**Introduction**

The study of biota diversity has been a long-living topic fascinating a number of investigators in biological science ever since the time of natural history. Starting from recording the macrofauna and macroflora of an opening ecosystem, the focus on species recognition has been gradually shifted from the visible world to the micro ecosystem, the wonderland of microorganism. Unlike studying the species composition of macrobiota, the invisible nature of microbiota brings researchers a huge barrier in identifying and recording the composition of species diversity. The manageable microorganisms have once been the only group of species that can be identified from the standard protocol like, for example, isolating and culturing pure clones from a bunch of mixed species to determine their classification with biochemical analyses; those species that cannot be cultured then will leave no trace of themselves. Such workflows, unsurprisingly, depict only a cruel contour of the true picture of species diversity in microbiota; moreover, cell culturing is a time-consuming step, which seriously hampers the efficiency of workflow. Consequently, species identification without cell culturing became the critical point in build an efficient workflow in studying biodiversity of microbiota.

The emergence of metagenomics, however, solves the problem once and for all. Metagenomics refers to the study of retrieving and analyzing the genetic information from environmental samples directly without species identification in advance. A snapshot of species composition can now be recorded through the powerful sequencing technology and analyzed by a variety of bioinformatic tools. A series of software has been published and designed for different purposes. For example, PHACCS, a bioinformatic tool designed for estimating of the structure and diversity of uncultured viral communities; QIIME, an open-source bioinformatics pipeline for performing microbiome analysis including quality filtering, OTU picking, taxonomic assignment, phylogenetic reconstruction, and diversity analyses; MLST, a nucleotide sequence based approach for the unambiguous characterization of isolates of bacteria and other organisms via defining the distinct pattern of alleles combination in a series of loci involved in microbial metabolism. With the aid of metagenomics, the study microbiota has become a popular topics among a number of biological fields including environmental science, industrial fermentation, gastrointestinal health, and even mental disorders, showing the amazing linkage between the fluctuation of microbiota and a great deal of phenomena including soil fertility, diet preference, metabolomics, and human health. These studies unveil the hierarchical nature of the biological system, suggesting that the facts of macroscopic scale are tightly regulated by the composition of microbiota.

Whether the pipelines of metagenomics provide great help in documenting the biodiversity of multicellular communities would be a challenging and valuable trial. With the great success in the study of microbial diversity, metagenomics has been proved to have great potential in identifying biota diversity directly from environmental samples. Here, by using a RNA-seq dataset extracted from a mixed collection of body fragments from several species of arthropod, we tested the efficiency of two pipelines, based on Basic Local Alignment Search Tool (BLAST) against all known sequences of NADH dehydrogenase subunit 6 from NCBI database, with either raw RNA seq or de novo transcripts assembly in advance, in identifying species composition of the meta-transcriptomic dataset. Collecting originally from an activity of BioBlitz, the popular field study of surveying and recording all the living species within a designated area, we have a complete inventory of the species contributing to the RNA-seq dataset as a standard in evaluating the efficiency of each of the pipelines.

More than showing the potential of metagenomics in species identifying biodiversity of arthropods, the workflows built in this study provide great value in agricultural application. Soil mesofauna, the study of tiny invertebrates such as mites, nematodes, or spiders that are usually 0.1 mm to 2 mm in length bears a great requirement for establishing the metagenomic pipelines for species identification directly from environmental samples. The complex communities of these small creatures, dwelling in soil or in a leaf cover layer on the soil surface, are hard to be completely covered and analyzed by traditional workflow based on morphology analyses, and, surprisingly, only a handful of publications have tried to address such questions with sequencing technology. Considering the great number of soil-transmitted pests that erode the total output value of agriculture globally, our work, contributing to species identification of invertebrates from meta-transcriptomic data, suggests a possibility of building a pipeline testing the presence of pests from soil samples. Our study shed light on applying metagenomics to species identification of invertebrates, providing a possible alternative of pest control in industrial agriculture.

