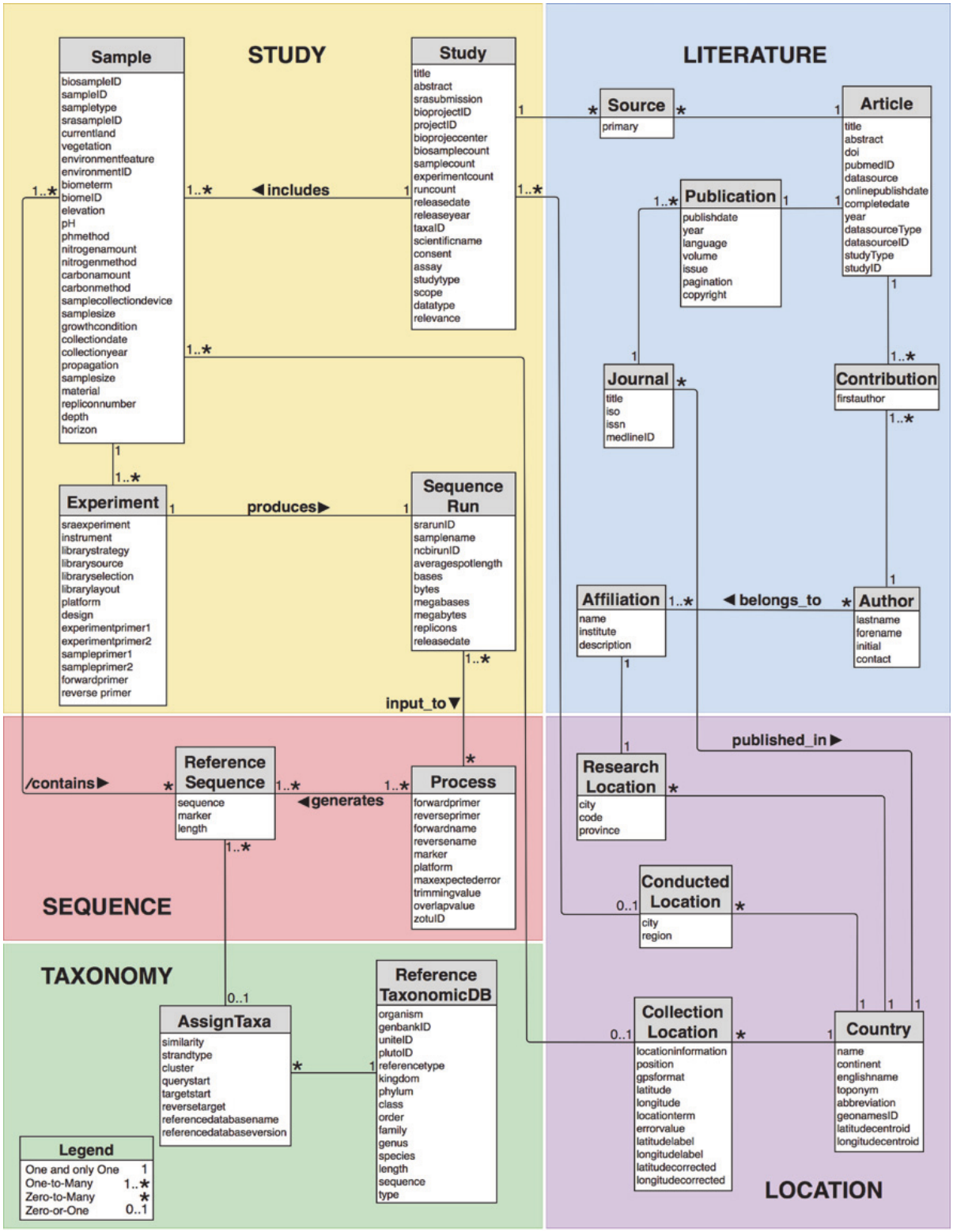

Figure 3: UML data model of the MDDB. Source: [MHK+20]

The raw sequence data from the samples are processed to form Zero-radius Operational Taxonomic Units (ZOTUs), which are OTUs with a clustering threshold of 100%, instead of the more typical 97% [SG94] . This means that two different ITS sequences will always be seen as different taxonomic units. This choice was made because ZOTUs have the advantage of being directly comparable between datasets without the use of reclustering [RHNM+14], which is the novelty of MDDB. Resulting ZOTUs are mapped to theit most similar SH in UNITE to derive the taxonomy information for each sequence. Before this thesis, only the sequence run files belonging to the SRA Study SRP043706 were processed and included in the sequence and sample tables in MDDB. The remaining 24 studies have thus far only be included in the literature section.

MDDB is built using the database management system MonetDB. This is a column-oriented system, where the data is stored in columns instead of rows. This type of system excels in read-heavy workloads, as they only retrieve the columns relevant to a query, allowing it to quickly retrieve large data quantities [IGN+12].

1.1.3 ARISE

The goal of MDDB is to provide a framework for retrieving data from collections of fungal metabarcoding studies, and can be further extended with the latest studies using the tools for integration and curation. An example of a research project that will yield new, unseen fungal ITS metabarcoding data is the ARISE project [3]. This project is an initiative by Naturalis with the aim of creating a mapping of all Dutch species and creating an infrastructure to recognize them. The new data includes fungal ITS sequence data from soil at specified locations around Leiden. The results from this study have not been formally published yet, nor are samples publicly available in SRA. The data has however been made available for this thesis, with the aim of analyzing the potential integration in MDDB.

1.2 Phylogenetic diversity

Biodiversity studies are often just focused on studying the number and distribution of species found in a specified location. While great work has been done studying patterns in species diversity, this approach has limitations for achieving a complete measure of biodiversity. This is because biodiversity can be seen than more than species diversity alone. For example, two samples with the same number of reported species can vary in the evolutionary background of the species and their function in their ecosystems. Biodiversity can therefore be best summarized by the interrelationships between three primary components: species diversity, functional diversity, and phylogenetic diversity [Swe11]. Figure 4 shows the interconnected relationships between these components.

Phylogenetic diversity (PD) described the evolutionary distance between a community of species. Many traits in species show a phylogenetic signal, suggesting that PD can be used as an estimator for functional diversity in an ecosystem [SCM+12]. Measures of PD can therefore be a insightful addition to species diversity in most biodiversity studies, by providing valuable insights into the evolutionary relationships between species.

[3] https://www.naturalis.nl/en/science/arise-knowing-nature-in-the-netherlands

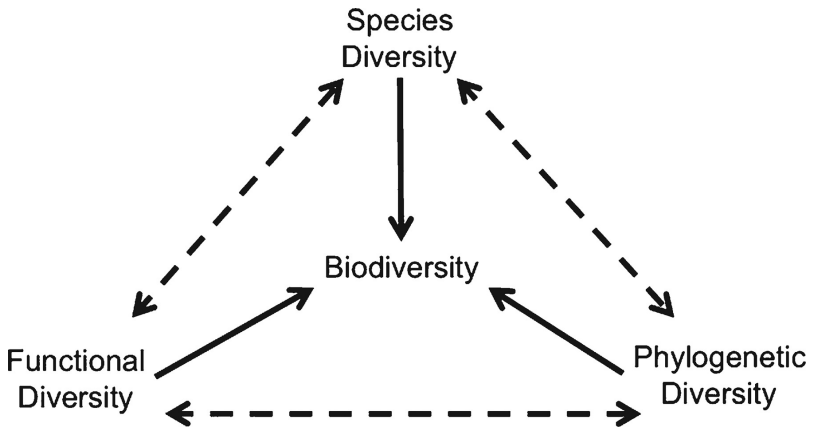

Figure 4: Triangle showing the interconnected relationships between the components of biodiversity. Source: [Swe11]

1.2.1 Phylogenetic trees

Phylogenetic analysis results are usually displayed in branching diagrams, called phylogenetic trees, also called phylogenies or evolutionary trees. In the past, these trees were generally created for organisms based on morphological traits. But with the rise in available data from DNA barcoding studies, phylogenetic trees can now be constructed based on genetic differences alone [ZJ12].

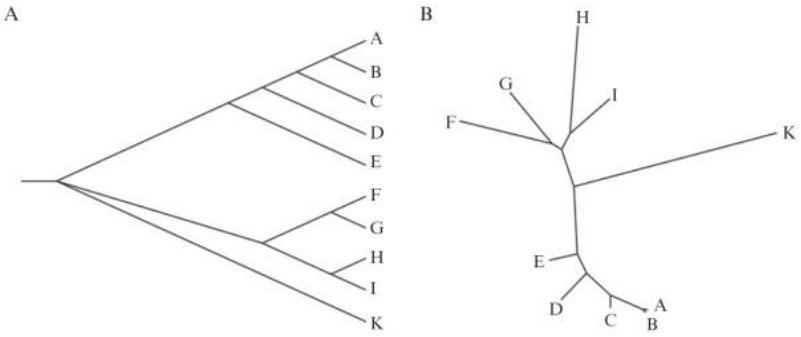

Figure 5: Examples of phylogenetic trees, including a rooted (A) and unrooted (B) tree. Source: [ZJ12]

Figure 5 shows examples of phylogenetic trees. Trees consist of three components: leaves, nodes and branches. The leaves represent different species, or groups of species, called taxa if they represent a formally named group. The nodes of the tree, defined as the branching points, represent the last common ancestor of the two subtrees descended from that node. Branches between two nodes are classified as internal branches, while branches from a node to a leaf are classified as external branches [Bau08]. Trees can either be rooted or unrooted, with rooted trees signifying that one branch corresponds to the common ancestor of all taxa in the tree.

A phylogenetic tree can be depicted in multiple ways, as only the topology is what differentiates them. For example, we can swap the positions of leaves A and B in tree A in figure 5 without changing its topology. When not indicated, the branch lengths convey no information. In practice

however, branch lengths are usually used to indicate the evolutionary distance between two nodes or a node and a leaf. [Ram18]

Phylogenetic trees are constructed from a set of homologous DNA sequences, by first performing multiple sequence alignment on them. Accurate alignment results form the basis for inferring evolutionary relationships. The alignment is then provided to an algorithm for phylogenetic tree inference. There are two main categories of methods for phylogenetic tree inference: distance-based models and character-based models [ZZZ+24].

1.2.2 Phylogenetic placement

Phylogenetic placement is class of methods used to place unknown sequences, called query sequences, onto a fixed phylogenetic reference tree. The query sequences are not added to the tree, but merely mapped to the branches they are most likely to fit in terms of evolutionary distance to the sequences in the reference tree [CSDB22]. Phylogenetic placement can be used for taxonomic assignment of the query sequences, if the taxonomies of the reference tree are known, or for visualizing the evolutionary distribution of the query sequences on the reference tree. Phylogenetic placement is often used in metabarcoding studies. These studies produce sequence sets that are either too large in number or too short in length to infer comprehensive phylogenetic trees. Phylogenetic placement methods are therefore a way to still be able to derive measures of phylogenetic diversity from these sequences. The query sequences also need not be placed on a single branch, but instead multiple potential branches can be derived, each with a different probability.

A sample can contain multiple queries, which are then first independently aligned to the reference tree. The alignment is then, together with the reference tree, provided to a phylogenetic placement algorithm. There are two main categories of phylogenetic placement algorithms: Maximum likelihood methods and distance-based methods.


```
{
    "tree": "((A:0.2{0},B:0.09{1}):0.7{2},C:0.5{3}){4};",
    "placements":
    [{
        "p": [
            [1, 22578.16, 0.777385, 0.004132, 0.0006],
            [0, 22580.15, 0.107065, 0.000009, 0.0153]
        ],
        "n": ["fragment1", "fragment2"]
    }, {
        "p": [[2, 22576.46, 1.0, 0.003555, 0.000006]],
        "nm": [["fragment3", 1.5], ["fragment4", 2]]
    }],
    "fields": [
        "edge_num", "likelihood", "like_weight_ratio",
        "distal_length", "pendant_length"
    ],
    "metadata": {
        "invocation": "epa-ng --ref-msa $REF_MSA
            --tree $TREE --query $QRY_MSA --model $MODEL"
    },
    "version": 3
}
```


Figure 6: Example of a jplace file showing the placement results on a reference tree

The output of phylogenetic placement algorithms is usually stored in the jplace format [MHGS12]. Figure 6 shows an example of such an output file. This format is based of the widely used json format. The `tree` field contains the topology of the reference tree using a custom augmentation of the Newick format, where each branch is annotated with a unique number. The placements field contains for each query sequence a list of placements, that show for each placement from left to right: the edge number, the likelihood, the likelihood weight ratio (LWR), the distal length and the pendant length. The pendant length is the length the new branch containing the query sequence would have if it were to be placed in the tree at that edge.

1.3 Research questions

The aim of this research is to integrate the ARISE fungal ITS data into MDDB in a way that allows for accurate and reliable biodiversity analysis between the sampled locations and other studies. The main research question can therefore be formulated as:

> *RQ1: How can the ARISE fungal ITS data best be integrated in the MycoDiversity Database for biodiversity analysis?*

All ZOTUs currently present in MDDB are truncated to 250bp, due to the increased expected error rate on longer sequences using Roche 454 sequencing and because 250bp covered for all fungal species a good enough spectra for ITS1 and ITS2 regions. [Mar24]

The ARISE data has been sequenced using an Illumina platform, using paired-end sequencing. Illumina sequencing has been shown to have lower overall error rate compared to Roche 454 sequencing [LBM15]. Paired-end sequencing also decreases the expected error rate, as each PCR fragment is sequenced twice, once in every direction.

Because of these facts, truncating the ARISE ZOTUs to 250bp is not strictly necessary for quality reasons. A decision therefore needs to be made on the truncating length for this data by studying the difference in species and phylogenetic diversity at full length versus truncated to 250bp. This leads to the following sub-question:

> *RQ1.1: What effect does truncating the ARISE reference ZOTUs to 250bp have on biodiversity analysis?*

After deciding on the truncation value can the ZOTUs, together with the taxonomy, location and study information of the ARISE data be added to MDDB. Because the end goal is for users of the database to be able to perform their own large-scale biodiversity analyses of patterns in space and time, the final sub-question arises:

> *RQ1.2: How can biodiversity analysis of the ARISE fungal ITS data be performed on the database server?*

1.4 Thesis overview

This section gives an overview of how this thesis is structured.
Chapter 2 contains descriptions of the data, software and hardware used for this research.
Chapter 3 contains an in-depth explanation of the implementation design. Chapter 4 contains an overview of the experiment results.
Chapter 4 contains the results of the experiments.
Chapter 5 contains the conclusion and further research.

