Genome Annotation and Curation Using MAKER and MAKER-P

Michael S. Campbell,[1] Carson Holt,[1,2] Barry Moore,[1,2] and Mark Yandell[1,2]

[1]Eccles Institute of Human Genetics, University of Utah, Salt Lake City, Utah
[2]USTAR Center for Genetic Discovery, University of Utah, Salt Lake City, Utah

This unit describes how to use the genome annotation and curation tools MAKER and MAKER-P to annotate protein-coding and noncoding RNA genes in newly assembled genomes, update/combine legacy annotations in light of new evidence, add quality metrics to annotations from other pipelines, and map existing annotations to a new assembly. MAKER and MAKER-P can rapidly annotate genomes of any size, and scale to match available computational resources. © 2014 by John Wiley & Sons, Inc.

Keywords: genome annotation • comparative genomics • gene finding • plants

How to cite this article:
Campbell, M. S., Holt, C., Moore, B. and Yandell, M. 2014.
Genome Annotation and Curation Using MAKER and MAKER-P.
Curr. Protoc. Bioinform. 48:4.11.1-4.11.39.
doi: 10.1002/0471250953.bi0411s48

INTRODUCTION

In this unit, we describe the MAKER genome annotation and curation pipeline. All of the input files used in the following protocols are found in `CPB_MAKER.tar.gz`, available for download at *http://weatherby.genetics.utah.edu/CPB_MAKER/CPB_MAKER.tar.gz*. Also described is MAKER-P, a version of MAKER optimized for plant genome annotation efforts that offers a number of new functionalities such as ncRNA annotation capabilities and support for pseudogene identification (Zou et al., 2009; Campbell et al., 2014). Both MAKER and MAKER-P are available for download from *http://www.yandell-lab.org*. MAKER-P is also installed in the Texas Advanced Computing Center as part of the iPlant Cyberinfrastructure (Goff et al., 2011); see *https://pods.iplantcollaborative.org/wiki/display/sciplant/MAKER-P+at+iPlant* and *UNIT 1.22* in this manual.

MAKER and MAKER-P annotate and mask repetitive elements in the genome, and align protein and RNA evidence to the assembly, in a splice-aware fashion to accurately identify splice sites. They also run multiple ab initio gene predictors, compare all predicted gene models to RNA and protein alignment evidence, and then revise the ab initio gene models in light of this evidence. The best supported gene models are chosen using a quality metric called Annotation Edit Distance (AED), developed by the Sequence Ontology (Eilbeck et al., 2009). MAKER and MAKER-P's outputs include FASTA files (Lipman and Pearson, 1985; see *APPENDIX 1B* for description of FASTA format) of transcripts and proteins for each annotated gene, and GFF3 (Generic Feature Format version 3; see Internet Resources) files that describe the gene models and their supporting evidence. These GFF3 files also provide a number of quality metrics (including AED) for each gene model. This basic workflow is visually represented in Figure 4.11.1.

Though MAKER was originally developed for de novo annotation of emerging model organisms, it has expanded into a multiuse genome annotation and curation tool (Holt

Current Protocols in Bioinformatics 4.11.1-4.11.39, December 2014
Published online December 2014 in Wiley Online Library (wileyonlinelibrary.com).
doi: 10.1002/0471250953.bi0411s48