**Materials and Methods**

**Dataset:**

The provided dataset comes from a BioBlitz sampling. It comprises 202,209,951, 100 bases-long, paired-end reads from a RNA-seq Illumina run.

**Quality check:**

The quality of the dataset was assessed using the software FastQC (Andrews, 2010) using the following command:

fastqc -t 2 -o /ptmp/660\_project\_bugs/fastqc\_output \

bioblitz/BioBlitz\_NoIndex\_L008\_R1\_001.fastq.gz bioblitz/BioBlitz\_NoIndex\_L008\_R2\_001.fastq.gz

**Pipeline I:**

First, we obtained the nucleotide sequence for every mitochondrial genome present into NCBI RefSeq database from (<ftp://ftp.ncbi.nlm.nih.gov/refseq/release/mitochondrion/>, specifically the files “[mitochondrion.1.1.genomic.fna.gz](ftp://ftp.ncbi.nlm.nih.gov/refseq/release/mitochondrion/mitochondrion.1.1.genomic.fna.gz)” and “[mitochondrion.2.1.genomic.fna.gz](ftp://ftp.ncbi.nlm.nih.gov/refseq/release/mitochondrion/mitochondrion.2.1.genomic.fna.gz)”). We then aligned the dataset against this mitochondrial genomic database using Bowtie2 (Langmead and Salzberg, 2012). For this we used the following commands:

For concatenate both files

cat mitochondrion.1.1.genomic.fna mitochondrion.2.1.genomic.fna > mitochondrion.all.fna

For building the bowtie2 reference genome index

bowtie2-build mitochondrion.all.fna mito\_gen

For making the alignment

bowtie2 -p 32 --al bugs\_unp\_al.fastq --al-conc bugs\_conc\_al.fastq --un bugs\_unp\_unal.fastq \

--un-conc bugs\_nonconc.fastq -x mitoGen/mito\_gen -1 BioBlitz\_NoIndex\_L008\_R1\_001.fastq.gz \

-2 BioBlitz\_NoIndex\_L008\_R2\_001.fastq.gz -S bugs\_vs\_mito.sam

The --al and --al-conc arguments output the reads that align unpaired and concordant, respectively, to the database. Whereas the --un and --un-conc arguments output the reads that have one unaligned pair or both pairs unaligned, respectively.

We then performed a “de novo” transcript assembly on the concordantly aligned reads obtained from the Bowtie2 alignment step. This was achieved using Trinity (Grabherr *et al*, 2011) with the following command:

Trinity --seqType fq --left /ptmp/660\_project\_bugs/bugs\_conc\_al.1.fastq \

--right /ptmp/660\_project\_bugs/bugs\_conc\_al.2.fastq --CPU 48 --max\_memory 10G

For the next step, we extracted the aminoacidic sequence of the mitochondrial protein “NADH dehydrogenase subunit 6” (ND6) for every specie in NCBI RefSeq database (<ftp://ftp.ncbi.nlm.nih.gov/refseq/release/mitochondrion/>, specifically from [mitochondrion.1.protein.faa.gz](ftp://ftp.ncbi.nlm.nih.gov/refseq/release/mitochondrion/mitochondrion.1.protein.faa.gz) and [mitochondrion.2.protein.faa.gz](ftp://ftp.ncbi.nlm.nih.gov/refseq/release/mitochondrion/mitochondrion.2.protein.faa.gz)). We then performed a tBLASTn alignment on our previously “de novo” assembled transcripts versus the ND6 proteins using BLAST+ software (Camacho *et al*. 2009). For this we used the following commands:

For making the BLAST database

makeblastdb -in trinity\_out\_dir/Trinity.fasta -dbtype nucl -out blast\_trinityDB \

-title trinity\_out

For making the tBLASTn alignment

time tblastn \

-task tblastn-fast \

-query nd6\_2-mito.faa \

-db /ptmp/BCB660/660\_project\_bugs/blast\_trinityDB \

-evalue 0.001 \

-num\_threads 16 \

-max\_target\_seqs 1000000000 \

-outfmt '7 qseqid sseqid qlen slen qstart qstop sstart sstop bitscore pident positive evalue'\

> nd6\_2-blast.tab

After getting the BLAST results, we used a bash script to re-format the obtained file for an easier extraction of the required data. This script will remove the comment lines from the BLAST output and will add a column with the name of the specie for every ND6 protein accession number (see supplementary materials for scripts).