2 Material & Methods

This chapter describes the data, software and hardware which was used for this research.

2.1 Data

The main data used for this research is the sequencing data from the ARISE soil samples. This data was provided by Dr. Rutger Vos from Naturalis. For phylogenetic placement we will be using a reference tree based on the UNITE backbone tree created by *Luuk Romeijn and Casper Carton (2022)* [CR22]. The following subsections describe the data in more detail.

2.1.1 ARISE samples

The ARISE soil samples were collected in 2021 at three different locations around Leiden, chosen based on the vegetation of the soil. The three locations are:

1. 'Leidse Hout' (52.176954, 4.477630), Type of location: woods
2. 'Lentevreugd' (52.163225, 4.391914), Type of location: grassland
3. 'Berkheide' (52.164047, 4.392443), Type of location: dunes, sand

The design for sample collection used was suggested by *Arita and Rodriguez (2002)* [AR02] and described by *Gavito et al. (2019)* [GLMVP+19]. The total plot is divided in three subplots (S1-S3) of 80 x 80 meters, which are connected diagonally. Each subplot is then divided in 64 subplots of 10 x 10 meters, of which a sample was taken from 32, following a checkerboard pattern. The sampling grids for locations A and C are shown in figures 7 and 8, respectively.
DNA was extracted from the samples with the MagAttract Powersoil DNA KF kit (QIAGEN) and the KingFisher Flex System (ThermoFisher). A positive control sample was added, which came from the Naturalis collection (TH9240) and was identified as *Lactarius sp.*

A PCR was performed on the samples for the amplification of the ITS2 region. Following the suggestions of *Tedersoo (2014)* [TBP+14], a mix of five forward primers (ITS3NGS1-5) and one reverse primer (ITS4NGS) were used to increase the likelihood of matching all fungi species in the samples. Gel electrophoresis was performed on the PCR products to determine the effectiveness of DNA extraction. Location B did not show bands at the expected length, so these samples where excluded from sequencing.

Figure 7: Sampling grid for location A. Subplots marked red are not present in the sample data.

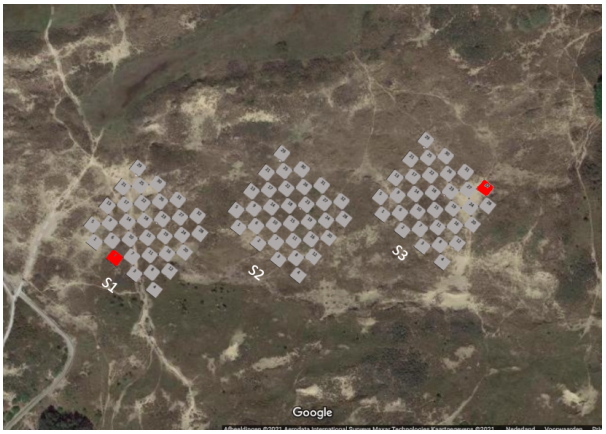

Figure 8: Sampling grid for location C. Subplots marked red are not present in the sample data.

A total of 190 samples were sent to BaseClear for sequencing. These include 95 samples from location A, 94 samples from location C, and the positive control sample. Sequencing was performed using the Illumina MiSeq system.

The resulting data from BaseClear contains a directory with the compressed raw sequence reads in FASTQ format for each sample, and a checksum file which can be used to verify the integrity of the data. There are a total of 7.451.149 read-pairs over all samples, with an average quality score of 32.83, resulting in 2.4 GB of compressed FASTQ files. The name of each FASTQ file contains a unique NBCLAB number, which can be mapped to a sample at one of the locations. The name also contains the location, the subplot, and an indicator stating if it is the forward (R1) or reverse (R2) read.

2.1.2 MDDB-phylogeny reference tree

Luuk Romeijn and Casper Carton (2022) [CR22] proposed a method to generate a phylogenetic reference tree using SH reference sequences in the UNITE database, with the goal of adding this tree to MDDB as a tool for phylogenetic diversity analysis. As generating a tree from all sequences at once proved to be too complex, a divide-and-conquer approach was chosen. All sequences were split in chunks based on taxonomic rank, and a tree is then generated for each chunk. Within each chunk, two representative sequences are selected using an alignment-free distance measure. These representative sequences from each chunk are then used to generate a representative tree. Finally, The forks in the representative tree for each chunk representatives are replaced by the chunk trees to generate the full backbone tree. The backbone creation algorithm accepts several parameters, for which two recommendations are given. The backbone tree generated using the first recommendation: `10.2_s3_4_1500_o1.0_a0_constr_localpair` is used for this research. In this tree, the chunks are separated by either the order or family rank. The files for this tree can be found on GitHub [4]. This tree is based on the UNITE QIIME release for Fungi (version 8.3) [5].

[4] https://github.com/luukromeijn/MDDB-phylogeny/tree/main/results/thesis%20results/10.2_s3_4_1500_o1.0_a0_constr_localpair

[5] https://doi.plutof.ut.ee/doi/10.15156/BIO/1264708

The backbone tree files are contained in three directories. The `discarded` directory contains FASTA files including the sequences which are not included in the tree, either because they are too long or too short, are too distant from their chunk representatives, form chunks that are too small to form a proper tree, or have undefined taxonomy up to splitting rank. The `chunks` directory contains aligned and unaligned FASTA files for each chunk together with the generated tree for each chunk. The `supertree` directory contains the FASTA file including all sequences in the final backbone tree, the final backbone tree file, and the aligned and unaligned FASTA files for the representative tree together with the tree file for the representative tree. The final backbone tree includes 23237 sequences, spread over 229 chunks.

For this research, the decision was made not to regenerate the tree using the most recent UNITE release at the time of writing (version 10.0), as recalculating the distance matrix for all sequences and regenerating every chunk tree can take more than 12 hours.

2.2 Software

The software used for this research comprises of a combination of pipelines available on the mycodiversity GitHub page [6], standalone sequence analysis tools, and the tools proposed by *Lena ten Haaft (2023)* [tH23] for phylogenetic placement on the MDDB-phylogeny reference tree.

For the implementation of the analysis, a combination of Bash scripts (version 5.1.16) and Python (version 3.10.12) were used manipulate the output of standalone tools using either the AWK or Python scripting languages. Analysis of the generated data was done in a Jupyter Notebook using the `pandas` package. The server-side phylogenetic placement tool was written in PHP (version 8.1.2), using the htmx library to allow for partial page reloads.

2.2.1 PROFUNGIS

The Processing of Fungal ITS Sequences (PROFUNGIS) pipeline is a pipeline developed specifically for MDDB. It downloads SRA reads and constructs a unique set of ZOTUs for each sample. The pipeline uses Snakemake as workflow management system to chain the different steps of the process, which are implemented by a combination of standalone sequence analysis tools, Python scripts and Bash scripts.

The pipeline requires the following parameters to be specified: the forward and reverse primers used during PCR amplification, which ITS subunit was sequenced (ITS1 or ITS2), the sequencing platform which was used (454, illumina, iontorrent) and a single or set of SRA sequences read IDs.

The starting script used to run PROFUNGIS validates the provided parameters values and creates a configuration file for the Snakemake workflow. This workflow then carries out the following steps of ZOTU construction:

1. **Filter Primers**: Cutting the provided primers from both ends of the reads.

[6] https://github.com/naturalis/mycodiversity

2. **Merge Reads**: Merging the forward and reverse reads if the sequencing platform is Illumina. If the platform is 454 or iontorrent, the reads are truncated to 250bp instead.

3. **Quality Filter**: The average estimated error is used to filter out low quality reads.

4. **Dereplicate**: The remaining reads are dereplicated.

5. **Discard Singletons**: Singleton reads are discarded.

6. **Donoise**: Create ZOTUs using the UNOISE3 algorithm. This performs error correction on the reads to predict correct biological sequences.

7. **ZOTU table**: Create a mapping table with the abundance of each ZOTU in the reads.

8. **Abundance Filter**: Filter out ZOTUs that occur less than 0.5% of the total reads

9. **Contamination Filter**: Filter out ZOTUs that do not have at least a 70% BLAST hit against the UNITE database.

Afterwards, the ZOTUs are located in a timestamped output directory containing subdirectories with the output files of steps 7, 8 and 9. Output files are named after their respective SRA sequence read IDs.

2.2.2 PROFUNGIS post processing

After generating the ZOTUs for all SRA read, they need to be incorporated in MDDB. The Reference Sequence table and the contains relationship table in the sequence section of the MDDB UML model therefore need to be updated to include the new ZOTUs. The PROFUNGIS post processing pipeline includes two Python scripts that achieve this goal. The `generate_zotu_ref1.py` script accepts a FASTA file containing the ZOTUs for a single SRA read and outputs two tables in csv format, `refseq_table_pk.csv` and `mapping_table_pk_zotu_srr.csv`, which correspond to the above stated database tables respectively. The `update_ref_map.py` script is used to add ZOTUs from a SRA read to an existing reference sequence table. The script will only add reference sequences which are new to the table and will otherwise map to the existing reference sequence primary key in the contains relationship table. Figure 9 shows an example of how both scripts can be used successively to create both tables for two SRA reads. When adding new ZOTU sequences to MDDB, only the `update_ref_map.py` script will be used, as the sequences will be added to the current tables contents of the database.

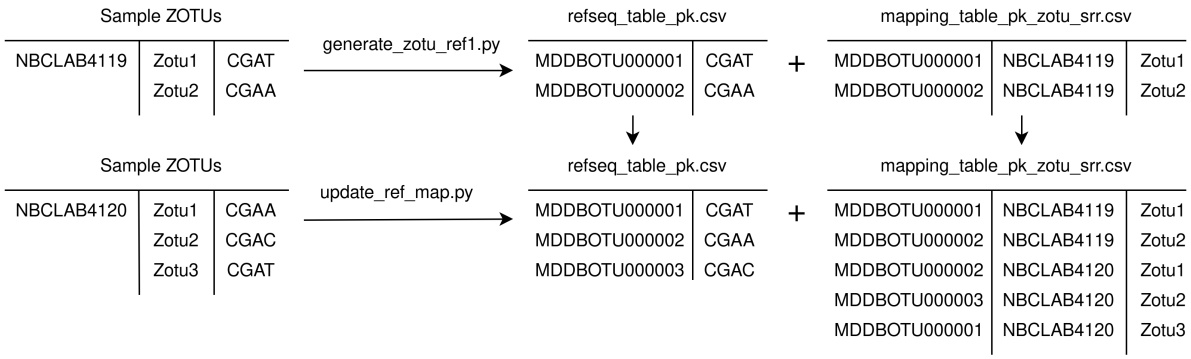

Figure 9: Example of the usage of both PROFUNGIS post processing scripts to create reference sequence tables from two SRA reads. As there are sequences in the second read that are already present in the reference sequence table, only the new ones are be added.

2.2.3 USEARCH

USEARCH is a sequence analysis tool developed by Robert Edger that implements many different algorithms in a single binary executable [Edg10]. The stable version at the time of writing is v11, which is closed source software. USEARCH has a detailed documentation page [7] where all its functionality in sequence analysis can be found.

The PROFUNGIS pipeline uses USEARCH algorithms for the truncation of sequences, quality filtering, singleton discarding, denoising and ZOTU table creation.

In this research, USEARCH will additionally be used for truncating the full length ARISE ZOTUs and the included global alignment algorithm will be used to map the ARISE ZOTUs to the UNITE database.

2.2.4 BLAST

Basic Local Alignment Search Tool (BLAST) is a program for finding local similarities between biological sequences, and calculates the statistical significance of matches. It can align DNA or protein sequences to standard databases, using the web BLAST interface [8]. Using BLAST+, the BLAST command line application, we can perform alignment searches using a locally created database [CCA+09]. BLAST+ version 2.12.0 was used for this research.

2.2.5 MAFFT

MAFFT (multiple alignment using fast Fourier transform) is a program for performing multiple sequence alignment of amino acids or DNA sequences. MAFFT was first released in 2002 and currently supports options for various alignment strategies, such as progressive methods, iterative refinement methods, and structural alignment methods for RNA [KS13]. For this research, MAFFT

[7] https://www.drive5.com/usearch/manual/
[8] https://blast.ncbi.nlm.nih.gov/Blast.cgi

version v7.526 was used for aligning query sequences to the base alignment of the phylogenetic reference tree described in section 2.1.2.

2.2.6 RAxML

RAxML (Randomized Axelerated Maximum Likelihood) is a program used to infer phylogenetic trees using a reference alignment. This is a maximum likelihood algorithm, which falls under the character-based models for phylogenetic tree inference. RAxML was used for generating the chunk trees and representative tree for the phylogenetic reference tree described in section 2.1.2. The results of these constructed trees in GitHub do however not include the `RAxML_info` files, which are required for the phylogenetic placement method used for this research. RAxML version 8.2.13 was therefore used during this research to regenerate the chunk trees for the reference tree, using the same parameters used by *Luuk Romeijn and Casper Carton (2022)* [CR22] for the first recommendation.

2.2.7 pplacer

pplacer is a software package for phylogenetic placement and subsequent visualization. pplacer can place a large number of sequences in parallel on a reference tree, with linear time and memory complexity [CIKA10]. The algorithm falls under the maximum likelihood category of phylogenetic placement methods, calculating the likelihood weight ratio of each placement by summing the likelihood scores and normalizing them to sum to one. The algorithm can also be run in Bayesian mode, where instead the posterior probability of each placement is calculated. pplacer version v1.1.alpha19 was used during this research for performing phylogenetic placement of MDDB sequences on the phylogenetic reference tree described in section 2.1.2. Only the Maximum Likelihood mode as used during the experiments. pplacer requires a reference package as input, containing information about the reference tree to be placed on. Taxtastic is a python package used to build and maintain such reference packages.

2.3 Hardware

The ARISE integration analysis, including all software mentioned in section 2.2, were run on a personal laptop with an Intel Core i7-6700HQ CPU (8 cores) and 8.0 GiB of memory. For integrating the ARISE data in MDDB, and querying the database, a SSH tunnel was used to connect to the MDDB server at LIACS. This server runs on an Intel(R) Xeon(R) x5355 CPU (8 cores) with 32GB memory.

3 Implementation

The implementation of this research can be split up in two independent sections: analyzing the effect of truncation on ZOTUs constructed from the ARISE data (3.1), followed by integrating in MDDB, and implementing tools in the MDDB server to allow end users to perform server-side phylogenetic placement on the MDDB reference tree using the ZOTUs in MDDB (3.2).

3.1 ARISE integration analysis

Figure 10 shows an overview of the method for constructing ZOTUs from the ARISE samples, followed by steps for empirically validating the assumption made in MDDB that truncating the ZOTUs to 250bp does not significantly impact the derived biodiversity within the data. This section contains a in-depth description of each step in this method.

