Using MAKER for Genome Annotation

The example here is from a workshop by Mark Yandell Lab

Further readings:

1. Yandel Lab Workshop. http://weatherby.genetics.utah.edu/MAKER/wiki/index.php/MAKER_Tutorial_for_WGS_Assembly_and_Annotation_Winter_School_2018 .
2. MAKER protocol from Yandell Lab. It is good reference. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4286374/
3. A protocol contributed by Daren Card. Especially the part of training Augustus model is handy when you work on your own genome. In this exercise, you will only use SNAP for gene prediction. https://gist.github.com/darencard/bb1001ac1532dd4225b030cf0cd61ce2

Practice using Maker

1. Prepare working directory (replacing "xxxxx" with your user ID). Copy the data file from /shared_data/annotation2018/ into /workdir/xxxxx, and de-compress the file. You will also copy the maker software directory to /workdir. The maker software directory including a large sequence repeats database. It would be good to put it on /workdir which is on local hard drive.

```
mkdir /workdir/xxxxx

mkdir /workdir/xxxxx/tmp

cd /workdir/xxxxx

cp /shared_data/annotation2018/maker_tutorial.tgz ./

cp -rH /programs/maker/ ./

cp -rH /programs/RepeatMasker  ./

tar -zxf maker_tutorial.tgz

cd maker_tutorial

 ls -1
```

2. Repeat masking the genome and align transcriptome sequences to the genome.
   We will use the example data in the directory "example_02_abinitio".

Optional step: it is recommended by the authors to build a custom repeat masking database using the genome sequence you are working on. A repeat fasta file can be used for repeat masking in the next step. We will not use the custom database for this exercise, but I give you the commands here.

```
/programs/RepeatModeler/BuildDatabase -name pyu -engine ncbi pyu_contig.fasta

/programs/RepeatModeler/RepeatModeler -pa 2 -engine ncbi -database pyu  >& repeatmodeler.log
```

- To run MAKER on BioHPC computers, set maker environment and create the MAKER control files:

```
export PATH=/workdir/xxxxx/maker/bin:/workdir/xxxxx/RepeatMasker:/programs/snap:$PATH

export ZOE=/programs/snap/Zoe

export LD_LIBRARY_PATH=/programs/boost_1_62_0/lib

cd example_02_abinitio

maker -CTL
```

- Three control files are created: maker_bopts.ctl, maker_exe.ctl, maker_opts.ctl.
  In most cases, you do not need to change content of the files maker_bopts.ctl and maker_exe.ctl.
- You do need to modify the file maker_opts.ctl. You can use Notepad++ (Windows) or BBEdit (Mac) to edit this file.

Open the maker_opts.ctl file in a text editor. Modify the following values. Put the modified file in the same directory “example_02_abinitio”.

```
genome=pyu_contig.fasta

est=pyu_est.fasta
protein=sp_protein.fasta

model_org=simple    #If you know the species, put species name here
rmlib=      #fasta file name of your custom repeat sequence
softmask=1

est2genome=1
protein2genome=1

TMP=/workdir/xxxxx/tmp   #important for big genome, as the default /tmp is too small
```

The modified **maker_opts.ctl** file instructs MAKER to do two things.

a) Run RepeatMasker. "model_org=simple" only mask the low complexity sequence (e.g. "AAAAAAAAAAAAA". To mask transposon elements, you can put the species name here, which instruct RepeatMaskter to use the general Repbase database. Or you can put you custom fasta file next to "rmlib=". "softmask=1" to make sure converting repeats to lower case, instead of converting to "N". This is important so that simple repeats within genes can still be annotated as part of gene.

b) The instructions ("est2genome=1" and "protein2genome=1") tell MAKER to align the transcript sequences from the pyu_est.fasta file and protein sequences from the sp_protein.fasta file to the genome and infer evidence supported gene model.

Now, you will run **maker** using MPI. MPI parallelized the maker jobs. As this will take a long time, you will need to run it in "screen" mode. "-n 2" tells maker to run two processes simultaneously, you can increase the number to 40 if you use the 2nd generation medium memory computer. "-q" tells maker to run in quiet mode, not showing too detailed status in the log file. "-base pyu_rnd1" tell maker to write output files in a directory named "pyu_rnd1.maker.output".

```
screen

/usr/local/mpich/bin/mpiexec -n 2 maker -base pyu_rnd1 >& log1 &

## then press ctrl-"a" followed by pressing "d" to detach from screen
```

You can monitor the progress by the command "top" or "tail log". The "tail log" command would show the last 10 lines in the log file. If the job is finished, the log file should end with the line:

Maker is now finished!!!