A summary of the script is provided below (for a full detailed and commented version, see supplementary materials):

join -t $'\t' \

<( cat $protein\_reference |

sed -nr 's/>([^ ]+).\*\[(.\*)\]/\1\t\2/p' |

tr ' ' '\_' | sort ) \

<( grep -Pv '^#' $input\_blast\_result | sort ) > $output\_blast\_result

Once obtained the re-formatted BLAST results, we calculated the most represented species in the BioBlitz sample using a R script (see supplementary materials for the commented script).

The resulting file is a list of the most represented species, which was used to generate a phylogenetic tree using NCBI Common Tree web site (<https://www.ncbi.nlm.nih.gov/Taxonomy/CommonTree/wwwcmt.cgi>).

Finally, the obtained tree was visualized using TreeGraph2 software (Stöver and Müller, 2010).

**Pipeline II**

For this, we first used the provided reads in the dataset to build a BLAST database using the following command (for the full script, please refer to supplementary materials):

For converting the reads from fastq to fasta format

zcat $r1 $r2 |

awk 'NR % 4 == 1 {print ">" $0} NR % 4 == 2 {print}' > $tmpfa

For building the BLAST database

makeblastdb -in $tmpfa -dbtype nucl -out blastdb/bioblitz -title bioblitz

After building the BLAST database we aligned the mitochondrial protein “NADH dehydrogenase subunit 6” (ND6) dataset (the same dataset from pipeline 1) to it using the following command:

time tblastn \

-task tblastn-fast \

-query nd6\_2-mito.faa \

-db /ptmp/BCB660/arendsee/bcb660-project/blastdb/bioblitz \

-evalue 0.001 \

-num\_threads 16 \

-max\_target\_seqs 1000000000 \

-outfmt '7 qseqid sseqid qlen slen qstart qstop sstart sstop bitscore pident positive evalue' \

> nd6\_2-blast-all.tab

After getting the BLAST results, we performed the same steps as in Pipeline I. Briefly, used a bash script to re-format the obtained file, calculated the most represented species in the BioBlitz sample using a R script, generate a phylogenetic tree using NCBI Common Tree web site and visualized the tree using TreeGraph2 (as stated before, for a full description of the scripts please refer to the supplementary materials).

**Results / Discussion**

Quality check: The FastQC run reported the following errors: Per base sequence quality (there was a considerable drop in base call quality at the last bases), Per base sequence content, Sequence duplication levels and Kmer content (figure X). Since our project is to identify the different species present in a BioBlitz sampling, we decided to keep all reads without performing any trimming or filtering process on them, regardless of the quality check results.