For implementing all steps in this method, a fork was made of the original mycodiversity GitHub repository [9] where every sequential step can be performed using numbered Bash scripts. Instructions for running the scripts together with modifications and bug fixes for the PROFUNGIS pipeline are documented in the README file.

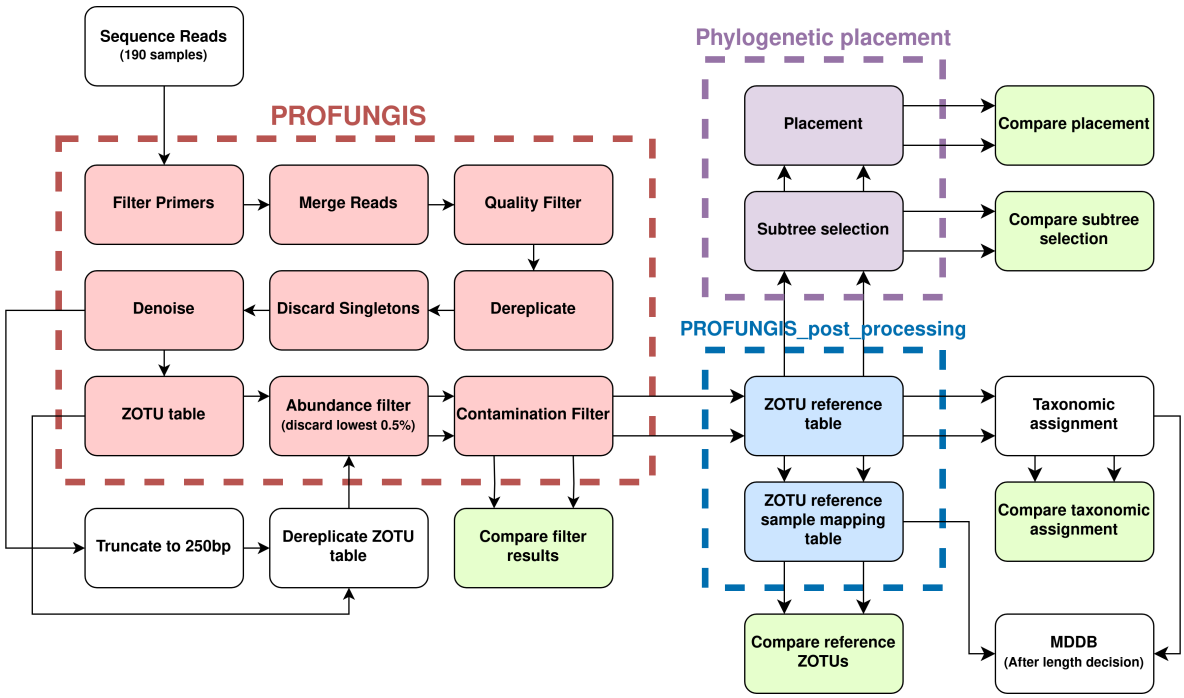

Figure 10: An overview of the full method used for analyzing the effect of truncation on the measured biodiversity in the ARISE data. The results from the comparison steps highlighted in green will be studied.

3.1.1 ZOTU construction

Before being able to construct ZOTUs using the PROFUNGIS pipeline, a few pre-processing steps need to be performed. First the provided checksum of the ARISE data is verified, to ensure that no sequence data has been corrupted. Afterwards the raw ARISE sequence files need to be extracted to the samples directory in PROFUNGIS. PROFUNGIS was designed to download SRA reads automatically, but has the option for using locally stored files, using the `-l` argument. A script was written to extract the compressed FASTQ files to the samples directory, resulting in a directory for each NBCLAB number containing the forward and reverse FASTQ files using the naming scheme scheme expected by PROFUNGIS. The NBCLAB numbers will therefore be used as unique

[9] https://github.com/SethGG/mycodiversity-arise

sample identifiers instead of SRA sequences read IDs. The script also extracts information on the sample location and subplot to a csv file, called `sample_mapping.csv`, which is used in later analysis for mapping samples to either location. Finally, the script generated a txt file listing all NBCLAB numbers of the extracted samples. This file is passed to the `-m` argument, which allows PROFUNGIS to process the provided samples in parallel. The forward and reverse primers were manually added to the `primers.data` file. The decision was made to only run PROFUNGIS on the ARISE data using the first forward primer, ITS3NGS1. This is because running PROFUNGIS on all five forward primers increases the runtime significantly and another pipeline developed by Naturalis[10], which was specifically designed for this data, showed that the default error margin used when cutting the primers from the reads is large enough to cover the difference of all forward primers. PROFUNGIS is then run with the following command:

```
$ python startPROFUNGIS.py -f ITS3NGS1 -r ITS4NGS -p illumina -l -m
↪  sample_list.txt
```

3.1.2 ZOTU truncation

After running running PROFUNGIS, the full-length ZOTU sets are generated before and after the sequential abundance and contamination filter steps. Because the ZOTUs are truncated to 250bp before these filter steps when using the 454 or iontorrent platform, the decision was made to truncate the ARISE ZOTUs before these steps and rerunning both filters after truncation.

Truncation of the ZOTUs was performed using USEARCH, the same method used in PROFUNGIS. After truncation, ZOTUs which originally were unique can become equal to other ZOTUs in the sample. A second USEARCH function is therefore called to find the set of unique ZOTUs after truncation, where only the first occurrence of each sequence is kept. This function also outputs a mapping for all ZOTU names before and after truncation. These two functions are run with the following commands:

```
$ usearch11 -fastx_truncate $zotus_full -trunclen 250 -fastaout $zotus_trunc
$ usearch11 -fastx_uniques $zotus_trunc -fastaout $zotus -tabbedout $zotus_derep
```

It is important to note that sequences smaller than the truncation length are discarded by USEARCH, so these ZOTUs will no longer be represented after the truncation process. The effect of discarding these ZOTUs on the measured biodiversity will play a big part in deciding the final truncation value for the ARISE data.

A Python script was written to then dereplicate the full length ZOTU tables by combining the occurrences of the ZOTUs which sequences have become equal after truncation.

Finally, the abundance and contamination filters are rerun to create the filtered truncated ZOTU sets.

[10] https://github.com/naturalis/arise-metabarcoding-biodiversity/blob/main/src/DADA2.R#L84-L97

3.1.3 ZOTU filter mapping

To allow analysis of the effects truncation has on the ZOTUs in combination with the abundance and contamination filter, a Python script was written that aggregates the results of these steps over all 190 samples. A total of three csv files are created. `truncate_mapping.csv` aggregates the mapping of ZOTU names before and after truncation and includes a indicator stating if the ZOTU was discarded during truncation. The `filter_mapping_full.csv` and `filter_mapping.csv` files have columns for every ZOTU in every samples stating if it passed the abundance and contamination filter, for the full-length and truncated ZOTUs respectively.

Because truncation discards a portion of the sequences and dereplication of the truncated ZOTUs increases the occurrence counts, the truncation process allows ZOTUs to pass the abundance filter which would not have at full-length. Figure 11 shows an example of the three scenarios in which this could happen. The fist scenario is when a ZOTU gets mapped to a ZOTU after truncation which already passed the abundance filter at full length (ZOTU4 in figure 11). In this case no new sequence information is added to the truncated ZOTU set. The second scenario is when multiple ZOTUs which do not pass the abundance filter at full-length map to a single truncated ZOTU, increasing its abundance above the cutoff value (ZOTUs 5 and 6 in figure 11). The last scenario is when a ZOTU now passed the abundance filter because of the lower abundance cutoff value, which is the result of the discarded sequences lowering the total abundance (ZOTU 3 in figure 11). In these last two scenarios there is new sequence information added to the truncated ZOTU set. For the remainder of this research, these ZOTUs will be referred to as "new" ZOTUs.

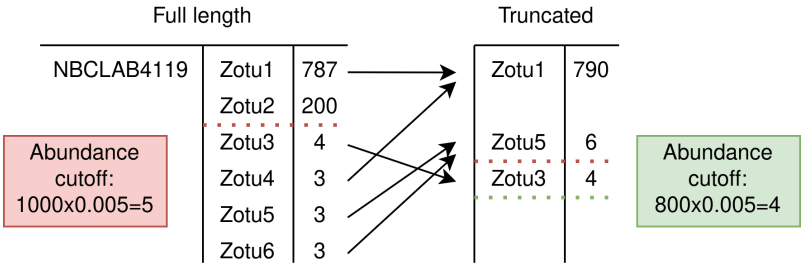

Figure 11: Example scenario showing the effect truncation has on which ZOTUs pass the abundance filter. The red dotted line shows the abundance cutoff.

3.1.4 Reference sequences

The next step is to create the reference sequence tables for both the full length and truncated ZOTU sets using the PROFUNGIS post processing scripts. The first sample was used to create the base tables using the `generate_zotu_ref1.py` script, and the `update_ref_map.py` script was subsequently used on the rest of the samples to construct the final tables.

Analyzing the reference sequence tables gives us insight in the sequence diversity between the sample locations, and how much of the specificity of sequences to a location changes after truncation.

3.1.5 Taxonomic assignment

Every reference sequence in MDDB is assigned a taxa in the way of a mapping to a SH in UNITE, only if a close enough match can be found. The same method for taxonomic assignment will be used during this research for comparing species diversity before and after truncation, as end users of MDDB will be using these mappings to perform biodiversity analysis.

The mapping of a reference sequence to a UNITE SH is done by performing a global alignment of the reference sequences against UNITE and taking the top identity hit. This is done using the USEARCH alignment algorithm included in the USEARCH tool. A script was written to convert the reference sequence tables to FASTA files, run the USEARCH algorithm, and map each sequence to its top hit in a csv file where taxonomic ranks are split to columns. USEARCH was run with the following command:

```
$ usearch11 -usearch_global $output_fasta -db
↪  Unite/sh_general_release_dynamic_04.04.2024.fasta -strand plus -id 0.80
↪  -userout $output_usearch -userfields query+id+target
```

Taxonomic assignment was performed on both the full-length and truncated reference sequences to allow comparison between the observed species diversity in both locations before and after truncation and the difference in taxa assigned to each individual reference sequence before and after truncation.

3.1.6 Phylogenetic placement

Finally, we study the difference in observed phylogenetic diversity before and after truncation. This analysis is performed using the method proposed by *Lena ten Haaft (2023)* [tH23]. This method performs phylogenetic placement for a set of sequences on the phylogenetic reference tree described in section 2.1.2. The method does not directly perform placement on the final backbone tree, as `pplacer` can not handle trees with more than 5000 leaves. We therefore perform placement on the subtrees derived from the chunks used to generate the final backbone tree.

The first step of the method uses BLAST to determine the chunk each sequence most likely belongs to. All full-length and truncated reference sequences are blasted to the full set of sequences present in the reference tree. The top 10 BLAST hits for each reference sequence are mapped to their respective chunk names and then used to decide the majority chunk. BLAST was run with the following command, where `backbone_blastdb/backbone` refers to the BLAST database created from the full set of sequences in the backbone tree:

```
$ blastn -query $output_fasta -db backbone_blastdb/backbone -out $output_blast
↪  -outfmt 6 -max_target_seqs 10
```

The BLAST results for both the full-length and truncated reference sequences were then studied by comparing the taxonomic splitting level of the derived majority chunks for each sequence with the assigned taxonomies using USEARCH.

For the second step, we perform phylogenetic placement using `pplacer` on the subtrees for the

chunks present among the majority chunks. The input for the placement on each subtree is a FASTA file containing the full length and truncated reference sequences to be placed in the subtree, aligned against the sequences in the subtree. The alignment was performed with MAFFT using the `--addfragments` and `--keeplength` arguments, which is recommended for adding relatively short sequences to an existing alignment, and ensures that no gaps are added to the original alignment [CSDB22][KF12]. An important aspect of the `--addfragments` option is that each reference sequence gets independently added to the alignment, which ensures that the results of phylogenetic placement do not depend on the other reference sequences placed in the same subtree. MAFFT was run with the following command:

```
$ mafft-linux64/mafft.bat --addfragments $input_query_file --keeplength
 ↪  $chunk_dir/$base_name.fasta > $query_align_dir/$base_name.fasta
```

Besides the extended alignment, `pplacer` also requires a reference package as input, which includes the FASTA file containing the original alignment of the subtree sequences, the subtree file and an info file containing relevant parameters used during the construction of the subtree. Unfortunately, this metadata regarding the tree construction has not been made available for the subtrees generated from the chunks. The decision was therefore made to regenerate the subtrees for each chunk the reference sequences had to be placed in, using the same method used by *Luuk Romeijn and Casper Carton (2022)*[CR22]. The subtrees were regenerated using RAxML, using the GTRCAT model, with the following command:

```
$ standard-RAxML-8.2.13/raxmlHPC-PTHREADS-SSE3 -s $chunk_dir/$base_name.fasta -n
 ↪  $base_name_num.out -w $regen_trees_dir_abs -m GTRCAT -p 12345 -T 4 -o
 ↪  OUTGROUP
```

The `RAxML_bestTree` and `RAxML_info` files outputted were then used, together with the original alignment file, to create the reference package for each subtree. This was done with taxtasic using the following command:

```
$ taxit create -l its -P $refpkg_dir/$base_name.refpkg --aln-fasta
 ↪  $chunk_dir/$base_name.fasta --tree-stats
 ↪  $tree_dir/RAxML_info.$base_name_num.out --tree-file
 ↪  $tree_dir/RAxML_bestTree.$base_name_num.out
```

Finally, `pplacer` is run for each subtree which has reference sequences to be placed using the following command:

```
$ pplacer-Linux-v1.1.alpha19/pplacer -c $refpkg_dir/$base_name.refpkg
 ↪  $query_align_dir/$base_name.fasta -o $pplacer_out_dir/$base_name.jplace
```

The placement results for the full length reference sequences were then compared to their truncated counterparts to analyze the difference of their placements in the subtree, with the goal of quantifying the effect of truncation on the reliability of phylogenetic placement on the reference tree.

3.2 MDDB integration

After analyzing the effect of truncating the ZOTUs, will the data from the ARISE study be integrated in MDDB. The integration consist of two parts, each described separately: the integration of the ARISE data in the database tables 3.2.1, followed by implementing tools in the database server to allow end users to perform server-side phygenetic placement directly on MDDB query results 3.2.2.