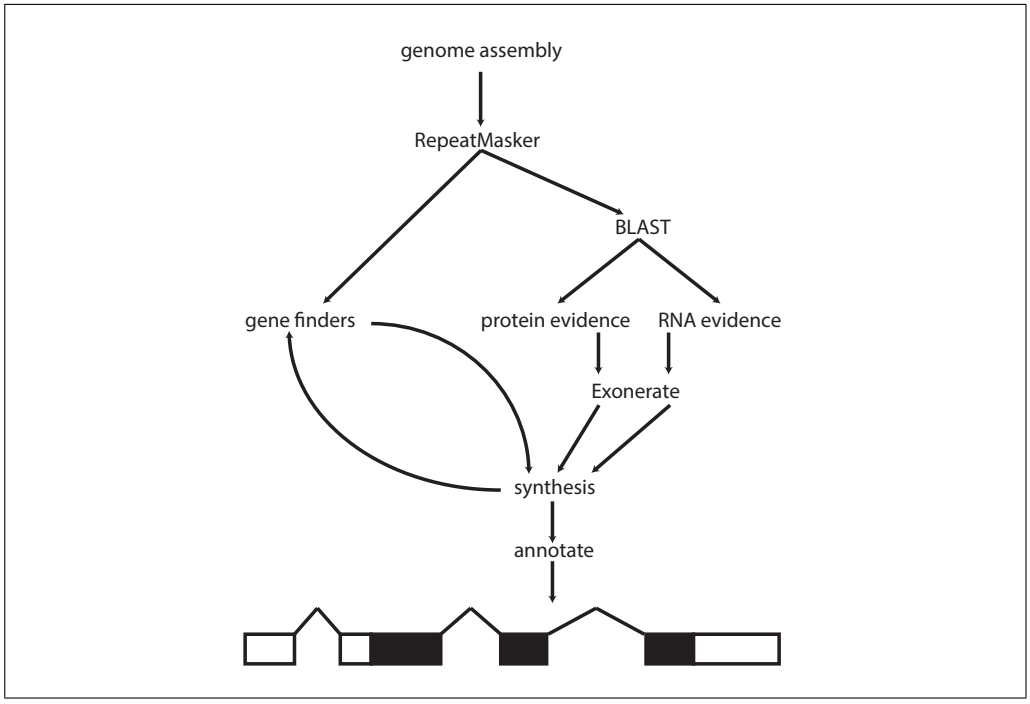

Figure 4.11.1 MAKER annotation workflow. MAKER masks repeats with RepeatMasker, aligns evidence to the genome with BLAST, polishes those alignments around splice sites using Exonerate, and runs a number of gene finders. MAKER also feeds evidence-based hints to the gene finders in order to improve their accuracy. These data are then synthesized into gene annotations.

and Yandell, 2011). In addition to de novo annotation, MAKER and MAKER-P can also be used to update existing annotations in the light of new experimental evidence and for quality control of gene models produced by other annotation pipelines (Campbell et al., 2013; Law et al., 2014)

MAKER and MAKER-P are both highly parallelized applications with support for the Message Passing Interface (MPI); this allows them to efficiently utilize multiple CPUs. Given enough CPUs, MAKER can annotate large mammalian and plant genomes in hours (Campbell et al., 2013). MAKER-P, which is available in massively parallel mode as part of the iPlant project (Goff et al., 2011), is even more powerful. For example, it was recently used to annotate the entire 22-GB loblolly pine genome assembly in less than 24 hr using over 8000 CPUs (Neale et al., 2014; Zimin et al., 2014). The highly parallelized architectures of MAKER and MAKER-P mean that users can experiment with alternate parameters and datasets to optimize annotation quality. It also makes it trivial to regularly update annotations as new evidence and assemblies become available.

STRATEGIC PLANNING

Know your organism

Knowledge of your organism's phylogenetic relationships and any previously annotated close relatives is crucial. The NCBI taxonomy browser can help identify closely related organisms and help find corresponding transcript and protein sequences to use as evidence while annotating your genome. UniProt/Swiss-Prot, NCBI genomes, and Ensembl are good places to look for protein data, while the sequence read archive (SRA), Genbank, and Ensembl are good places to look for RNA evidence.

Get the best assembly you can

MAKER has been used successfully on genomes derived from many different sequencing platforms and assemblers. For a comparison of assemblers, see the Assemblathon 2 paper

(Bradnam et al., 2013). As a rule of thumb, if the scaffold N50 of your assembly is less than the expected average gene length (including introns and UTR), the assembly should be improved before attempting to annotate it with MAKER (Yandell and Ence, 2012). You should also consider evaluating the "completeness" of the assembly using tools like CEGMA (Parra et al., 2007), which can indicate the upper limit of recoverable gene content from draft assemblies.

Sequence the genome with its eventual annotation in mind

Use a portion of your genome-sequencing budget to produce expression data. mRNA-seq data from multiple tissue types and stages of development helps greatly with gene annotation. Likewise, small RNA-seq data sets provide evidence to support ncRNA annotations.

DE NOVO GENOME ANNOTATION USING MAKER

BASIC PROTOCOL 1

Identifying the protein-coding genes in a newly assembled genome is a common first step in genome analysis. These protein-coding gene annotations enable further computational analyses and serve as the basis for diverse molecular biology experiments. Successful downstream analyses and experiments are contingent upon the quality of the underlying gene annotations.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules [automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed]. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

Files

Genome assembly to be annotated in FASTA format

Protein evidence in FASTA format

Assembled mRNA-seq transcripts from the species of interest in FASTA format

Optional: a species parameter/HMM file for SNAP generated for the organism of interest or a closely related species. The process used to create a species parameter/HMM file is described in SNAP's internal documentation (Korf, 2004).