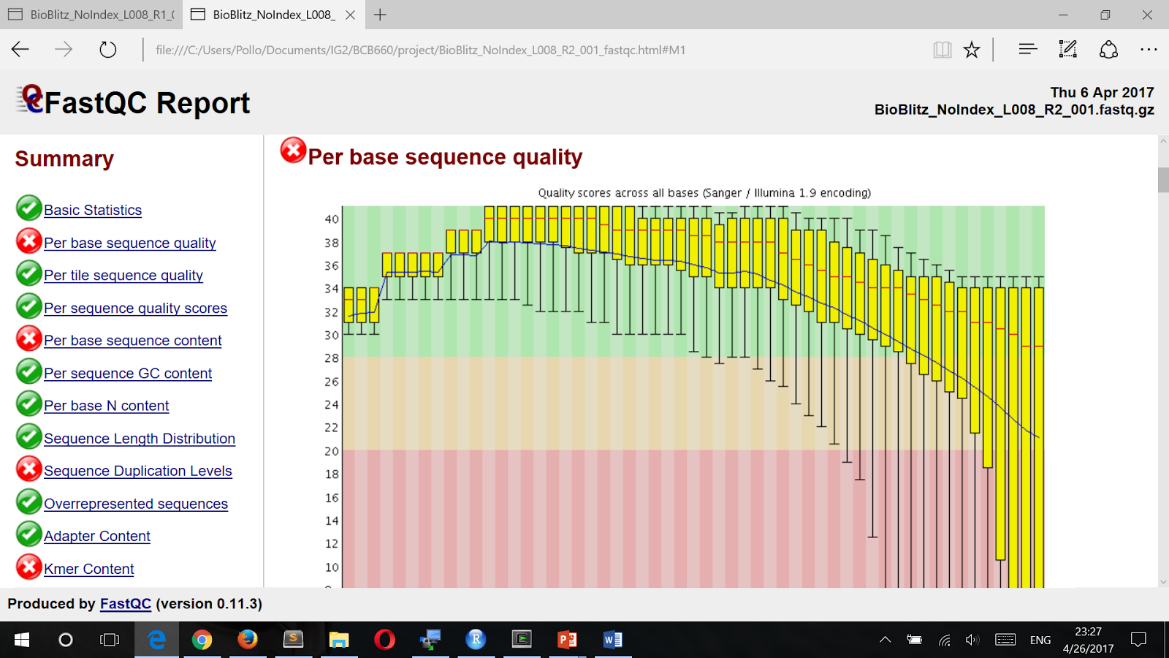
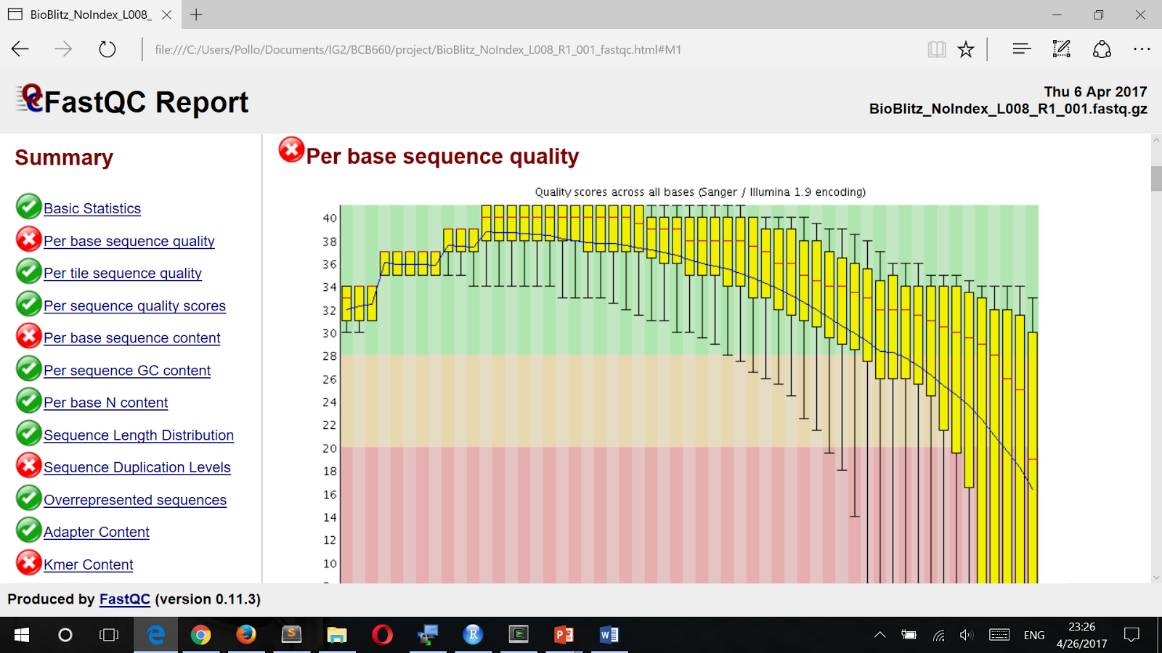


Figure X. Results from FastQC analysis performed on the provided dataset. On the left side of each panel we can appreciate the result for each test, failed tests are depicted by red circles with a white cross. On the right side of each panel we see the phred value (average and median) per nucleotide position. Upper panel: analysis on forward reads; lower panel: analysis on reverse reads.