3.2.1 ARISE data integration

First, the final truncation value for the ZOTUs will be decided, using the conclusions from the method described in section 3.1. We then update the relevant database tables shown in figure 3 with the ARISE sample information and metadata. The following database tables are updated for category in the UML diagram:

Sequence The ReferenceSequence table and contains relationship table are updated using the `update_ref_map.py` script from the PROFUNGIS post processing pipeline. The current contents of these tables are first exported from MDDB using the following SQL queries:

```
sql>COPY SELECT * FROM "RefSequence" INTO 'refseq_table_pk.csv' ON CLIENT
↪  USING DELIMITERS ',' , E'\n', '';
sql>COPY SELECT * FROM "Contain" INTO 'mapping_table_pk_zotu_srr.csv' ON
↪  CLIENT USING DELIMITERS ',' , E'\n', '';
```

The exported csv files are then updated with the ARISE ZOTUs for all samples before being inserted back into MDDB. We will also report on how many reference sequences found among the ARISE sample were not yet present in MDDB.

Study For the Study section of MDDB, we update only the Sample and Study tables. The Sample table will include a new entry for each of the 190 samples. These entries will use the NBCLAB number of place of the SRA sample number, as we are not dealing with sample from SRA. The location tables figure 3 have not yet been implemented yet, so the sample coordinates of the two locations are added as columns to the entries in the Sample table.

The Study table receives a single new entry for the ARISE metabarcoding study, including the relevant study information received from Naturalis.

Taxonomy For the taxonomy section of MDDB, we update both the AssignTaxa and ReferenceTaxonomicDB tables using the taxonomic assignments derived from the taxonomic assignment method described in section 3.1.5. Scripts for updating these tables were made available by the original MDDB contributors.

3.2.2 Server-side phylogenetic diversity analysis

MDDB currently has limited methods of user interaction. The current search tool[11], created by *Haike van Thiel (2022)*[vT22], allows users to filter for reference sequences using geographical and

[11] https://mycodiversity.liacs.nl/search-tools/biodiversity-and-distribution-search

taxonomic filters. This tool allows for large scale analysis of species diversity across the multiple studies included in MDDB, but it does not allow for analysis of phylogenetic diversity. A new tool was therefore implemented on the MDDB web server that allows for phylogenetic placement of the reference sequences using the method proposed by *Lena ten Haaft (2023)* [tH23].

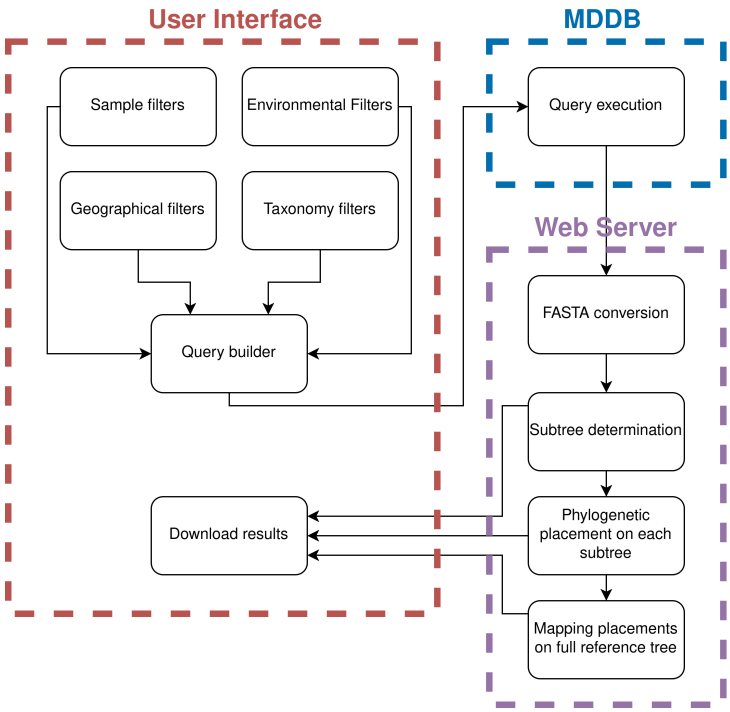

Figure 12: Design of the new tool for phylogenetic analysis on the MDDB web server

Figure 12 shows the design layout for the new phylogenetic placement tool. The user is presented with a webpage with different filters used to filter the total reference sequence set in MDDB. The filters are split in four categories and allow the user to filter the following database columns:

Sample Filters

- BioProject
- BioSample
- SRA Study
- SRA Sample

Geographical Filters

- Continent
- Subregion
- Country

Environmental Filters

- Environment Feature
- Biome Term
- Material

Taxonomy Filters

- Phylum
- Class
- Order
- Family
- Genus
- Species

The selected filters are then appended as WHERE clauses to the base query, used to get the reference sequence subset the base query is structured as follows:

```
SELECT RS.refsequence_pk, RT.sh_unite_id, RT.phylum_name, RT.species_name,
 ↪  COUNT(DISTINCT SP.biosample_id) AS count_biosample_id, COUNT(DISTINCT
 ↪  SP.sra_sample) AS count_sra_sample
FROM "Sample" SP
JOIN "Contain" CN ON SP.sample_pk = CN.sample_pk
JOIN "RefSequence" RS ON CN.refsequence_pk = RS.refsequence_pk
JOIN "Include" IC ON SP.sample_pk = IC.sample_pk
JOIN "Study" ST ON IC.study_pk = ST.study_pk
JOIN "AssignTaxa" AX ON RS.refsequence_pk = AX.refsequence_pk
JOIN "RefTaxonomicDB" RT ON AX.refsequence_taxonomic_pk =
 ↪  RT.refsequence_taxonomic_pk
GROUP BY RS.refsequence_pk, RT.sh_unite_id, RT.phylum_name, RT.species_name
ORDER BY RS.refsequence_pk
```

When the user executes the query, a request is made to the MDDB server and the resulting reference sequences are presented in a table for the user to explore. When the user then wants to perform phylogenetic placement on the sequences, a request is made to the web server which will convert the query results to a FASTA file used as input for the phylogenetic placement method described in section 3.1.6. The results of the majority chunk determination, together with the jplace files for each subtree are bundled in a archive file for the user to download. The archive also incudes a jplace file for the entire reference tree, which combines the placements for all reference sequences across all subtrees. This is useful as it allows users to use measures of phylogenetic diversity, such as Faith's PD [Fai92], on all placed reference sequences over the full phylogeny.

The tool was written in PHP, using The Microsoft Open Database Connectivity (ODBC) interface to connect with MDDB. The scripts for phylogenetic placement were all written as bash scripts.

4 Experiments & Results

The results of this research are once again split up in the integration analysis of the ARISE data in MDDB and the actual integration of the data and implementation of server-side phylogenetic diversity analysis in MDDB. In this section, the results of the implementations described in section 3 will be analyzed.

4.1 ARISE intergration analysis

Every step of the method for analyzing the effect of truncation on the ZOTUs in figure 10 has been performed, resulting in multiple csv files containing the results of individual steps of the method. A Jupyter Notebook was created for analyzing this data, where the `pandas` Python package was used to merge the intermediate results tables to allow for detailed comparisons between the full-length and truncated ZOTU sets.

4.1.1 Filter results

Figure 13 shows an overview of the number of ZOTUs constructed from the ARISE data by PROFUNGIS, together with the number of ZOTUs that pass the abundance filter, the contamination filter and the number of unique reference sequences derived over all samples. This is shown for both the full-length and truncated ZOTU sets, where the abundances of discarded and new ZOTUs are shown along every step in the pipeline. The full results tables for the truncation and filter analysis can be found in appendix A.

The results show that over the 190 samples, 65595 full-length ZOTUs have been constructed, averaging to 345 ZOTUs per sample. Of these ZOTUs, only 2% was discarded during the truncation process, indicating that the vast majority of the ZOTUs have a length greater than 250bp. The truncated ZOTU sets decreases in size by only 4.5% after dereplicaring, indicating that the vast majority of the ZOTUs are unique in the first 250 bp. The abundance filter reduces the ZOTU count to 9-10% of the full ZOTU set for both the full-length and truncated ZOTUs, indicating that a distribution of ZOTUs abundances is significantly skewed to the left, as the cutoff percentage is only 0.5%. The contamination filter reduces the ZOTU count only by 1-2% of the ZOTUs that pass the abundance filter. This is a strong indication that the abundant ZOTUs are highly likely to have a BLAST hit against UNITE.

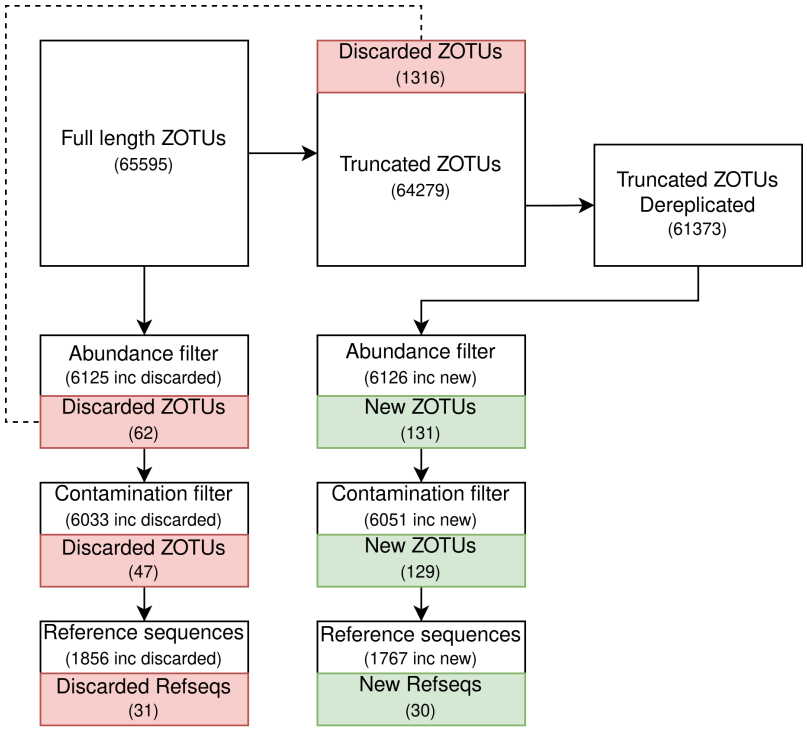

Figure 13: Overview of the truncation results combined with the results of the abundance and contamination filters before and after truncation. The filter results of the discarded and new ZOTUs are tracked separately.

Table 12 gives a comparison of the filter results before and after truncation. A total of 564 ZOTUs only pass the abundance filter after truncation as a result from the scenarios described in figure 11,

of which 546 also pass the contamination filter. These ZOTUs dereplicate to 381 unique truncated ZOTUs. Of these, 129 are new ZOTUs. This is only 2% of all truncated ZOTUs. Table 12 also shows that no ZOTUs no longer pass the contamination filter after truncation when they would have passed at full length.

4.1.2 Reference sequences

Table 1 shows the distribution of the reference sequences derived from the full-length ZOTUs over both sample locations. A total of 1856 unique reference sequences are derived over all samples. 31 of these are the result of the discarded ZOTUs, only 1.7% of all reference sequences. Of the remaining 1825 reference sequences, 52 are found in both locations at full-length. 15 reference sequences that are exclusively found in one location map to truncated reference sequences that are found in both locations, resulting in an increase from 2.9% to 3.7% of sequences found in both locations, which is relatively small. From this we can conclude that truncation only has a small impact on the specificity of sequence diversity found in both locations.

Table 1: Location exclusivity for full-length reference sequences after truncation

Of the 1856 full-length reference sequences:			
passed trunc	refseq full exclusive to loc	refseq trunc exclusive to loc	count
False	True	False	31
True	False	False	52
	True	True	1758
		False	15

Table 2 shows the distribution of truncated reference sequences over both locations, of which there are a total of 1767. We see that the 381 unique truncated ZOTUs that only pass the abundance and contamination filters after truncation map to 233 unique reference sequences. 30 of these are derived from the 129 new ZOTUs and are only found in a single location. 2 others are now found in both locations because of the addition of the new ZOTUs. From this we can conclude that the addition of the new ZOTUs as the result of truncation has a negligible effect on the specificity of the sequence diversity found in both locations.

Table 2: Location exclusivity for truncated reference sequences

Of the 1767 truncated reference sequences:			
passed contam full	refseq trunc exclusive to loc	refseq trunc new to loc	count
False	True	False	185
		True	30
	False	False	16
		True	2
True	True	False	1683
	False	False	54

4.1.3 Taxonomic assignment

The full tables showing the results of the taxonomic assignment analysis can be found in appendix B. First, we look at the proportion of reference sequence that do not have a taxonomy hit. Afterwards, we look at the effect the discarded and new reference sequences have on the observed biodiversity in both locations. Finally, we study the difference truncation has on assigned taxonomies and the effect these changes have on the observed biodiversity in both locations.

Missing taxonomic assignment Table 14 shows the distribution of full-length reference sequences that do and do not get assigned a UNITE SH using the method described in 3.1.5, comparing them to their truncated counterparts. A total of 254 full-length sequences are missing a taxonomy hit, 13.7% percent of all full length reference sequences.

The fact that reference sequences can lack a taxonomy hit while having passed the contamination filter can be explained by the difference between local and global alignment algorithms. The taxonomic assignment method looks for an end-to-end match, which is more stringent than the contamination filter, which also accepts partial matches. Figure 14 shows the distribution of sequence length of all full-length reference sequences with and without a taxonomy hit. We see that the sequences missing a taxonomy are mostly situated among the longer sequences. This is to be expected as longer sequences are less likely to have a global alignment hit.

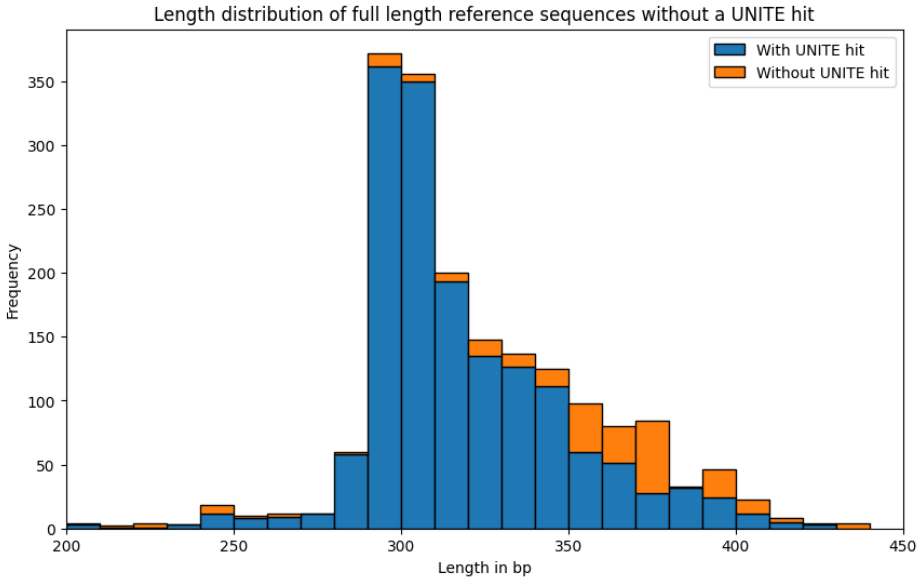

Figure 14: Length distribution of full-length reference sequences with and without a taxonomy hit.

