Genome annotation

In this exercise, you will do genome annotation using Braker2 (https://github.com/Gaius-Augustus/BRAKER) and PASA (https://github.com/PASApipeline/PASApipeline/wiki), and will integrate the results with EVM (https://evidencemodeler.github.io/).

Some steps could take very long time to run (marked as **[DO IT AT HOME]** in the instructions). Please skip these steps for now. Use the result files we provide you to continue with. You can finish the "DO IT AT HOME" steps later during the week.

1. Prepare the working directory

1.1 Copy the data files to you working directory.

```
mkdir -p /workdir/$USER/project2

cd /workdir/$USER/project2

cp /shared_data/genomic_2020/project2/* ./
```

1.2 Examine the input files

You will use 4 of these files as input for this exercise. The rest are intermediate results from the pipeline.

- genome.fa: Genome assembly in fasta;
- r1.fastq.gz & r1.fastq.gz: Illumina RNA-seq reads from the same species;
- transcript.fasta.gz: Cleaned sequences of full length transcripts from the same species;

```
ls -l

# the assembly file
head genome.fa
grep ">" genome.fa

# the RNA-seq reads
zcat r1.fastq.gz |head -n 20
zcat r2.fastq.gz |head -n 20

# the transcript file
zcat transcript.fasta.gz | head -n 20
```

1.3 Install GeneMark Ex.

GeneMark-EX, one of the dependencies of Braker2, requires an academic license. You will need to download the software and the license files by yourself.

Go to the GeneMark web site: http://exon.gatech.edu/GeneMark/license_download.cgi. Check the boxes for "GeneMark-ES/ET/EP ver 4.61_lic" and "LINUX 64" next to it, fill out the form, then click "I agree". In the next page, right click and copy the two link addresses: "download program" and "64 bit" license, and paste the two link addresses in the commands below. (To be compatible with braker2 version, use the gmes_linux_64.tar.gz file we provided, which is compatible with the braker2 in our docker container. The file is located in /shared_data/genomic_2020/project2/)

```
cd /workdir/$USER/project2

#use the gmes_linux_64.tar.gz we provided, which is a version compatible with
braker2 version in our docker container.

wget "replace with license URL"

zcat gm_key_64.gz > .gm_key

tar xvfz gmes_linux_64.tar.gz

cd gmes_linux_64

perl change_path_in_perl_scripts.pl "/usr/bin/env perl"

cd ..
```

1.4 Pull Docker images for Braker2 and PASA from the Dockerhub

A Docker image file is a template file that can be used to start a Docker container, an enclosed virtual Linux instance. All BioHPC computers run CentOS Linux 7.5. But within a Docker container, you can run a different type of Linux, for example, Ubuntu.

The Docker image of biohpc/braker2 is built by the BioHPC team, it includes all software required by the Braker2 pipeline, except the GeneMark-EX software and license files. When running the Braker2 pipeline, you need to put the GeneMark software directory and license file in the same directory as the input files. Detailed information about this image can be found at https://biohpc.cornell.edu/lab/userguide.aspx?a=software&i=56#c .

The PASA image is provided by the software developer.

For security reasons, on the BioHPC system you need to run Docker through a wrapper "docker1", which is developed by Cornell Bioinformatics Facility. Detailed information about docker1 can be found at https://biohpc.cornell.edu/lab/userguide.aspx?a=software&i=340#c . In the following commands, you will download two Docker images from the Dockerhub.

```
docker1 pull biohpc/braker2

docker1 pull pasapipeline/pasapipeline

docker1 images
```

If someone else has already pulled the image on the same computer, you would see a message that this image exists.

2. Repeat Masking the genome

The repetitive regions of the genome need to be masked before doing *ab initio* annotation. It is recommended to do "soft" masking (converting repeat regions to lower case), instead of "hard" masking (converting repeats to "N"). As "Repeat masking" is time consuming, we provide you with pre-run results.