3. Train the SNAP model using the alignment results from last step.

If you just started a new ssh session, do not forget to set the environment again in the new session, skip these commands if you continue in the same ssh session.

```
export PATH=/workdir/xxxxx/maker/bin:/workdir/xxxxx/RepeatMasker:/programs/snap:$PATH

export ZOE=/programs/snap/Zoe

export LD_LIBRARY_PATH=/programs/boost_1_62_0/lib
```

SNAP is software to do *ab initio* gene prediction from a genome. In order to do gene prediction with SNAP, you will first train a SNAP model with alignment results of transcriptome sequences to the genome, which you produced in the previous step. The following commands will convert the MAKER output from last step to input files for building a SNAP mode.

```
cd /workdir/xxxxx/maker_tutorial/example_02_abinitio

mkdir snap1

cd snap1

gff3_merge  -d  ../pyu_rnd1.maker.output/pyu_rnd1_master_datastore_index.log

maker2zff -l 50 -x 0.5  pyu_rnd1.all.gff
```

The "-l 50 -x 0.5" parameter in maker2zff commands specify that only gene models with AED score>0.5 and protein length>50 are used for build models. You will find two new files: genome.ann and genome.dna. Now you will run the following commands to train SNAP. The basic steps for training SNAP are first to filter the input gene models, then capture genomic sequence immediately surrounding each model locus, and finally uses those captured segments to produce the HMM. You can explore the internal SNAP documentation for more details if you wish.

```
fathom -categorize 1000 genome.ann genome.dna

fathom -export 1000 -plus uni.ann uni.dna

forge export.ann export.dna

hmm-assembler.pl pyu . > ../pyu1.hmm

mv pyu_rnd1.all.gff ../

cd ..
```

After these steps, you will find two new files in the directory example_02_abinitio:

pyu_rnd1.all.gff: A gff file from round 1, which is evidence based genes.

pyu1.hmm: A hidden markov model trained from evidence based genes.

4. Run MAKER with the SNAP model.

Modify the maker_opts.ctl file. Open this file using a text editor. Modify the following values.

```
maker_gff= pyu_rnd1.all.gff

est_pass=1 # use est alignment from round 1
protein_pass=1   #use
rm_pass=1   # use repeats in the gff file

snaphmm=pyu1.hmm

est=  # remove est file, do not run EST blast again
protein=   # remove protein file, do not run blast again

model_org=  #remove repeat mask model, so not running RM again
rmlib=   # not running repeat masking again
repeat_protein= #not running repeat masking again

est2genome=0   # do not do EST evidence based gene model
protein2genome=0  # do not do protein based gene model.

pred_stats=1 #report AED stats
```

Run maker with the new control file. This time I use the "-base pyu_rnd2" so that you will not overwrite the results from first run.

```
screen

/usr/local/mpich/bin/mpiexec -n 2 maker -base pyu_rnd2 >& log2 &

## then press ctrl-"a" followed by pressing "d" to detach from screen
```

5. Let's retrain SNAP.

```
mkdir snap2
cd snap2
gff3_merge -d ../pyu_rnd2.maker.output/pyu_rnd2_master_datastore_index.log
maker2zff  -l 50 -x 0.5  pyu_rnd2.all.gff
fathom -categorize 1000 genome.ann genome.dna
fathom -export 1000 -plus uni.ann uni.dna
forge export.ann export.dna
hmm-assembler.pl pyu . > ../pyu2.hmm
mv pyu_rnd2.all.gff ..
cd ..
```

6. Modify the maker_opts.ctl file, and run maker again

Now, you will modify the maker_opts.ctl file for round 3. Open this file using a text editor. Modify the following values.

```
maker_gff=pyu_rnd2.all.gff

snaphmm=pyu2.hmm
```

Run maker with the new control file. This time I use the "-base pyu_rnd2" so that you will not overwrite the results from first run.

```
screen
/usr/local/mpich/bin/mpiexec -n 2 maker -base pyu_rnd3 >& log3 &
## then press ctrl-"a" followed by pressing "d" to detach from screen
```

Use the following command to create the final merged gff file. The "-n" option would produce a gff file without genome sequences.

```
gff3_merge -s -n -d pyu_rnd3.maker.output/pyu_rnd3_master_datastore_index.log>pyu_rnd3.noseq.gff
```

You will get a new gff3 file: pyu_rnd3.noseq.gff.

7. Generate AED plots.

```
/programs/maker/AED_cdf_generator.pl  -b 0.025  pyu_rnd2.all.gff > AED_rnd2

/programs/maker/AED_cdf_generator.pl  -b 0.025  pyu_rnd3. noseq.gff > AED_rnd3
```

You can plot the two files AED_rnd2 and AED_rnd3 in Excel or any plotting software.

8. You can load the gff file into IGV or JBrowse.

Instructions for IGV and JBrowse can be found at:

IGV: http://software.broadinstitute.org/software/igv/UserGuide

JBrowse: https://biohpc.cornell.edu/lab/userguide.aspx?a=software&i=357#c