We also observe from table 14 that 11 full-length reference sequences only have a taxonomy hit in their truncated form, which can be explained by the fact that shorter sequences are easier to match end-to-end. The opposite can also be true, as 1 sequence loses their taxonomy hit after truncation. As these sequences add up to <1% of the total full-length reference sequences, we can conclude that truncation has a negligible effect on whether a taxonomy hit is found.

Finally, table 14 shows that for the 31 reference sequences that are discarded during truncation, 11 do not have a taxonomy hit (35.5%). Using a confidence interval for a single proportion, we calculate the 95% CI for this observation to be 19.24% to 54.65%. As the proportion of all full-length reference sequences missing a taxonomy hit is below this range, we can conclude with 95% certainty that the higher missing taxonomy rate of the discarded reference sequences is not the result of random sampling.

Table 15 shows the distribution of missing taxonomies for the 233 truncated reference sequences that are mapped to by the 381 truncated ZOTUs that only pass the abundance filter after truncation. 57 of these do not have a taxonomy hit (24.4%). Of the 32 new reference sequences, 7 do not have a taxonomy hit (21.9%). These proportions are again higher than than the proportion of all full-length reference sequences missing a taxonomy, indicating that the reference sequences mapped to by the less abundant ZOTUs are more likely to miss a taxonomy hit.

Taxonomy of discarded reference sequences Table 16 shows for the 20 discarded reference sequences whether their mapped to UNITE ids are also present in the remaining full length and truncated reference sequences in the same location, paired with the lowest taxonomic rank that is also found there. We see that while none of the UNITE ids mapped to the discarded reference sequences are found elsewhere, most of the assignments are still present up to species level elsewhere. This indicates that the loss of these reference sequences after truncation has a very small effect of the observed biodiversity.

Taxonomy of new reference sequences Table 17 shows the same overview of the uniqueness of taxonomic assignment for the 25 new reference sequences after truncation. We again see that a majority of assigned taxonomies are present up to species level in the remaining truncated references in the same location, indicating that the introduction of the new reference sequences does not greatly effect the observed biodiversity. In contrary to the discarded reference sequences, we even see that most get mapped to UNITE ids that are also found elsewhere in the same location.

Effect of truncation on assigned taxonomy Table 18 shows for the 1631 full-length reference sequences that have taxonomy hits before and after truncation, whether their mapped to UNITE ids at full length are still present in their locations after truncation, together with the lowest taxonomy rank still present in their locations. This is a less strict measure compared to comparing the taxonomic assignment of each individual sequence after truncation, as a change in taxonomic assignment for a sequence is less impactful to the observed biodiversity if the original taxonomy is still found elsewhere in the same sample, or another sample in the same location. We see that the truncation of 1384 reference sequences (84.9%) does not result in the loss of their full-length assigned UNITE-ids within their locations. This increases to 1447 (88.7%) when we include the sequences that do result in a loss of their UNITE-ids in their locations, but keep their taxonomies up to species rank present in their locations. Including the sequences that only result in a loss of found biodiversity at species rank increases this further to 1585 (97.2%). From this we can conclude that 88.7% of the observed biodiversity within locations remains equal op to species rank and 97.2% remains equal up to genus rank, using this method for taxonomic assignment.

Table 19 shows the pair-wise comparison of the taxonomic assignment for all 1581 unique full-length reference sequences that have taxonomy hits before and after truncation. Note that the number of sequences is less compared to table 18, as sequences that appear in both locations are only listed once. We see that 1285 sequences (81.3%) map to the exact same UNITE id after truncation, increasing to 1321 (83.6%) when including the sequences that get mapped to a different UNITE id but keep identical taxonomy up to species level. Including the sequences that have equal taxonomic assignment up to genus level increases this further to 1459 (92.3%). These percentages are lower than what we derived from table 18, which is to be expected as we are no longer comparing to all taxa found in the sequences respective locations. The drop in the percentages is small however, from which we can conclude that only around 5% more of the full-length assignments at species and genus level get preserved when looking at the taxa found per location. Table 19 also provides the distribution of how the taxonomic assignment changes below the lowest equal rank, indication wether the switches were from or to identified or unidentified taxa. We see a clear trend where changes in lower taxonomic ranks result in relatively more switches from defined to other defined taxa, while changes in higher ranks more often switch to and from unidentified taxa. This can be explained by the fact that the differences in the ITS2 marker get more subtle the lower in taxonomic rank you are comparing them.

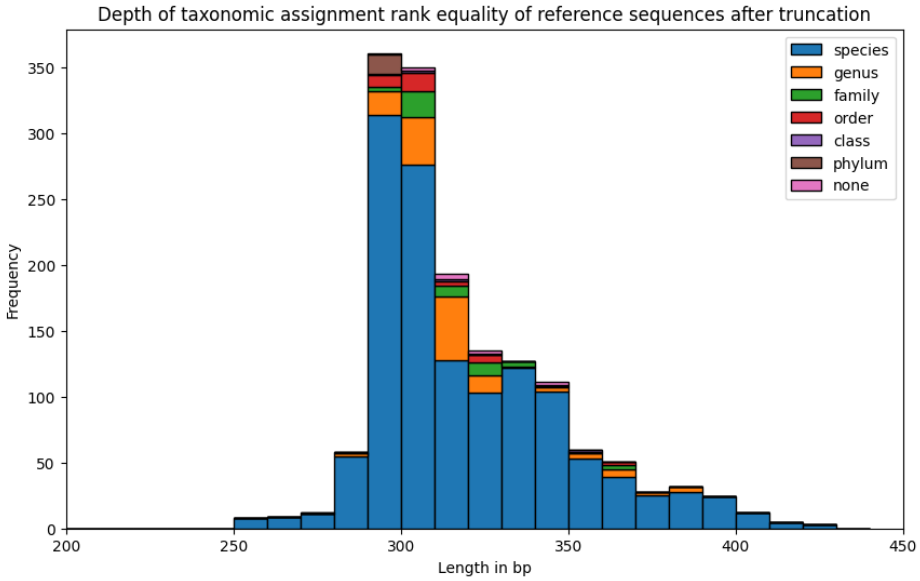

Figure 15: Length distribution of the depth of taxonomic assignment rank equality of reference sequences after truncation

Figure 15 shows how the lowest taxonomic assignment equalities shown in table 19 are distributed by length of the full-length reference sequences. Longer reference sequences lose more information during truncation, which can impact the global alignment results. One would therefore expect longer reference sequences to have more severe differences in taxonomic assignment after truncation compared to sequences closer in length to the truncation length. This is however not what we observe. The changes in taxonomic assignments seem to be fairly evenly distributed over all sequence lengths. A possible explanation for this is that certain taxa could be more sensitive to changes in sequence length compared to others.

Positive control results The positive control sample is marked as NBCLAB4311 in the sample list. Table 3 shows the results of taxonomic assignment for the ZOTUs found in this sample. We see that from the 7 ZOTUs found, only 1 has a taxonomy hit using USEARCH. The mapped UNITE id is the same before and truncation, and is identified as a member of the Lactarius genus. These results are in line with our expectations, and show that truncation did not impact the accuracy of taxonomic assignment in this single instance where we could verify the ground truth.

Table 3: Result of taxonomic assignment on the ZOTUs found in the positive control sample NBCLAB4311

zotu id	identity full	UNITE id full	identity trunc	UNITE id trunc	species
Zotu1	99.7	SH0854870.10FU	100	SH0854870.10FU	s__Lactarius_aurantiolamellatus
Zotu2	NaN	NaN	NaN	NaN	NaN
Zotu6	NaN	NaN	NaN	NaN	NaN
Zotu3	NaN	NaN	NaN	NaN	NaN
Zotu4	NaN	NaN	NaN	NaN	NaN
Zotu5	NaN	NaN	NaN	NaN	NaN
Zotu7	NaN	NaN	NaN	NaN	NaN

4.1.4 Phylogenetic placement

First, we look at the results of the majority chunk determination using BLAST for both the full-length and truncated reference sequences, comparing them to the previously assigned taxonomies. Then, we look at the results of phylogenetic placement within the chunk subtrees, using a weighted distance measure to quantify the placement divergence between the full-length and truncated reference sequence pairs.

Majority chunk determination The full tables showing the results of the majority chunk determination can be found in appendix C. Tables 20 and 21 show for the full-length and truncated reference sequences respectively if a chunk could be assigned and wether the taxonomy at the splitting rank of the chunk equals the taxonomy assigned by USEARCH. We observe in both tables that more than half of the sequences get assigned to the chunk which is in agreement with USEARCH. These sequences notably have on average only 1 chunk in the BLAST results, with a majority chunk count of 10. This is a strong indication that these sequences can be assigned to these chunks with high confidence. Around 20% of the sequences do not get assigned to the chunk in agreement with USEARCH. We see that these sequences have on average 3 chunks among the BLAST results with a average majority count of 7, indicating more difficulty in finding the correct chunk for these sequences to be placed. Similar results are observed for the sequences which do not get assigned a taxonomy by USEARCH, but do get assigned a chunk. The final group worth taking a closer look at are the 8% of sequences that do get assigned a taxonomy using USEARCH, but do not get assigned a chunk due to the lack of BLAST hits. Tables 4 and 5 show that the vast majority of these sequences get assigned to the Inocybaceae family. This observation is likely the result of the difference in the UNITE releases used for the taxonomic assignment (version 10.0) and for constructing the reference tree (version 8.3). The Inocybaceae family is is one of the larger families among the Agaricales order [RLJ10], so it is plausible that new species have been added in the 3 years between these releases, which can not be placed in the reference tree build using the older data. Table 4 also shows an overrepresentation of reference sequences shorter than 250bp, which get discarded after truncation. This is a clear indication that the discarded reference sequences can less reliably be placed on the reference tree, strengthening our argument for discarding them.

Table 4: Taxonomy of the full-length reference sequences that do have a taxonomy hit but are not assigned a chunk

Of the 143 full-length reference sequences that do have a taxonomy hit but are not assigned a chunk:			
order	family	length <250bp	count
o__Agaricales	f__Inocybaceae	False	123
o__Rozellomycota_ord_Incertae_sedis	f__Rozellomycota_fam_Incertae_sedis	True	7
		False	5
o__Saccharomycetales	f__Dipodascaceae	True	3
o__Fungi_ord_Incertae_sedis	f__Fungi_fam_Incertae_sedis	False	2
		True	2
o__Sporidesmiales	f__Sporidesmiaceae	False	1

Table 5: Taxonomy of the truncated reference sequences that do have a taxonomy hit but are not assigned a chunk

Of the 125 full-length reference sequences that do have a taxonomy hit but are not assigned a chunk:		
order	family	count
o__Agaricales	f__Inocybaceae	119
o__Rozellomycota_ord_Incertae_sedis	f__Rozellomycota_fam_Incertae_sedis	5
o__Sporidesmiales	f__Sporidesmiaceae	1

Table 22 shows for all reference sequences that have taxonomy hits before and after truncation whether they get assigned to the same chunk, and if the taxonomy at the splitting rank of the chunk is equal to the taxonomic assignment for either the full-length or truncated sequence. We observe an overwhelming similarity between the results before and after truncation, with no sequences gaining or losing chunk assignment after truncation, and 99,8% of sequences getting assigned to the same chunk after truncation. We do however again observe that sequences that get assigned a chunk which is in agreement with the USEARCH assignment have a lower number of chunks in the BLAST results, both at full-length and truncated.

These observations show that the majority chunk determination method is very robust against the truncation of the reference sequences, more so than the taxonomic assignment method using USEARCH. Table 6 shows this for the sequences that differ in taxonomic assignment at the order rank after truncation, showing that they get assigned to the same chunk after truncation. There

are two factors that provide a possible explanation for this robustness. First of all, it is important to remember that all UNITE SH sequences that have undefined taxonomy up to the splitting rank of the chunks are not included in the reference tree, meaning that the majority chunk is decided from a smaller subset of the UNITE SH sequences. This decrease in options makes it more likely for the full-length and truncated reference sequences to have similar results. The second difference between the methods is the fact that the chunk assignment method uses a majority vote, while the taxonomic assignment method takes the top USEARCH hit. Looking only at the top hit understandably introduces more variance in the results after truncation, and the shown robustness that the majority vote method provides calls in to question the accuracy of the current taxonomic assignment method.

Table 6: Majority chunk assignment at the order splitting rank are equal after truncation, even when the taxonomic assignment using USEARCH differs

order_full	order_trunc	maj_split_tax
o__Agaricomycetes_ord_Incertae_sedis	o__Trechisporales	o__Trechisporales
o__Agaricomycetes_ord_Incertae_sedis	o__Trechisporales	o__Trechisporales
o__Sordariales	o__Sordariomycetes_ord_Incertae_sedis	o__Sordariales
o__Sordariomycetes_ord_Incertae_sedis	o__Hypocreales	o__Hypocreales
o__Sordariomycetes_ord_Incertae_sedis	o__Microascales	o__Microascales
o__Leotiales	o__Thelebolales	o__Thelebolales
o__Leotiales	o__Helotiales	o__Onygenales
o__Hypocreales	o__Diaporthales	o__Diaporthales

Placement divergence in subtrees We quantify the difference of placements in the subtrees before and after truncation by calculating the distance divergence in the subtrees between each full-length and truncated reference sequence pair, weighed by the likelihood weight ratio of each placement and normalized by the longest branch of each subtree. The formula for calculating weighted distance divergence D_w for a sequence pair is described by equation 1.

$$D_w = \sum_{i=1}^{N} \sum_{j=1}^{N} P_{\text{full}}(n_i) \cdot P_{\text{trunc}}(n_j) \cdot \frac{d(n_i, n_j)}{L} \tag{1}$$

In this formula, N is the total number of nodes in the union of the two placement sets. $P_{\text{full}}(n_i)$ and $P_{\text{trunc}}(n_j)$ denote the likelihood weight ratio of the full-length and truncated placements at nodes i and j. $d(n_i, n_j)$ denotes the phylogenetic distance between nodes n_i and n_j, and L denotes the branch length of the longest branch in the subtree. The resulting metric shows us how many times the longest branch length the placements differ before and after truncation.