2.1 **[DO IT AT HOME]** Build a custom repeat database with RepeatModeler.

```
cd /workdir/$USER/project2

export PATH=/programs/RepeatModeler-2.0.1:$PATH

BuildDatabase -name mySpecies genome.fa

RepeatModeler -database mySpecies -pa 4 -LTRStruct >& repeatmodeler.log &
```

-pa 4: use 4 processors

-LTRStruct: including the LTR discovery module in the run.

After this step, you should see a directory named "RM_xxxxx.*date*" and two new files.

mySpecies-families.fa: Consensus repeat sequences in FASTA format.

mySpecies-families.stk: Multiple sequence alignment of the repeats in STOCKHOLM format.

The file *mySpecies-families.fa* is what you want. It can be used as database for RepeatMasker.

For plant genomes, an alternative software for building custom database is EDTA (https://github.com/oushujun/EDTA)

2.2 **[DO IT AT HOME]** Mask the genome with RepeatMasker

```
export PATH=/programs/RepeatMasker_4-1-0:$PATH

RepeatMasker -pa 4 -xsmall -lib mySpecies-families.fa -dir genome_mask genome.fa
```

The output file "genome_mask/genome.fa.masked" is the soft masked genome file.

2. Map RNA-seq reads to genome assembly

As the annotation software Braker2 cannot use RNA-seq reads directly, you will need to map the reads to the genome assembly first. You will use the RNA-seq read mapping software STAR. Run STAR with "--twopassMode Basic" to improve alignment to the splicing junctions.

2.1 **[DO IT AT HOME]** Create a STAR database from the assembly FASTA file

```
export PATH=/programs/STAR-2.7.5a/bin/Linux_x86_64_static:$PATH

STAR --runMode genomeGenerate --runThreadN 8 --genomeDir STARgenome --
genomeFastaFiles genome.fa
```

The output from this step is a directory named STARgenome which contains the STAR database.

2.2 **[DO IT AT HOME]** Map RNA-seq reads to the assembly

```
STAR --twopassMode Basic --genomeDir STARgenome --runThreadN 8 \
--readFilesIn r1.fastq.gz r2.fastq.gz --readFilesCommand zcat \
--outFileNamePrefix RNA_ --outSAMtype BAM Unsorted

samtools sort -T ./ -m 4G --threads 8 -o RNA.sorted.bam RNA_Aligned.out.bam

samtools index RNA.sorted.bam
```

After these steps, you should see a file RNA.sorted.bam. This file will be used as input for Braker2.

3. Run Braker2 for *ab initio* prediction

In this exercise, you will train the gene prediction model using Illumina RNA-seq data. Braker2 requires two input files: repeat masked genome file and RNA-seq BAM file.

3.1 Examine the two input files for Braker2

3.1.1 Repeatmasker output: genome_mask/genome.fa.masked

```
cd /workdir/$USER/project2

tar xvfz genome_mask.tar.gz

mv genome_mask/genome.fa.masked ./

less genome.fa.masked
```

3.1.2 START output: RNA.sorted.bam

```
samtools view -H RNA.sorted.bam | head
```

3.2 **[DO IT AT HOME]** Run Braker2.

You the run the command through Docker ("docker1" command on BioHPC). It will take hours to finish. Run it in screen session.

```
docker1 run --rm -v /workdir/$USER/project2:/data  biohpc/braker2 sh -c ".
/root/source.sh; braker.pl --species=mysp --genome=genome.fa.masked --
bam=RNA.sorted.bam --softmasking --gff3 --cores=20"
```

- Make sure that you have all four items directly under the data directory.   1) genome assembly file; 2)RNA-seq bam file; 3) Genemark directory "gmes_linux_64" ; 4) Genemark ".gm_key";

- On BioHPC, the data directory must be under "/workdir/$USER", as "docker1" cannot access files outside "/workdir/$USER".
- "-v /workdir/$USER/project2:/data" is to mount your data directory "/workdir/$USER/project2" to "/data" in the Docker container. (The directory name "/data" within Docker container is hard coded, so do not change);
- "braker.pl --species=mySpecies --genome=genome.fa.masked --bam=RNA.sorted.bam --softmasking --cores=8" is the Braker commands, modify the parameters as needed;
- "--rm": remove the docker container after the run finishes.