Pipeline I: From the downloaded RefSeq mitochondrion database we used all of 7,809 available genome sequences to build our Bowtie2 reference index. The Bowtie2 alignment gave us a total of 8,949,517 paired reads that aligned concordant to our mitochondrial genomes database (roughly 4.4% of the total reads). The Trinity “de novo” assembly on these files predicted 534 transcripts. These transcripts were used as the database for our BLAST search which gave us a total of 7,486 hits. Using tBLASTn alignment score and results filtering, we could indicate how many times a certain specie’s ND6 protein had a maximum score hit for any different predicted transcript. We used these results to suggest this specie’s RNA abundance in out BioBlitz sample. From this data, we obtain that 12 species were likely to be present. Every specie was represented either 1 or 2 times at most. The exception was one specie, represented 25 times (the butterfly *Tirumala limniace*, table X and figure X).

|  |  |
| --- | --- |
| **Table X. List of species present according to results from Pipeline I** | |
| **Specie Name** | Max-score hits count |
| ***Camponotus atrox*** | 2 |
| ***Chrysopa pallens*** | 1 |
| ***Chrysoperla nipponensis*** | 1 |
| ***Lamproptera curius*** | 1 |
| ***Narceus annularus*** | 1 |
| ***Neoneuromus tonkinensis*** | 1 |
| ***Nevromus exterior*** | 1 |
| ***Pteronarcella badia*** | 1 |
| ***Ruspolia dubia*** | 1 |
| ***Simosyrphus grandicornis*** | 2 |
| ***Tenodera sinensis*** | 1 |
| ***Tirumala limniace*** | 25 |

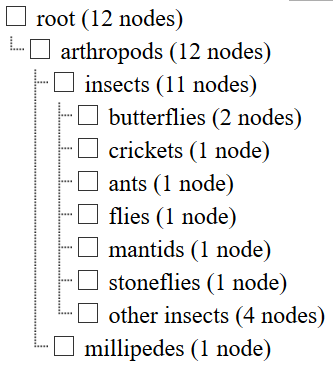
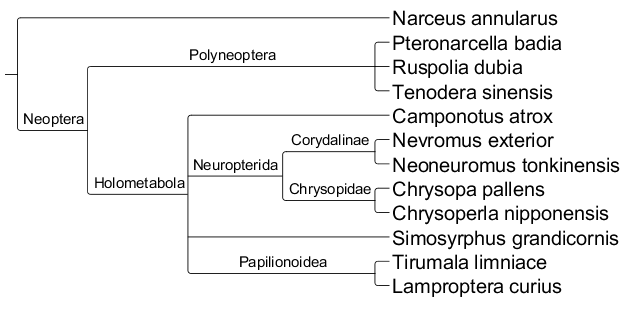


Figure X. Phylogenetic tree obtained from Pipeline I (Left) and the different sub-categories present, according to NCBI taxonomy web site (Right).

Pipeline II: We performed a tBLASTn using the entire 202,209,951 BioBlitz dataset read pairs as our database and all of the 7,418 ND6 protein sequences available on NCBI RefSeq as our query. The BLAST alignment produced 4,646,398 total hits. Again, using tBLASTn alignment score and results filtering, we could indicate how many times a certain specie’s ND6 protein had a maximum score hit for any different read of our BioBlitz dataset. We used these results to suggest this specie’s RNA abundance in our sample. From this processing, we obtained a total of 214 species with a maximum-score hit. In order to narrow down our results, we applied an extra filter for extracting only those species with a number of maximum hits above a defined threshold. Table x shows the different selected threshold values and the number of species obtained. We could observe that setting our max-score hits threshold above 30 showed little effect on the number of species extracted. Nevertheless, we decided to use 200 max-score hits as threshold value to narrow as much as we can our output. Figure X shows the obtained tree.