Figure 16 shows the distribution of the weighted distance divergence scores for all 1146 reference sequence pairs which are assigned to the same chunk subtree before and after truncation. We observe an exponential distribution, with the vast majority of truncated sequences being placed in close proximity to the full-length placement. 90% of truncated placements are within 0.59 times the longest subtree branch and 95% of observations fall within 0.97 times the longest subtree branch.

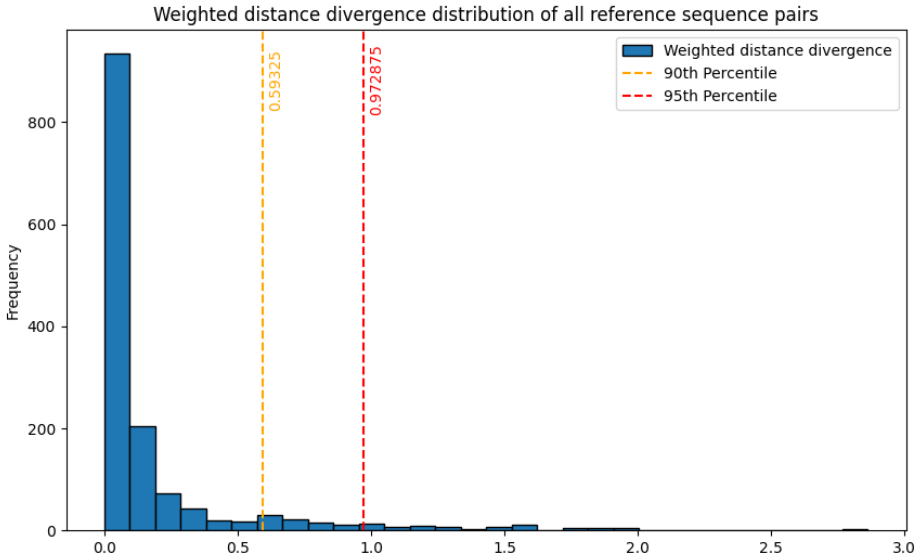

Figure 16: Distribution of weighted distance divergence of all 1146 reference sequence pairs assigned to the same subtree

Figure 17 shows an example of what the weighted distance divergence scores represent in practical terms. We see the difference in placements between a full-length and truncated reference sequence with a weighted distance divergence score of 0.9737. The full-length sequence gets placed on a single edge with a LWR score of 1, while the truncated sequences gets placed in a different subbranch with lower LWR scores. We can clearly see that truncating can negatively influenced the accuracy of the placements for this sequence. The practical effect of these difference however depends on how these placements are used for further research. While the distance divergence score is relatively high, the distance to the root of the tree stays relatively similar, as the subtree is much larger than the fragment shown here. This difference becomes even more negligible when calculating the distance to the root of the complete backbone tree. We have seen that 95% of sequence pairs score better than this example. In the end, it is up to biologists to decide how acceptable these placement differences are.

Figure 17: Comparison of the phylogenetic placement of a sequence pair in the subtree for chunk 50 (Boletales, size: 258 sequences), scoring a weighted distance divergence score of 0.9737. The size of the circles show the relative LWR scores of the placements. The left image shows the full-length placements, the right shows the truncated placements.

Figure 18 shows the distribution of the placed sequence pairs over the chunk subtrees, also showing the size of the subtrees. The weighted distance divergence scores are colored to show the sequence pairs in each subtree that score below, in between, and above the 90th and 95th percentile scores. A lookup table for translating the chunk numbers to their splitting taxonomy can be found in appendix D. We see in general that larger subtrees have more sequences in them. You would expect larger subtrees to have a higher rate of sequence pairs with a high weighted distance divergence, as larger trees allow for truncated sequences to be placed further away compared to smaller trees. We do indeed observe most of the placements with a weighted distance divergence above the 95th percentile occur at large subtrees. There are however some notable outliers among the smaller subtrees, as subtrees 202, 211, and especially 210 perform very poorly across most of their placements. These chunk numbers map to Acarosporales, Baeomycetales and Lecideales orders respectively. All of these are members of the Lecanoromycetes class. Manually looking at the placement for these subtrees show low LWR scores for both the full-length and truncated reference sequences, indicating that even before truncation could the sequences not be placed with high confidence.

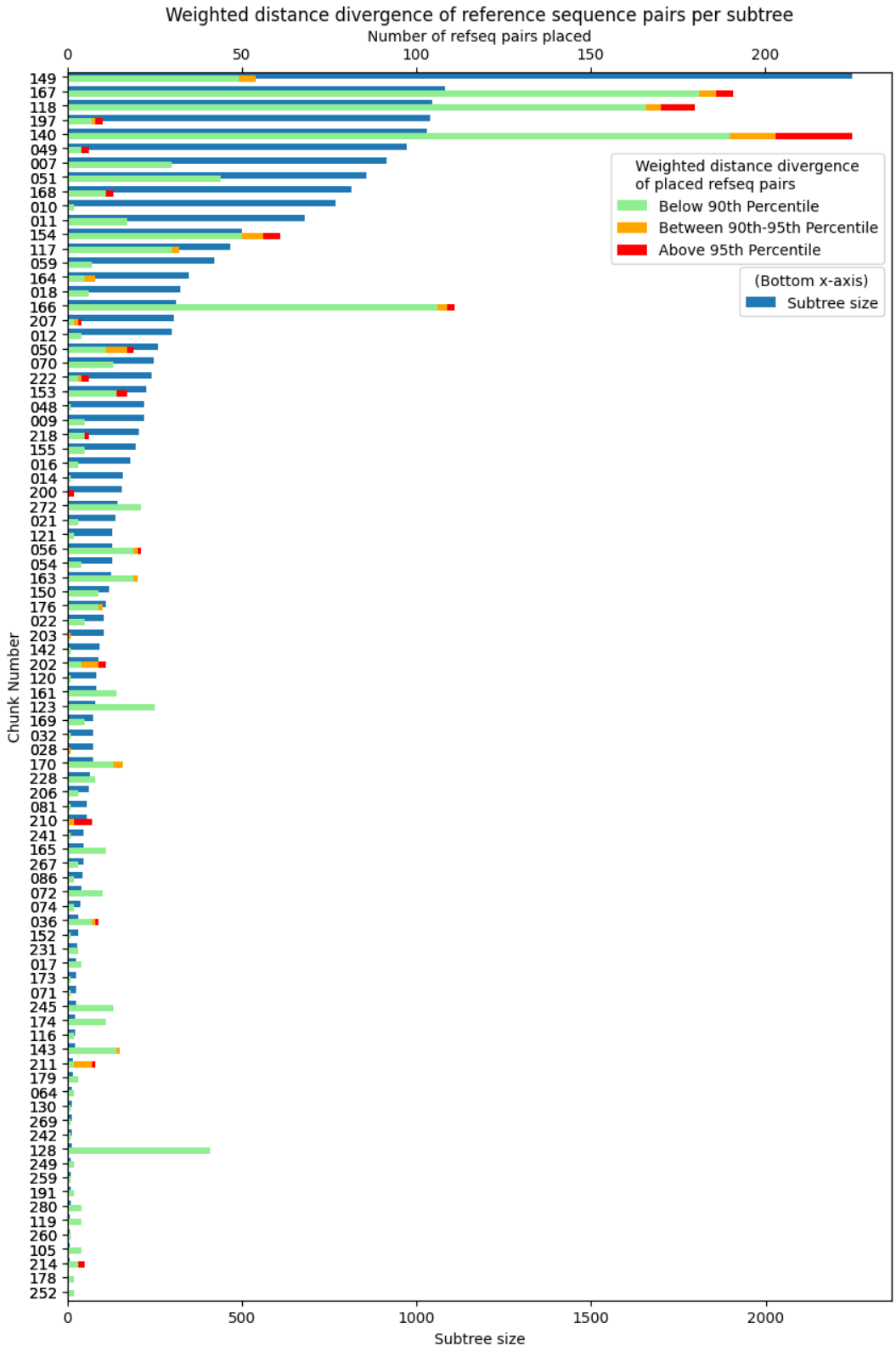

Figure 18: Distribution of the placed sequence pairs over all subtrees, compared to the size of the subtrees. The color coding shows the distribution of high distance divergence scores over all subtrees.

4.2 MDDB integration

We first look at the results of integrating the ARISE reference sequences in MDDB, followed by the implemented web application for performing phylogenetic placement analysis on the MDDB server. De

4.2.1 ARISE data integration

After analysing the results from section 4.1, the decision was made to integrate the ARISE ZOTUs truncated to 250bp. The observed loss in sequence specificity per location and changes in observed biodiversity were not large enough to justify changing the current truncation length for this data. The ZOTU list for all 190 samples were processed using the `update_ref_map.py` script, adding them to the existing ReferenceSequence table and contains relationship tables.

Table 7 shows the increase of the size of the ReferenceSequence table after adding the ARISE ZOTUs for all 190 samples. A total of 1210 sequences have been added, meaning that these had not yet been found in the samples previously present in MDDD. The total unique number of reference sequences found in the ARISE samples was determined to be 1767, which implies that 68.5% of reference sequences found were not yet present in MDDB.

Description	Value
Number of sequences before ARISE integration	172,463
Number of sequences after the last sample	173,673
Total increase in sequences from all samples	1,210

Table 7: Summary of the number of sequences in the ReferenceSequence table before and after the ARISE data integration

The remaining tables mentioned in section 3.2.1 have been updated with the ARISE sample, study, and taxonomy data as described.

4.2.2 Server-side phylogenetic diversity analysis

The source code for the server-side phylogenetic placement tool can be found on GitHub [12]. The tool has not yet been integrated on the MycoDiversity web server, but is fully functional when run locally, connecting to the MDDB server over a SSH tunnel.

5 Conclusions & Further Research

During this research, the main goal was to integrate the ARISE ITS data in MDDB, and to provide the necessary tools for performing biodiversity analysis on these samples, comparing them to the samples already present in MDDB. To achieve this, we first analyzed the effect of the preprocessing steps performed on the raw sequence data by the PROFUNGIS pipeline. Specifically, we analyzed the effect truncating the sequences to 250bp has on the observed sequence diversity

[12] https://github.com/SethGG/mycodiversity-pplacer

for both sampling locations, as well the difference truncation makes on taxonomic assignment and phylogenetic placement of the sequences. Afterwards, a decision was made for the optimal truncation length of the ARISE sequences, and have implemented them in the MDDB database tables, together with the metadata regarding the samples and study. Finally, a online tool was introduced to allow for phylogenetic diversity analysis on all sequences in MDDB.

5.1 Research questions

Based on the obtained results, we can now answer the research questions stated in section 1.3.

> *RQ1.1: What effect does truncating the ARISE reference ZOTUs to 250bp have on biodiversity analysis?*

We have shown that truncating the sequences to 250bp does not meaningfully impact the sequence diversity in both sampling locations. We have also shown that 88.7% of unique taxonomies found within the sampling locations stay the same up to species level after truncation, increasing to 97.2% up to genus level. The discarded sequences smaller than 250bp have not been shown to significantly impact the species diversity results. The same was found for the new sequences introduced after truncation as the result of the abundance filter. Comparing the phylogenetic placement results of the truncated reference sequences showed limited placement divergence for most sequences, with 99.8% of sequences getting assigned to the same chunk after truncation.

> *RQ1.2: How can biodiversity analysis of the ARISE fungal ITS data be performed on the database server?*

Biodiversity analysis of the ARISE data can be performed on the database server for both species diversity and phylogenetic diversity measures. Species diversity analysis could already be performed using the existing search tool on the MDDB website[13]. For phylogenetic diversity analysis, a new web based tool was proposed that allows for phylogenetic placement for a subset of the sequences present in MDDB. The resulting placement files can then be visualized on the Interactive Tree Of Life (iTOL)[14], an online tool for the display, annotation and management of phylogenetic trees.

> *RQ1: How can the ARISE fungal ITS data best be integrated in the MycoDiversity Database for biodiversity analysis?*

The ARISE fungal ITS data can best be integrated in MDBB using a more manual approach compared to the automated workflow for sample data in SRA. The raw sequence data has to be processed by the PROFUNGIS pipeline, with the sample archive files extracted to the folder structure PROFUNGIS uses when downloading sequence run files from SRA. The current version of PROFUNGIS does not truncate sequences from Illumina platforms, leaving them at variable lengths after ZOTU construction. The truncation value of 250bp is not strictly necessary for this data, as its sequencing quality is higher than the samples currently included in MDDB, which used Roche 454 sequencing. We can however safely conclude that the ARISE sequences can be truncated

[13] https://mycodiversity.liacs.nl/search-tools/biodiversity-and-distribution-search
[14] https://itol.embl.de/

to 250bp, in line with all other MDDB sequences. This is ideal for MDDB as all sequences being the same length allows for sequence diversity comparisons among all samples included in MDDB.

The Sequence and contains relationship table can then be updated by exporting their current contents and integrating the ZOTUs found in the ARISE samples using the PROFUNGIS post processing scripts. The Sample table then has to be updated by adding an entry for every ARISE sample, using the NBCLAB numbers in place of the SRA sample number, together with the location coordinates. Because the sample tables in MDDB are designed for the structure of sample data in SRA, not all database tables can be expanded with data regarding the ARISE samples. The entire Literature section has to remain empty for now, as the ARISE project has not yet been formally published in a scientific article.

5.2 Discussion

In this section, several point of discussion are presented which were encountered during this research.

When discussing the progress of integrating the ARISE data in MDDB, Irene Martorelli, one of the main contributor for MDDB, mentioned that the contamination filter step in PROFUNGIS was not used for the sequences from the SRP043706 study. The reason for this was that an unusual number of ZOTUs did not pass the filter at the time, resulting in a meaningful loss of sequence diversity. The contamination filter was therefore adjusted to be less strict, but only after the data was added to MDDB, without the use of the contamination filter. This problem was not encountered during this research, as the contamination filter only reduced the ZOTU count by 1-2%. An argument can however be made for also skipping the contamination filter step for this data. We have observed during the phylogenetic placement analysis that different versions of the UNITE database can lead to different results using alignment algorithms, with some sequences not getting a match using older versions. As the contamination filter uses a BLAST search against UNITE, it is therefore possible for legitimate ZOTUs to be filtered out due to the UNITE version used. Integrating all ZOTUs that passed the abundance filter would eliminate this. The taxonomic assignment using USEARCH also indirectly filters out sequences which are not associated with known fungi, without discarding them, and can be redone on all reference sequences when a new UNITE version is released.