When you run a software in Docker, you run it as root. After the job is finished, you will need to get back the ownership of the output results. We have a special tool for this purpose:

```
docker1 claim
```

As docker images takes storage space on the server. To remove a docker image file that is not needed anymore, use the command "docker1 rmi imageName". (use "docker1 images" to find image names if you do not know the name.)

3.3 Examine the Braker2 output

```
tar xvfz braker_results.tar.gz

ls -l braker_results

less braker_results/augustus.hints.gff3

less braker_results/augustus.hints.gtf
```

Detailed information about the Braker2 output can be found at https://github.com/Gaius-Augustus/BRAKER#output-of-braker.

- "augustus.hints.gtf" and "augustus.hints.gff3": Augustus annotation results. These are the result files we want to use in the pipeline.
- "braker.gtf" and "braker.gff3": Union of augustus.hints.gtf and reliable GeneMark-EX predictions, which is supposed to be more sensitive but less specific. These two files have errors. I cannot use them in any downstream software. The same errors exist in the result files provided by the BRAKER developers. I have reported the errors on its github repository.

4. Run PASA for evidence based annotation

The PacBio Iso-seq is good platform to generate full length transcript sequence, which can be assembled and cleaned through its proprietary SMRT Link Iso-Seq application. Alternatively, the transcript.fasta file could be output from reference genome guided RNA-seq reads assembly software, e.g. StringTie2, cufflinks and Trinity.

The transcript sequence file is expected to be free of adapters and poly-A tail. Other software, e.g. SMRT Link Iso-Seq application should give you cleaned transcript sequences. You can also use software like cutadapt to trim poly-A tail and adapter sequences.

4.1 **[DO IT AT HOME]** Run PASA

4.1.1 To customize the output file names (otherwise the default prefix is "sample_mydb_pasa.sqlite"), you need to edit the provided template file sqlite.confs/alignAssembly.config, and change the line "DATABASE=/tmp/sample_mydb_pasa.sqlite" to "DATABASE=/tmp/mySpecies".

```
tar xvfz sqlite.confs.tar.gz

# edit the text in sqlite.confs/alignAssembly.config
# change the line  "DATABASE=/tmp/sample_mydb_pasa.sqlite" to
"DATABASE=/tmp/mySpecies"
```

4.1.2 Run PASA. As it is not aware of soft masking, you can use either the genome.fa or genome.fa.masked.

```
mkdir /workdir/$USER/tmp

cd /workdir/$USER/project2

zcat transcript.fasta.gz > transcript.fasta

docker1 run --rm -it \
      -v /workdir/$USER/tmp:/tmp \
      -v /workdir/$USER/project2/:/data  \
       pasapipeline/pasapipeline:latest \
        bash -c 'cd /data \
              && /usr/local/src/PASApipeline/Launch_PASA_pipeline.pl \
              -c sqlite.confs/alignAssembly.config -C -R \
              --ALIGNER gmap -g genome.fa -t transcript.fasta --CPU 4' >pasa.log
```

4.2 Examine PASA results

PASA produced many result files. The two most relevant annotation results are:

- mySpecies.pasa_assemblies.gff3
- mySpecies.pasa_assemblies.gtf

```
tar xvfz pasa_results.tar.gz

ls -l pasa_results

head pasa_results/mySpecies.pasa_assemblies.gff3

head pasa_results/mySpecies.pasa_assemblies.gtf
```

5. Run EVM (EVidenceModeler) for integration of annotations from Braker2 and PASA

Read the EVM (https://evidencemodeler.github.io/) documentation for details.

5.1 Validate the gff3 files from PASA and Braker2.

EVM provides a validation tool gff3_gene_prediction_file_validator.pl. If there are errors, you will get error messages. Otherwise, not output.

```
ln -s braker_results/augustus.hints.gff3 ./
ln -s pasa_results/mySpecies.pasa_assemblies.gff3 ./

export EVM_HOME=/programs/EVidenceModeler-1.1.1

$EVM_HOME/EvmUtils/gff3_gene_prediction_file_validator.pl
mySpecies.pasa_assemblies.gff3

$EVM_HOME/EvmUtils/gff3_gene_prediction_file_validator.pl augustus.hints.gff3
```

 * use the "ln -s" command to create symbolic links, so that we can use the files in the current directory.

