Task 1 - Projecting gene annotations with TOGA

TOGA is a method that integrates gene annotation, inferring orthologs, and classifying genes as intact or lost. We want to use TOGA to detect genetic innovations in *Lasallia pustulata*. Therefore, we are searching for genes that are present in *Lasallia hispanica* but lost in *Lasallia pustulata*.

Installation

1. Create an own environment for TOGA and install nextflow:

```
conda create --name toga python=3.7.3
conda activate toga
conda install -c bioconda nextflow
conda install -c anaconda biopython
```

2. Clone TOGA from github and install it:

```
git clone https://github.com/hillerlab/TOGA.git
cd TOGA
python3 -m pip install -r requirements.txt --user
./configure.sh
# ignore the warnings
./run_test.sh micro
```

Your output should look something like this in the end:

```
#### STEP 11: Cleanup: merge parallel steps output files
Saved results to /home/ecoevo01/TOGA/TOGA/micro_test_out
Done! Estimated time: 0:01:40.027813
Program finished with exit code 0

JH567521    463144    506100    ENST00000262455.1169    1000    -
463144    506100    0,200,255    8
102,103,142,112,117,58,116,185,0,1982,30295,31351,36911,38566,41322,427
71,
JH567521    395878    449234    ENST00000259400.1169    1000    +
395878    449234    0,0,200    7    123,66,226,116,51,87,240,
0,11871,38544,45802,45994,52305,53116,
JH567521    299723    336583    ENST00000618101.1169    1000    +
299723    336583    0,0,200    7    28,923,130,173,200,179,248,
0,1256,6085,6677,19146,21311,36612,
Success!
```

Input preparation

TOGA uses at least 3 different input data:

1. Gene annotation of the **reference genome**, in our case from *L.hispanica*, as a bed-12 file
2. Genome alignment between the **reference** (*L.hispanica*) and **query genome** (*L.pustulata*) as a chain file
3. Reference and query genome sequences as 2bit files

We need to create the required inputs for TOGA from our assemblies. This require different tools, including *lastz*, *axtChain*, *chainNet*, *chainSort*, *chainCleaner*, *faToTwoBit*, *gff3ToGenePred*, *GenePredToBed*, *RepeatFiller* and some self-written scripts. The tools from UCSC (such as *faToTwoBit*, *axtChain*,..) are pre-compiled and can be downloaded here using the command below. Please note that you need to find the correct name for the tools that are displayed on that website, for example *lastz-1.04.00* for *lastz*.

```
rsync://hgdownload.soe.ucsc.edu/genome/admin/exe/linux.x86_64/<tool_name> ./
```

We have downloaded and installed all of them for you  Add the following folder to you variable $PATH by adding this line to your `~/.bashrc` file:

```
export PATH=/share/gluster/Projects/vinh/ecoevo/src/for_toga:$PATH
```

Then, apply the change with `source ~/.bashrc` (or restart the terminal) and re-activate toga conda environment.

Now, we can generate the input files for TOGA.

1. First of all, we convert our **assemblies** from *L. pustulata* and *L. hispanica* into 2bit files using *faToTwoBit*:

   ```
   faToTwoBit L_hispanica_assembly.fasta L_hispanica.2bit
   faToTwoBit L_pustulata_assembly.fasta L_pustulata.2bit
   ```

2. As the next step, we do the whole-genome alignment with the workflow mentioned in Pippel et. al. 2020 that requires different steps and tools
   1. Do genome alignment with lastz. This will take about 6 hours, please use Slurm with 1 CPU and 4GB Ram.  While *lastz* is running, you can do step 2c and 3 

      ```
      lastz L_hispanica.2bit L_pustulata.2bit --action:target=multiple --hspthresh=2400 --gappedthresh=3000 --format=axt --out=hispanica_pustulata_aln.axt
      ```

   2. Compute chain file from axt alignments using axtChain

      ```
      axtChain hispanica_pustulata_aln.axt L_hispanica.2bit L_pustulata.2bit hispanica_pustulata_aln.chain -linearGap=medium
      ```

   3. Get length of each sequence in the L_hispanica and L_pustulata assembly using BASH or the scripts below. You will need to:

      ```
      python /share/gluster/Projects/vinh/ecoevo/src/for_toga/fasta_len.py <LasHis.fa> <LasHis.sizes>
      ```

```
python
/share/gluster/Projects/vinh/ecoevo/src/for_toga/fasta_len.py
<LasPus.fa> <LasPus.sizes>
# make sure that the identifiers in your .sizes files match the
ones in your .axt file
```