Another point of discussion regards the current taxonomic assignment method used for MDDB. We have seen that the method of using the top USEARCH hit is less robust to truncation compared to the majority vote used for chunk determination. This calls in into question the accuracy of the current method. Just using the top alignment hit can be regarded as an overly optimistic assignment, as a hit with a identity of 85% is not strictly a better taxonomic guess than a hit with a identity of 84%. Common methods for taxonomic assignment of ITS sequences use probabilistic models for assigning probabilities for each rank of the taxonomy. An example of such a method is Protax-fungi [ASN+18]. The downside of replacing the taxonomic assignment method with a probabilistic method is that reference sequences can than no longer be mapped to a single UNITE SH. This would therefore not work in the current database design.

The last point of discussion regards the accuracy of phylogenetic placement of sequences with taxonomies undefined before the splitting rank of the chunks in the reference tree. During construc-

tion of the reference tree, all UNITE SH sequences undefined before the splitting taxonomies are discarded, and therefore not included. As we perform phylogenetic placement only on the chunk subtrees, we therefore do not expect accurate placements of reference sequences in MDDB that have undefined taxonomy before the splitting rank. These sequences may however still get assigned to a chunk using majority voting. The accuracy of the chunk assignment and subsequent placement are debatable however, as we can not confirm the validity of the chunk assignment. Ideally, we would want to place these reference sequences on a subtree starting at the lowest defined taxonomic rank, but the current method for phylogenetic analysis implemented in this research does not allow for this dynamic placement behavior, as alignments are not available for the set of sequences in these proposed combined subtrees.

5.3 Further Research

In this section, we outline several topics related to this research that could be further expanded on.

First of all, more effort can be made to fully integrate the Literature section of MDDB with the relevant data regarding the ARISE project, when they are made available by Naturalis.

Secondly, the server-based phylogenetic placement tool has to be implemented on the MDDB server, and its performance tested. There are also still impactful ways of increasing the runtime efficiency of the web tool. Currently, each run performs all steps of the process from beginning to end on the requested subset of reference sequences. This can significant be sped up however, as the chunk determination and subtree placement of each reference sequence is deterministic in nature and does not depend on the other sequences in the query. To capitalize on this, we can create cache files on the server that store the results from the BLAST searches, majority chunk determination and placement within the subtrees, with the goal of preventing redundant computations. When the tool then has to place a sequence it has already placed before, it can simply copy the results from the cache, while filling the cache with the results of previously unseen reference sequences. The server-based phylogenetic placement tool could also be expanded to allow for visualization of the placement on the user interface of the tool, without the need of first downloading the placement results.

References

[AHNL+10] Kessy Abarenkov, R. Henrik Nilsson, Karl-Henrik Larsson, Ian J. Alexander, Ursula Eberhardt, Susanne Erland, Klaus Høiland, Rasmus Kjøller, Ellen Larsson, Taina Pennanen, Robin Sen, Andy F. S. Taylor, Leho Tedersoo, Björn M. Ursing, Trude Vrålstad, Kare Liimatainen, Ursula Peintner, and Urmas Kõljalg. The unite database for molecular identification of fungi – recent updates and future perspectives. *New Phytologist*, 186(2):281–285, 2010.

[AR02] Héctor T. Arita and Pilar Rodríguez. Geographic range, turnover rate and the scaling of species diversity. *Ecography*, 25(5):541–550, 2002.

[ASN+18] Kessy Abarenkov, Panu Somervuo, R. Henrik Nilsson, Paul M. Kirk, Tea Huotari, Nerea Abrego, and Otso Ovaskainen. Protax-fungi: a web-based tool for probabilistic taxonomic placement of fungal internal transcribed spacer sequences. *New Phytologist*, 220(2):517–525, 2018.

[Bau08] David Baum. Reading a phylogenetic tree: The meaning of monophyletic groups. 2008.

[BKN+13] R. Blaalid, S. Kumar, R. H. Nilsson, K. Abarenkov, P. M. Kirk, and H. Kauserud. Its1 versus its2 as dna metabarcodes for fungi. *Molecular Ecology Resources*, 13(2):218–224, 2013.

[CCA+09] Christiam Camacho, George Coulouris, Vahram Avagyan, Ning Ma, Jason S. Papadopoulos, Kevin Bealer, and Thomas L. Madden. Blast+: architecture and applications. *BMC Bioinformatics*, 10:421 – 421, 2009.

[CIKA10] M E T H O D O L O G Y A R T I C, Frederick Albert Matsen IV, Robin B. Kodner, and Virginia Armbrust. pplacer: linear time maximum-likelihood and bayesian phylogenetic placement of sequences onto a fixed reference tree. *BMC Bioinformatics*, 11:538 – 538, 2010.

[CR22] Casper Carton and Luuk Romeijn. Building a phylogeny for the fungal kingdom with its data. 2022.

[CSDB22] Lucas Czech, Alexandros Stamatakis, Micah Dunthorn, and Pierre Barbera. Metagenomic analysis using phylogenetic placement—a review of the first decade. *Frontiers in Bioinformatics*, 2, 2022.

[Edg10] Robert C. Edgar. Search and clustering orders of magnitude faster than blast. *Bioinformatics*, 26(19):2460–2461, 08 2010.

[Fai92] Daniel P. Faith. Conservation evaluation and phylogenetic diversity. *Biological Conservation*, 61(1):1–10, 1992.

[FHBJ18] Magdalena Frac, Silja E. Hannula, Marta Bełka, and Małgorzata Jedryczka. Fungal biodiversity and their role in soil health. *Frontiers in Microbiology*, 9, 2018.

[GLMVP+19] Mayra E. Gavito, Ricardo Leyva-Morales, Ernesto V. Vega-Peña, Héctor Arita, Teele Jairus, Martti Vasar, and Maarja Öpik. Local-scale spatial diversity patterns of ectomycorrhizal fungal communities in a subtropical pine-oak forest. *Fungal Ecology*, 42:100860, 2019.

[HL17] David L. Hawksworth and Robert Lücking. Fungal diversity revisited: 2.2 to 3.8 million species. *Microbiology Spectrum*, 5(4):10.1128/microbiolspec.funk–0052–2016, 2017.

[IGN+12] Stratos Idreos, F. Groffen, Niels Nes, Stefan Manegold, Sjoerd Mullender, and Martin Kersten. Monetdb: Two decades of research in column-oriented database architectures. *IEEE Data Eng. Bull.*, 35, 01 2012.

[KE08] W. John Kress and David L. Erickson. Dna barcodes: Genes, genomics, and bioinformatics. *Proceedings of the National Academy of Sciences*, 105(8):2761–2762, 2008.

[KF12] Kazutaka Katoh and Martin C. Frith. Adding unaligned sequences into an existing alignment using mafft and last. *Bioinformatics*, 28:3144 – 3146, 2012.

[KS13] Kazutaka Katoh and Daron M. Standley. Mafft multiple sequence alignment software version 7: Improvements in performance and usability. *Molecular Biology and Evolution*, 30(4):772–780, 01 2013.

[LBM15] David Laehnemann, Arndt Borkhardt, and Alice Carolyn McHardy. Denoising dna deep sequencing data—high-throughput sequencing errors and their correction. *Briefings in Bioinformatics*, 17(1):154–179, 05 2015.

[Mar24] I. Martorelli. private communication, Nov. 2024.

[MHGS12] Frederick A. Matsen, Noah G. Hoffman, Aaron Gallagher, and Alexandros Stamatakis. A format for phylogenetic placements. *PLOS ONE*, 7(2):1–4, 02 2012.

[MHK+20] Irene Martorelli, Leon S Helwerda, Jesse Kerkvliet, Sofia IF Gomes, Chivany RA van der Werff Jorinde Nuytinck, Guus J Ramackers, Alexander P Gultyaev, Vincent SFT Merckx, and Fons J Verbeek. Fungal metabarcoding data integration framework for the mycodiversity database (mddb). *Journal of integrative bioinformatics*, 17(1), 2020.

[PH20] Teresita M. Porter and Mehrdad Hajibabaei. Putting coi metabarcoding in context: The utility of exact sequence variants (esvs) in biodiversity analysis. *Frontiers in Ecology and Evolution*, 8, 2020.

[Ram18] Andrew Rambaut. How to read a phylogenetic tree. 2018.

[RHNM+14] Jai Ram Rideout, Yan He, Jose A. Navas-Molina, William A. Walters, Luke K. Ursell, Sean M. Gibbons, John Chase, Daniel McDonald, Antonio Gonzalez, Adam Robbins-Pianka, Jose C. Clemente, Jack A. Gilbert, Susan M. Huse, Hong Wei Zhou, Rob Knight, and J. Gregory Caporaso. Subsampled open-reference clustering creates consistent, comprehensive otu definitions and scales to billions of sequences. *PeerJ*, 2014(1), 2014.

[RLJ10] Martin Ryberg, Ellen Larsson, and Stig Jacobsson. An evolutionary perspective on morphological and ecological characters in the mushroom family inocybaceae (agaricomycotina, fungi). *Molecular Phylogenetics and Evolution*, 55(2):431–442, 2010.

[SCM+12] Diane S. Srivastava, Marc W. Cadotte, A. Andrew M. MacDonald, Robin G. Marushia, and Nicholas Mirotchnick. Phylogenetic diversity and the functioning of ecosystems. *Ecology Letters*, 15(7):637–648, 2012.

[SCS09] Martin Shumway, Guy Cochrane, and Hideaki Sugawara. Archiving next generation sequencing data. *Nucleic Acids Research*, 38(suppl_1):D870–D871, dec 2009.

[SG94] Erko Stackebrandt and Brett M. Goebel. Taxonomic note: A place for dna-dna reassociation and 16s rrna sequence analysis in the present species definition in bacteriology. *International Journal of Systematic and Evolutionary Microbiology*, 44:846–849, 1994.

[SSH+12] Conrad L. Schoch, Keith A. Seifert, Sabine Huhndorf, Vincent Robert, John L. Spouge, C. André Levesque, Wen Chen, Fungal Barcoding Consortium, Fungal Barcoding Consortium Author List, Elena Bolchacova, Kerstin Voigt, Pedro W. Crous, Andrew N. Miller, Michael J. Wingfield, M. Catherine Aime, Kwang-Deuk An, Feng-Yan Bai, Robert W. Barreto, Dominik Begerow, Marie-Josée Bergeron, Meredith Blackwell, Teun Boekhout, Mesfin Bogale, Nattawut Boonyuen, Ana R. Burgaz, Bart Buyck, Lei Cai, Qing Cai, G. Cardinali, Priscila Chaverri, Brian J. Coppins, Ana Crespo, Paloma Cubas, Craig Cummings, Ulrike Damm, Z. Wilhelm de Beer, G. Sybren de Hoog, Ruth Del-Prado, Bryn Dentinger, Javier Diéguez-Uribeondo, Pradeep K. Divakar, Brian Douglas, Margarita Dueñas, Tuan A. Duong, Ursula Eberhardt, Joan E. Edwards, Mostafa S. Elshahed, Katerina Fliegerova, Manohar Furtado, Miguel A. García, Zai-Wei Ge, Gareth W. Griffith, K. Griffiths, Johannes Z. Groenewald, Marizeth Groenewald, Martin Grube, Marieka Gryzenhout, Liang-Dong Guo, Ferry Hagen, Sarah Hambleton, Richard C. Hamelin, Karen Hansen, Paul Harrold, Gregory Heller, Cesar Herrera, Kazuyuki Hirayama, Yuuri Hirooka, Hsiao-Man Ho, Kerstin Hoffmann, Valérie Hofstetter, Filip Högnabba, Peter M. Hollingsworth, Seung-Beom Hong, Kentaro Hosaka, Jos Houbraken, Karen Hughes, Seppo Huhtinen, Kevin D. Hyde, Timothy James, Eric M. Johnson, Joan E. Johnson, Peter R. Johnston, E.B. Gareth Jones, Laura J. Kelly, Paul M. Kirk, Dániel G. Knapp, Urmas Kõljalg, Gábor M. Kovács, Cletus P. Kurtzman, Sara Landvik, Steven D. Leavitt, Audra S. Liggenstoffer, Kare Liimatainen, Lorenzo Lombard, J. Jennifer Luangsa-ard, H. Thorsten Lumbsch, Harinad Maganti, Sajeewa S. N. Maharachchikumbura, María P. Martin, Tom W. May, Alistair R. McTaggart, Andrew S. Methven, Wieland Meyer, Jean-Marc Moncalvo, Suchada Mongkolsamrit, László G. Nagy, R. Henrik Nilsson, Tuula Niskanen, Ildikó Nyilasi, Gen Okada, Izumi Okane, Ibai Olariaga, Jürgen Otte, Tamás Papp, Duckchul Park, Tamás Petkovits, Raquel Pino-Bodas, William Quaedvlieg, Huzefa A. Raja, Dirk Redecker, Tara L. Rintoul, Constantino Ruibal, Jullie M. Sarmiento-Ramírez, Imke Schmitt, Arthur Schüßler, Carol Shearer, Kozue Sotome, Franck O.P. Stefani, Soili Stenroos, Benjamin Stielow, Herbert Stockinger, Satinee Suetrong, Sung-Oui Suh, Gi-Ho Sung, Motofumi Suzuki, Kazuaki Tanaka, Leho Tedersoo, M. Teresa Telleria, Eric Tretter, Wendy A. Untereiner, Hector Urbina, Csaba Vágvölgyi, Agathe Vialle, Thuy Duong Vu, Grit Walther, Qi-Ming Wang, Yan Wang, Bevan S. Weir, Michael Weiß, Merlin M. White, Jianping Xu, Rebecca Yahr, Zhu L. Yang, Andrey Yurkov, Juan-Carlos Zamora, Ning Zhang, Wen-Ying Zhuang, and David Schindel. Nuclear ribosomal internal transcribed spacer (its) region as a universal dna barcode marker for ¡i¿fungi¡/i¿. *Proceedings of the National Academy of Sciences*, 109(16):6241–6246, 2012.

[Swe11] Nathan G. Swenson. The role of evolutionary processes in producing biodiversity patterns, and the interrelationships between taxonomic, functional and phylogenetic biodiversity. *American Journal of Botany*, 98(3):472–480, 2011.