|  |  |
| --- | --- |
| Table X. number of species showing different max-score hit thresholds | |
| Number of max-score hit threshold | Number of species |
| 0 | 214 |
| 10 | 78 |
| 20 | 61 |
| 30 | 52 |
| 40 | 49 |
| 50 | 49 |
| 60 | 47 |
| 70 | 46 |
| 100 | 40 |
| 200 | 33 |

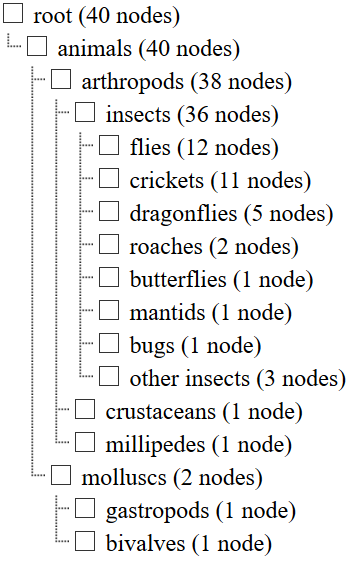
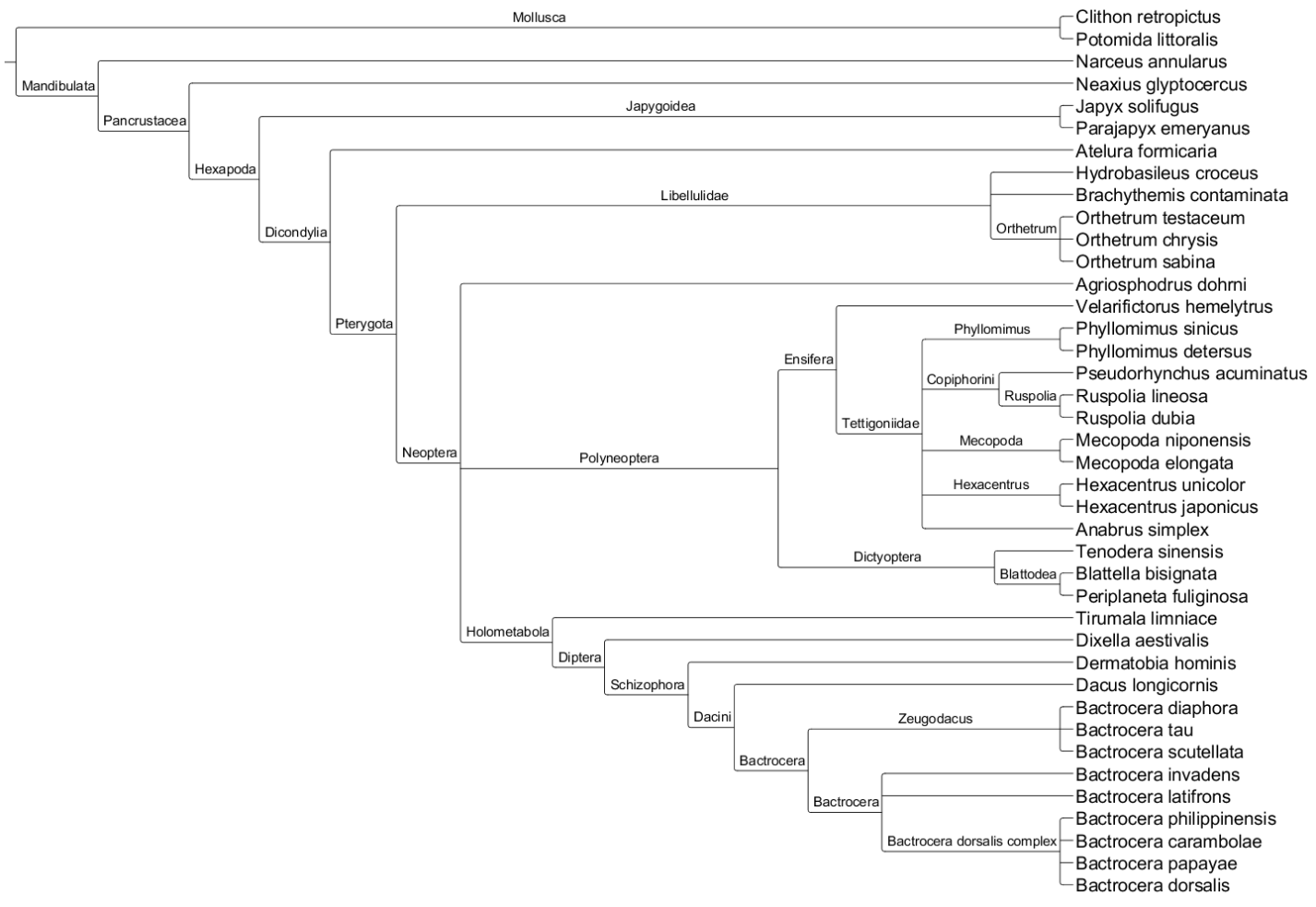