5.2 Create a EVM weight file

EVM requires a weights file, which has three columns including the evidence class, type, and weight. The "class" parameter can be one of the following: ABINITIO_PREDICTION, PROTEIN, or TRANSCRIPT. The "type" corresponds to the second column (source) in the input gff3 file. The weights can be set intuitively (ie. weight(pasa) >> weight (protein) >= weight(prediction)). (See EVM manual for example weights for different data types).

A weights.txt file has be prepared for you. Examine its content.

```
cat weights.txt
```

5.3 **[DO IT AT HOME]** Run EVM

5.3.1 Partition the inputs

The genome sequences and gff3 files are partitioned based on individual contigs, and large contigs are segmented into smaller overlapping chunks. This would allow you to process the chunks in parallel.

```
$EVM_HOME/EvmUtils/partition_EVM_inputs.pl \
     --genome genome.fa \
     --gene_predictions augustus.hints.gff3  \
     --transcript_alignments mySpecies.pasa_assemblies.gff3 \
     --segmentSize 1000000 --overlapSize 200000 --partition_listing
partitions_list.out
```

5.3.2 Generating the EVM command set

EVM would create one command for each chunk you partitioned in the previous step. The commands are in the file commands.list.

```
$EVM_HOME/EvmUtils/write_EVM_commands.pl \
      --genome genome.fa --weights `pwd`/weights.txt \
      --gene_predictions augustus.hints.gff3 \
      --transcript_alignments mySpecies.pasa_assemblies.gff3 \
      --output_file_name evm.out  --partitions partitions_list.out >
commands.list
```

5.2.3 Execute the commands:

Here you will use the GNU parallel to run the commands.list in parallel.

```
parallel -j 4 < commands.list >& evm.log
```

5.3.4 Combining the partitions and convert to gff3 output

```
$EVM_HOME/EvmUtils/recombine_EVM_partial_outputs.pl --partitions
partitions_list.out --output_file_name evm.out

$EVM_HOME/EvmUtils/convert_EVM_outputs_to_GFF3.pl  --partitions
partitions_list.out --output evm.out  --genome genome.fasta

find . -regex ".*evm.out.gff3" -exec cat {} \; > EVM.all.gff3
```

After these steps, you get a final EVM.all.gff3 annotation file.

6. Examine the output of annotation pipeline

6.1 Get some basic statistics of the gff3 file

Using the Linux "uniq -c" command on 3rd column of the gff3 file.

```
cd /workdir/$USER/project2

head -n 100 EVM.all.gff3

cut -f3 EVM.all.gff3 | sort |uniq -c
```

6.2 Check the results using genome browser IGV

Download these files to you laptop using Filezilla:

- genome.fa
- RNA.sorted.bam.bai
- RNA.sorted.bam
- braker_results/augustus.hints.gff3
- EVM.all.gff3

IGV is a JAVA software that can be run on Windows, MAC or a Linux computer. To launch IGV on your laptop, go to IGV web site (https://software.broadinstitute.org/software/igv/), click “Download”, and download the Windows or Mac version for your laptop. Double click the IGV installation tool to install IGV. On Windows computer, the software is installed in the directory C:\Program Files\IGV_2.6.3. Double click “igv.bat” to start IGV. After double clicking, it might take a few seconds before you see the software starting.

You will need t create our own genome database. Click “Genomes”->”Create .genome” file. Fill out the following fields:

- Unique identifier: mySpecies
- Descript name: mySpecies
- Fasta: use the “Browse” button to find the genome.fa file
- Gene file: use the “Browse” button to find the EVM.all.gff3 file

Then save the genome database on your computer.

From menu “File” -> “Load file”, open the “RNA.sorted.bam” and "augustus.hints.gff3"

Navigate around different regions of the genome, comparing the annotations from Augustus and EVM with RNA-seq evidence.