1. From the Unix command line (using the "bash" shell), generate the MAKER control files (in the text below, lines that start with `%` show the command prompt; the `%` should not be typed; lines starting with # are comments and should not be typed):

```
% maker -CTL
```

This command generates three files: `maker_opts.ctl`, `maker_bopts.ctl`, *and* `maker_exe.ctl`. *User input is given to MAKER through these three files. For a detailed explanation of the options and parameters in the* `*.ctl`, *files, please see Critical Parameters and Advanced Parameters sections, below.*

2. Edit the `maker_opts.ctl` file to specify the genome assembly sequence, experimental alignment evidence, and which gene-finding method to use. Any text editor will work, but for purposes of this protocol we will use 'emacs':

```
% emacs maker_opts.ctl
#-----Genome (these are always required)
genome=$PATH_TO_CBP_maker_inputs/dpp_data/dpp_contig
  .fasta
#genome sequence (fasta file or fasta embeded in GFF3
  file)
organism_type=eukaryotic #eukaryotic or prokaryotic.
  Default is eukaryotic
#-----EST Evidence (for best results provide a file for
  at least one)
est=$PATH_TO_CBP_makerinputs/dpp_data/dpp_est.fasta
#set of ESTs or assembled mRNA-seq in fasta format
#-----Protein Homology Evidence (for best results
  provide a file for at least one)
protein=$PATH_TO_CBP_maker_inputs/dpp_data/dpp
  _protein.fasta
#protein sequence file in fasta format (i.e., from
  multiple organisms)
#-----Gene Prediction
snaphmm=$PATH_TO_CBP_maker_inputs/dpp_data/
  D.melanogaster.hmm #SNAP HMM file
```

Relative or absolute paths can be used in all of the `*.ctl` *files. To ensure proper parsing of these files, make sure that there are no spaces between the equal sign and the path to the files. With the exception of the* `genome=` *parameter, multiple files can be given to MAKER as a comma-separated list of paths. Protein evidence and mRNA-seq data are commonly given to MAKER in multiple files to better keep track of evidence sources in the final outputs (see Support Protocol 4).*

3. Run MAKER:

```
% maker 2> maker.error
```

The locations of the control files for a MAKER run can be specified on the command line. If they are not specified, the control files in the current working directory are used. As MAKER runs, it will output a number of progress messages to the screen along with any error messages (you can reduce the volume of messages by running MAKER with a `--q` *(quiet mode) to limit the status messages, or* `--qq` *(very quiet mode) to eliminate everything but errors). It is often helpful to save these status and error messages to a file for future reference, which is what was done on the above command line with the* `2>` *redirect).*

In addition to status and warning messages, MAKER creates an output directory named after the input genome FASTA file (if you would rather specify the name of the output directory you can do that on the command line by using the `-base` *option). In this example, the name of the output directory is* `dpp_contig.maker.output`*. After MAKER runs, you will find a number of additional files and directories inside this output directory. Of primary interest are the* `datastore` *directory and the* `datastore` *index log (both of which are named after the base name if given on the command line, or by default using the name of the genome FASTA file).*

Because genome annotation can produce hundreds of files for each of tens of thousands of contigs in the assembly, the MAKER datastore directory uses a hashed directory tree structure to separate the outputs for individual contigs/scaffolds from your assembly. Inside each contig directory you can find all of the genome annotation results that pertain to that

contig, together with a number of intermediate files that are saved to speed up subsequent MAKER runs. The datastore index log is the key to easily locating results in the datastore directory. It provides the final path to output for every annotated contig. The datastore index also indicates the run status of each contig processed (whether a contig has started, finished, failed, or was skipped).