4. Compile chain nets with *chainNet*

```
# run chainSort
chainSort hispanica_pustulata_aln.chain
hispanica_pustulata_aln.sorted.chain
# run chainNet
chainNet hispanica_pustulata_aln.sorted.chain L_hispanica.sizes
L_pustulata.sizes hispanica.net pustulata.net
```

5. Improve genome alignment using chainCleaner

```
chainCleaner hispanica_pustulata_aln.chain L_hispanica.2bit
L_pustulata.2bit hispanica_pustulata_aln.cleaned.chain
hispanica_pustulata_aln.bed -net=hispanica.net linearGap=medium
```

6. Comprehensively align repetitive genomic regions with RepeatFiller (using Slurm again with 1 CPU and 4GB memory. This will take about 1.5 hours)

```
RepeatFiller.py -c hispanica_pustulata_aln.cleaned.chain -T2
L_hispanica.2bit -Q2 L_pustulata.2bit | tee
hispanica_pustulata_aln.cleaned.filled.chain
```

7. The output from *RepeatFiller* (saved in `hispanica_pustulata_aln.cleaned.filled.chain`) is the chain file that we want to use as an input for TOGA. Sometimes the *RepeatFiller* output includes empty lines which must be deleted. Use the following command:

```
sed -i '/^$/d' hispanica_pustulata_aln.cleaned.filled.chain
```

3. Convert the annotation gff3 file of the reference genome *L. hispanica* into bed file using *gff3ToGenePred* and *GenePredToBed*

```
gff3ToGenePred L_hispanica.gff3 L_hispanica.genePred
genePredToBed L_hispanica.genePred L_hispanica.bed
```

4. After those steps, you will have a lot of files. We suggest that you copy these 4 input files for TOGA to an additional folder before running the tool:
   1. The annotation in bed format `L_hispanica.bed`
   2. The genome alignment as chain file `hispanica_pustulata_aln.cleaned.filled.chain`
   3. 2 genome sequences in 2bit format `L_hispanica.2bit` and `L_pustulata.2bit`

Run TOGA

Use a Slurm script with 12 CPUs and 8GB memory to run TOGA

```
python3 /path/to/your/downloaded/TOGA/toga.py
hispanica_pustulata_aln.cleaned.filled.chain L_hispanica.bed
L_hispanica.2bit L_pustulata.2bit
```

TOGA produces different output files that you can look up here. The most interesting output file for us is `loss_summ_data.tsv`, which contains the classifications for each projection, transcript, and gene. TOGA identifies the following classes:

```
    N - no data due to technical reasons (such as CESAR memory requirements)
    PG - no orthologous chains identified, TOGA projected transcripts via
paralogous chains and cannot make any conclusion
    PM - partial & missing. Most of the projection lies outside scaffold
borders
    L - clearly lost (several inactivating mutations)
    M - missing, assembly gaps mask >50% of the prediction CDS
    G - "grey", there are inactivating mutations but not enough evidence for
"clearly lost" class. In other words: neither lost nor intact
    PI - partially intact: some fraction of CDS is missing, but most likely
this is intact
    I - clearly intact
    UL - uncertain loss (few inactivating mutations localized in a single
exon)
```

We are interested in the genes that are present in *L. hispanica* but lost in *L. pustulata* not by technical artifacts (e.g. errors from gene annotation process).

1. Search for TRANSCRIPTs and identify all gene IDs that are marked as L (lost), M (missing), PM (partial missing) or UL (uncertain loss) in the file `loss_summ_data.tsv`. These four sets of genes will form our genes of interest
2. Use genome viewer of g-nom to investigate some of those genes and their gene neighborhood in the *L. hispanica* contig assembly

Task 2 - Phylogenetic profile analysis with fDOG

From our previous analysis, we obtained a list of genes, which are missing in *L. pustulata* but present in *L. hispanica*. But,

1. What are they, or, what are their characteristics?
2. How old are they, or do they loss only in *L. pustulata* or also in other taxonomic clades? In other words, are they *L. pustulata* specific losses or *L. hispanica* specific gains?

In order to answer those questions, we are going to study their phylogenetic profiles across a wide range of species by using fDOG and PhyloProfile. fDOG is a directed ortholog search tool, which can look for orthologs of a single gene of interest in a set of search taxa. These search organisms can be diverse, from closely related to distantly taxa to our reference species, from which the seed genes are originated.

The taxon set we will use for this analysis consists of *L. hispanica*, *L. hispanica*, the close related species *U. muehlenbergii*, and the set of 78 QfO taxa (what is the “QfO”? Why do we work with these

78 taxa?).