[TBP+14] Leho Tedersoo, Mohammad Bahram, Sergei Põlme, Urmas Kõljalg, Nourou S. Yorou, Ravi Wijesundera, Luis Villarreal Ruiz, Aída M. Vasco-Palacios, Pham Quang Thu, Ave Suija, Matthew E. Smith, Cathy Sharp, Erki Saluveer, Alessandro Saitta, Miguel Rosas, Taavi Riit, David Ratkowsky, Karin Pritsch, Kadri Põldmaa, Meike Piepenbring, Cherdchai Phosri, Marko Peterson, Kaarin Parts, Kadri Pärtel, Eveli Otsing, Eduardo Nouhra, André L. Njouonkou, R. Henrik Nilsson, Luis N. Morgado, Jordan Mayor, Tom W. May, Luiza Majuakim, D. Jean Lodge, Su See Lee, Karl-Henrik Larsson, Petr Kohout, Kentaro Hosaka, Indrek Hiiesalu, Terry W. Henkel, Helery Harend, Liang dong Guo, Alina Greslebin, Gwen Grelet, Jozsef Geml, Genevieve Gates, William Dunstan, Chris Dunk, Rein Drenkhan, John Dearnaley, André De Kesel, Tan Dang, Xin Chen, Franz Buegger, Francis Q. Brearley, Gregory Bonito, Sten Anslan, Sandra Abell, and Kessy Abarenkov. Global diversity and geography of soil fungi. *Science*, 346(6213):1256688, 2014.

[tH23] Lena ten Haaf. Localized information comparison and analysis for mycodiversity database. 2023.

[THM+14] D. Lee Taylor, Teresa N. Hollingsworth, Jack W. McFarland, Niall J. Lennon, Chad Nusbaum, and Roger W. Ruess. A first comprehensive census of fungi in soil reveals both hyperdiversity and fine-scale niche partitioning. *Ecological Monographs*, 84(1):3–20, 2014.

[vT22] Haike van Thiel. Speeding up the multiscale access in the mycodiversity database. 2022.

[ZJ12] Nadine Ziemert and Paul R. Jensen. Chapter eight - phylogenetic approaches to natural product structure prediction. In David A. Hopwood, editor, *Natural Product Biosynthesis by Microorganisms and Plants, Part C*, volume 517 of *Methods in Enzymology*, pages 161–182. Academic Press, 2012.

[ZZZ+24] Yue Zou, Zixuan Zhang, Yujie Zeng, Hanyue Hu, Youjin Hao, Sheng Huang, and Bo Li. Common methods for phylogenetic tree construction and their implementation in r. *Bioengineering*, 11(5), 2024.

A Appendix: Filter & Truncation analysis tables

Table 8: Truncation results on full length ZOTUs

Of all 65595 full length ZOTUs found across all samples:			
passed trunc	count	proportion	count derep
True	64279	0.9799	61373
False	1316	0.0201	-

Table 9: Filter results on full length ZOTUs

Of all 65595 full length ZOTUs found across all samples:			
passed abun full	passed contam full	count	proportion
False	False	59470	0.9066
True	True	6033	0.0920
	False	92	0.0014
		6125	0.0934

Table 10: Filter results on discarded ZOTUs

Of the 1316 full length ZOTUs that are discarded (<250bp):			
passed abun full	passed contam full	count	proportion
False	False	1254	0.9529
True	True	47	0.0357
	False	15	0.0114
		62	0.0471

Table 11: Filter results on truncated ZOTUs

Of all 64279 truncated ZOTUs (61373 after dereplication within samples) found across all samples:					
passed abun trunc	passed contam trunc	count	proportion	count derep	proportion derep
False	False	57652	0.8969	55247	0.9002
True	True	6532	0.1016	6051	0.0986
	False	95	0.0015	75	0.0012
		6627	0.1031	6126	0.0998

Table 12: Comparing filter results before and after truncation

Of all 64279 truncated ZOTUs (61373 after dereplication within samples) found across all samples:					
passed abun full	passed abun trunc	passed contam full	passed contam trunc	count	proportion
False	False	False	False	57652	0.8969
	True	False	True	546	0.0085
			False	18	0.0003
				564	0.0088
True	True	True	True	5986	0.0931
		False	False	77	0.0012
				6063	0.0943

Table 13: Identification of new ZOTUs after truncation

Of the 564 truncated ZOTUs that passed the abundance filter only when truncated:			
new zotu trunc	passed contam trunc	count	count derep
False	True	317	252
	False	14	10
True	True	229	129
	False	4	2
		233	131

B Appendix: Taxonomic assignment analysis tables

Table 14: Comparing full-length reference sequences missing a taxonomy hit after truncation

Of the 1856 full-length reference sequences:					
passed trunc	tax missing full	tax missing trunc	refseq full exclusive to loc	count full	count trunc
False	False	NaN	True	20	NaN
	True	NaN	True	11	NaN
True	False	False	True	1531	1476
			False	50	49
		True	True	1	1
	True	False	True	11	10
		True	True	229	206
			False	3	3

Table 15: Taxonomy hits for truncated reference sequences mapped to by ZOTUs that pass the abundance filter only truncated

Of the 233 truncated reference sequences the ZOTUs that only pass the abundance filter after truncation map to:			
tax missing trunc	refseq trunc exclusive to loc	refseq trunc new to loc	count
False	True	False	136
		True	23
	False	False	15
		True	2
True	True	False	48
		True	7
	False	False	2

Table 16: Presence of the lowest taxonomy rank of discarded full-length reference sequences in the remaining full length and truncated reference sequences in the same sample location

Of the 20 discarded full-length reference sequences that have a taxonomy hit:					
UNITE id in refseq full	UNITE id in refseq trunc	tax in refseq full lowest	tax in refseq trunc lowest	count refseq	count UNITE id
False	False	species	species	15	10
		order	order	4	3
		class	species	1	1

Table 17: Presence of the lowest taxonomy rank of new truncated reference sequences in the full length and not new truncated reference sequences in the same sample location

Of the 25 new truncated reference sequences that have a taxonomy hit:					
UNITE id in refseq full	UNITE id in refseq trunc	tax in refseq full lowest	tax in refseq trunc lowest	count refseq	count UNITE id
True	True	species	species	13	12
False	False	species	species	5	5
		order	order	3	3
		family	family	2	2
		class	class	1	1
		species	class	1	1

Table 18: Presence of the lowest taxonomy rank of full-length reference sequences after truncation in the same sample location

Of the 1631 full-length reference sequences that have taxonomy hits before and after truncation:			
UNITE id in refseq trunc	tax in refseq trunc lowest	count refseq	count UNITE id
True	species	1384	826
False	genus	138	86
	species	63	50
	family	31	21
	order	13	9
	class	2	2

Table 19: Equality of taxonomic assignment for every unique full-length reference sequence after truncation

Of the 1581 unique full-length reference sequences that have taxonomy hits before and after truncation:				
UNITE id equal	tax equal lowest	tax change below equal	count refseq	count UNITE id mapping
True	species	none	1285	781
False	species	none	36	33
	genus	uniden→uniden	6	5
		species→species	132	95
	family	uniden→genus	3	2
		genus→uniden	10	10
		genus→genus	36	26
	order	uniden→family	12	5
		family→uniden	16	13
		family→family	10	7
	class	uniden→order	4	3
		order→uniden	1	1
		order→order	3	3
	phylum	uniden→uniden	1	1
		uniden→class	10	2
		class→uniden	5	4
	none	uniden→phylum	5	3
		phylum→uniden	6	4

C Appendix: Majority chunk determination analysis tables

Table 20: Comparing the assigned taxonomy of all full-length reference sequences with the taxonomy of the majority chunk, showing the median number of chunks in the BLAST results together with the mean majority count.

Of all 1856 full-length reference sequences:					
tax missing	chunk missing	maj split rank equal	refsequence pk count	num chunks median	maj chunk count median
False	False	False	386	3	7
		True	1073	1	10
	True	NaN	143	NaN	NaN
True	False	False	164	3	6
	True	NaN	90	NaN	NaN

Table 21: Comparing the assigned taxonomy of all truncated reference sequences with the taxonomy of the majority chunk, showing the median number of chunks in the BLAST results together with the mean majority count.

Of all 1767 truncated reference sequences:					
tax missing	chunk missing	maj split rank equal	refsequence pk count	num chunks median	maj chunk count median
False	False	False	382	3	7
		True	1043	1	10
	True	NaN	125	NaN	NaN
True	False	False	143	4	6
	True	NaN	74	NaN	NaN

Table 22: Comparing the majority chunk of all reference sequences before and after truncation, also showing if either the full-length or truncated majority chunk has taxonomy equal to the assigned taxonomy at the chunk splitting level.

Of the 1581 unique full-length reference sequences that have taxonomy hits before and after truncation:								
chunk missing full	chunk missing trunc	maj chunk equal	maj split rank equal either	refsequence pk full count	num chunks full median	num chunks trunc median	maj chunk count full median	maj chunk count trunc median
False	False	False	False	3	5	5	3	4
			True	1	3	3	5	5
		True	False	366	3	3	7	7
			True	1080	1	1	10	10
True	True	False	False	131	NaN	NaN	NaN	NaN

D Appendix: Chunk name lookup table

Num	Chunk name	Num	Chunk name	Num	Chunk name
001	Glomerales	002	Diversisporales	003	Gigasporales
004	Archaeosporales	005	Paraglomerales	006	GS24
007	Thelephorales	008	Gomphales	009	Hygrophoraceae
010	Cortinariaceae	011	Inocybaceae	012	Amanitaceae
013	Lycoperdaceae	014	Agaricaceae	015	Typhulaceae
016	Clavariaceae	017	Hydnangiaceae	018	Tricholomataceae
019	Marasmiaceae	020	Mycenaceae	021	Psathyrellaceae
022	Strophariaceae	023	Callistosporiaceae	025	Omphalotaceae
026	Cyphellaceae	027	Entolomataceae	028	Pluteaceae
029	Lyophyllaceae	030	Pleurotaceae	031	Pterulaceae
032	Bolbitiaceae	033	Catathelasmataceae	034	Stephanosporaceae
035	Cystostereaceae	036	Hymenogastraceae	037	Schizophyllaceae
038	Agaricales fam Incertae sedis	039	Crepidotaceae	041	Physalacriaceae
042	Nidulariaceae	045	Radulomycetaceae	046	Pseudoclitocybaceae
048	Hymenochaetales	049	Polyporales	050	Boletales
051	Russulales	052	Corticiales	054	Auriculariales
055	Geastrales	056	Trechisporales	057	Phallales
058	Gloeophyllales	059	Sebacinales	060	Hysterangiales
061	Atheliales	062	Amylocorticiales	063	Agaricomycetes ord Incertae sedis
064	Tremellodendropsidales	065	GS28	067	Lepidostromatales
070	Tremellales	071	Cystofilobasidiales	072	Trichosporonales
073	Holtermanniales	074	Filobasidiales	075	Cystobasidiomycetes ord Incertae sedis
076	Erythrobasidiales	077	Cyphobasidiales	078	Cystobasidiales
080	Sporidiobolales	081	Microbotryomycetes ord Incertae sedis	082	Microbotryales
083	Leucosporidiales	084	Kriegeriales	086	Agaricostilbales
087	Septobasidiales	088	Pucciniales	089	Platygloeales
090	Helicobasidiales	091	Exobasidiales	092	Entylomatales
093	Tilletiales	094	Georgefischeriales	095	Microstromatales
099	Tritirachiales	100	Geminibasidiales	101	Ustilaginales
102	Urocystidales	104	Atractiellales	105	Spiculogloeales
107	Dacrymycetales	108	Dacrymycetes ord Incertae sedis	110	Malasseziales
111	Wallemiales	116	Dothideomycetes ord Incertae sedis	117	Capnodiales
118	Pleosporales	119	Acrospermales	120	Tubeufiales
121	Botryosphaeriales	122	Dothideales	123	Venturiales
124	Myriangiales	125	Strigulales	127	Abrothallales
128	Mytilinidales	129	Patellariales	130	Mytilinidiales

131	Trypetheliales	132	Hysteriales	133	Jahnulales
135	Stigmatodiscales	137	Valsariales	139	Minutisphaerales
140	Helotiales	141	Erysiphales	142	Rhytismatales
143	Thelebolales	144	Triblidiales	146	Phacidiales
149	Aspergillaceae	150	Trichocomaceae	151	Thermoascaceae
152	Elaphomycetaceae	153	Onygenales	154	Chaetothyriales
155	Verrucariales	156	Phaeomoniellales	157	Mycocaliciales
158	Sclerococcales	159	Coryneliales	160	Pyrenulales
161	Glomerellales	162	Sordariomycetes ord Incertae sedis	163	Microascales
164	Diaporthales	165	Coniochaetales	166	Sordariales
167	Hypocreales	168	Xylariales	169	Magnaporthales
170	Chaetosphaeriales	171	Melanosporales	172	Phyllachorales
173	Pleurotheciales	174	Myrmecridiales	175	Branch06
176	Ophiostomatales	177	Conioscyphales	178	Hypoceales
179	Boliniales	181	Calosphaeriales	182	Annulatascales
183	Togniniales	184	Xenospadicoidales	185	Coronophorales
186	Pararamichloridiales	187	Trichosphaeriales	188	Lulworthiales
189	Phomatosporales	190	Falcocladiales	191	Savoryellales
196	Ostropales	197	Lecanorales	198	Caliciales
199	Rhizocarpales	200	Peltigerales	201	Umbilicariales
202	Acarosporales	203	Pertusariales	204	Arctomiales
206	Trapeliales	207	Teloschistales	208	Lecanoromycetes ord Incertae sedis
209	Leprocaulales	210	Lecideales	211	Baeomycetales
212	Candelariales	213	Sarrameanales	214	GS36
218	Orbiliales	219	GS33	221	Taphrinales
222	Saccharomycetales	223	GS34	224	Symbiotaphrinales
227	Coniocybales	228	Geoglossales	229	Laboulbeniales
230	Pyxidiophorales	231	Archaeorhizomycetales	232	GS31
233	Arthoniales	234	Lichenostigmatales	236	Sareales
237	Lichinales	241	GS05	242	GS08
243	GS07	244	GS03	245	GS11
246	Branch01	247	GS06	249	GS10
250	GS04	252	Branch03	254	Spizellomycetales
255	Rhizophydiales	256	Lobulomycetales	259	Rhizophlyctidales
260	Synchytriales	262	Basidiobolales	263	Endogonales
264	GS21	265	GS22	267	Mucorales
268	GS23	269	Umbelopsidales	270	Blastocladiales
271	GS15	272	Mortierellales	273	Neocallimastigales
274	GS16	277	Olpidiales	278	Monoblepharidales
280	Sanchytriales	281	Zoopagales	283	Kickxellales
284	Entorrhizales				

Table 23: Lookup table for chunk numbers and names.