Figure X. Phylogenetic tree obtained from Pipeline II (Right) and the different sub-categories present, according to NCBI taxonomy web site (Lower-left).

**Conclusions**

Author's Contribution

Zebulum Arendsee designed and wrote most of the scripts used. Ang-Yu Liu performed BUSCO analyses (though we did not include these results, they represented a great deal of work) and also performed exhaustive literature review. Christian Montes developed some minor scripts and performed the analyses using scripts provided by ZA. All of the authors contributed to the design and writing of this project.

References

Andrews S. (2010). FastQC: a quality control tool for high throughput sequence data. Available online at: <http://www.bioinformatics.babraham.ac.uk/projects/fastqc>

Camacho, C., Coulouris, G., Avagyan, V., Ma, N., Papadopoulos, J., Bealer, K. and Madden, T.L., 2009. BLAST+: architecture and applications. *BMC bioinformatics*, *10*(1), p.421.

Grabherr, M.G., Haas, B.J., Yassour, M., Levin, J.Z., Thompson, D.A., Amit, I., Adiconis, X., Fan, L., Raychowdhury, R., Zeng, Q. and Chen, Z., 2011. Trinity: reconstructing a full-length transcriptome without a genome from RNA-Seq data. *Nature biotechnology*, *29*(7), p.644.

Langmead, B. and Salzberg, S.L., 2012. Fast gapped-read alignment with Bowtie 2. *Nature methods*, *9*(4), pp.357-359.

Stöver, B.C. and Müller, K.F., 2010. TreeGraph 2: combining and visualizing evidence from different phylogenetic analyses. *BMC bioinformatics*, *11*(1), p.7.

Supplementary materials

Scripts used:

1. Get mitochondrial genomes

# Retrieve mitochondrial genomes

# If they are already downloaded, do nothing

outdir=data\_mitochondria

if [[ ! -d $outdir ]]

then

mkdir $outdir

wget \

-P $outdir -nd -r -A gz \

ftp://ftp.ncbi.nlm.nih.gov/refseq/release/mitochondrion

cd $outdir

gunzip \*gz

fi

1. Build a BLAST database

module load ncbi-blast

[[ -d blastdb/ ]] || mkdir blastdb

r1=/ptmp/bioblitz/BioBlitz\_NoIndex\_L008\_R1\_001.fastq.gz

r2=/ptmp/bioblitz/BioBlitz\_NoIndex\_L008\_R2\_001.fastq.gz

# Check existence of the input files, die if missing

[[ -f $r1 && -f $r2 ]] || ( echo "Cannot open fastq files: $r1 $r2" >&2 && exit 1 )

tmpfa=fastq-fasta-extract\_DELETEME.fa

# If z.fa already exists, and isn't an empty file, don't remake it

if [[ ! -s $tmpfa ]]

then

# Extract and combine the fasta sequences from the two fastq files

# NOTE: I am not doing any cleaning or trimming here

zcat $r1 $r2 |

awk 'NR % 4 == 1 {print ">" $0} NR % 4 == 2 {print}' > $tmpfa

fi

if [[ ! -d blastdb ]]

then

# Make the blast database, to access it with blast, use a command of the form:

# $ blastp -db /blastdb/bioblitz -query whatever.faa [options]

# makeblastdb -in $tmpfa -dbtype nucl -out blastdb/bioblitz -title bioblitz

echo "making a db" > &2

else

echo "not making a db, already exists" > &2

fi

# If makeblastdb fails,

if [[ $? -eq 0 ]]

then

rm $tmpfa

exit 0

else

echo "makeblastdb run failed" >&2

exit 1

fi

1. run the tBLASTn alignment

module load python/3.6.0

module load ncbi-blast

[[ -d smof ]] || git clone https://github.com/arendsee/smof

all\_pro='data\_mitochondria/mitochondrion\*protein.faa'

base=nd6\_2

protein\_name="NADH dehydrogenase subunit 6";

blastresult=${base}-blast.tab

my\_pro=data\_mitochondria/${base}-mito.faa

cat $all\_pro | smof/smof.py grep "$protein\_name" > $my\_pro

time tblastn \

-task tblastn-fast \

-query $my\_pro \

-db blastdb/bioblitz \

-evalue 0.001 \

-num\_threads 8 \

-max\_target\_seqs 1000000000 \

-outfmt '7 qseqid sseqid qlen slen qstart qstop sstart sstop bitscore pident positive evalue' > $blastresult

1. Re-formatting BLAST output (getting species names)

#!/usr/bin/bash

set -u

# USAGE:

# # assuming you are running this from the bcb660-project folder:

# scripts/autoslurm.sh scripts/add-names-to-blast-result.sh nd6-blast.tab

# This should be the name of the output file from BLAST, e.g. nd6-blast.tab

# 1. The file may have comments inside

# 2. The file is expected to have the extension `.tab`

input\_blast\_result=$1

# --- You shouldn't need to change this

# The protein fasta files are used just to extract the map between reference

# and species name. The fasta headers are assumed to have the format:

# >REFERENCE ... [ SPECIES\_NAME ]

protein\_reference=data\_mitochondria/mitochond\*protein.faa

# This is the name of the output file, the parameter expansion below replaces

# the extension on the input.

output\_blast\_result=${input\_blast\_result/.tab/-with-names.tab}

# Extract a reference to name map from the protein sequence file

# Join the map table with the blast result table

# 1. This adds the scientific species names as the second column in the output file

# 2. Replaces space in the species name with underscores

# 3. This will remove all comments from the blast result file

join -t $'\t' \

<( cat $protein\_reference |

sed -nr 's/>([^ ]+).\*\[(.\*)\]/\1\t\2/p' |

tr ' ' '\_' | sort ) \

<( grep -Pv '^#' $input\_blast\_result | sort ) > $output\_blast\_result

1. R script for extracting the most represented species

require(readr)

require(dplyr)

require(magrittr)

# Provide the name for the re-formatted BLAST output

blastfile <- 'nd6-blast-with-names.tab'

d <- read\_tsv(blastfile, col\_names=FALSE, comment="#") %>%

select(X2,X3,X8) %>%

rename(species=X2,read=X3,bitscore=X8)

d$species <- gsub("\_", " ", d$species)

# total number of species

(total <- length(unique(d$species)))

# species list (you can plug this into NCBI common tree)

write(unique(d$species), file="species-with-hits.txt")

# Get just the matches that are maximum for each read

d <- group\_by(d, read) %>% dplyr::filter(bitscore == max(bitscore))

# number of species with maximum against at least one read

(withmax <- length(unique(d$species)))

(counts <- summary(factor(d$species), maxsum=Inf))

write(unique(d$species), file="species-with-a-max-hit.txt")

hist(log(counts))

# modify this to get the list of species with more than N max-score hits

names(counts[counts > 200]) %>%

write(file="species-with-more-than-200-max-hits.txt")