4. Check the standard error output and datastore index file to see if MAKER is finished:

```
% tail --n 2 maker.error
maker is now finished!!!
% cat\
dpp_contig.maker.output/dpp_contig_master_datastore
  _index.log
contig-dpp-500-500
  dpp_contig_datastore/05/1F/contig-dpp-500-500/ STARTED
contig-dpp-500-500
  dpp_contig_datastore/05/1F/contig-dpp-500-500/
  FINISHED
```

   If everything went well, the last line of the `MAKER.error` *file will read* `MAKER is now finished!!!`*, and the datastore index log will have an entry for when MAKER started each entry in the genome FASTA file and when it finished or failed that entry. Since we have only one entry in our genome FASTA file, we have only two entries in our datastore index log. In this case, MAKER finished running and successfully finished our contig.*

5. Collect the results from all individual contigs into genome wide annotations:

```
% gff3_merge -d
dpp_contig.maker.output/dpp_contig_master_datastore
  _index.log
dpp_contig.all.gff
% fasta_merge -d
dpp_contig.maker.output/dpp_contig_master_datastore
  _index.log
dpp_contig.all.maker.proteins.fasta
dpp_contig.all.maker.transcripts.fasta
dpp_contig.all.maker.snap_masked.proteins.fasta
dpp_contig.all.maker.snap_masked.transcripts.fasta
dpp_contig.all.maker.non_overlapping_ab_initio
  .proteins.fasta
dpp_contig.all.maker.non_overlapping_ab_initio
  .transcripts.fasta
```

   MAKER uses two output formats, GFF3 and FASTA. Gene predictions, evidence alignments, repetitive elements, and the final gene models are output in GFF3 format, while transcript and protein sequences are output in FASTA format. Here we used two of the accessory scripts distributed with MAKER to collect the GFF3 and FASTA results from individual contigs and merge them to provide genome-wide results. These scripts use the directory paths present in the datastore index log to find the relevant files for each contig.

   After merging, you will have a single GFF3 file, together with protein and transcript sequences of the MAKER annotations. Depending upon runtime parameters, MAKER's outputs may also include additional FASTA files for the ab initio gene predictions, and/or rejected gene predictions with no evidence support that do not overlap a MAKER annotated gene. These additional files are given to the user for reference and evaluation purposes, and their presence depends on the user defined setting in the `MAKER_opts.ctl` *file. For example, if the* `keep_preds` *parameter in the* `maker_opts.ctl` *file is set to* `1`*, there will not be FASTA output for non-overlapping ab initio predictions because they will all be contained in the* `maker-transcripts` *and* `maker-protein` *files. These files may*

also be absent if every locus with a gene prediction was supported by evidence and thus annotated by MAKER.

Once the GFF3 and FASTA files are merged together, the structural protein-coding gene annotation is complete. Subsequent protocols document post-processing options and functional annotation protocols available to MAKER/MAKER-P users.

De novo genome annotation using MAKER-P

Building on MAKER, MAKER-P adds noncoding RNA and pseudogene annotation functionality as well as protocols for generating species specific repeat libraries. Mi-RPREFer was developed as part of the MAKER-P tool kit to annotate miRNAs, and can be found at *https://github.com/hangelwen/miR-PREFeR*. tRNAscan-SE (Lowe and Eddy, 1997) and snoscan (Lowe, 1999) are also integrated into the MAKER-P framework, and are run by using `trna=` and `snoscan_rrna=` in the `maker_opts.ctl` file (see `trna=` and `snoscan_rrna=` in Table 4.11.2 at the end of this unit). Pseudogenes are annotated using the method described here (Zou et al., 2009; Campbell et al., 2014). A protocol for annotating pseudogenes can be found at *http://shiulab.plantbiology.msu.edu/wiki/index.php/Protocol:Pseudogene*. See Campbell et al. (2014) for benchmarking results for MAKER-P annotated ncRNAs and pseudogenes on the Arabidopsis genome.

Adequate repeat masking is critical for accurate gene annotations. Basic and advanced protocols for generating species-specific repeat libraries can be found at *http://weatherby.genetics.utah.edu/MAKER/wiki/index.php/Repeat_Library_Construction--Basic* and *http://weatherby.genetics.utah.edu/MAKER/wiki/index.php/Repeat_Library_Construction--Advanced*. See Campbell et al. (2014) for benchmarking of these repeat library generation protocols on the Arabidopsis genome. MAKER-P is available for use on the iPlant infrastructure; see *https://pods.iplantcollaborative.org/wiki/display/sciplant/MAKER-P+at+iPlant* for MAKER-P usage on iPlant as an atmosphere image and on the Texas Advanced Computing Center (TACC) compute clusters.

ALTERNATE PROTOCOL 1

DE NOVO GENOME ANNOTATION USING PRE-EXISTING EVIDENCE ALIGNMENTS AND GENE PREDICTIONS

Aligning evidence to a genome assembly is one of the more time consuming and computationally expensive steps in genome annotation. Using pre-aligned evidence will substantially decrease the time it takes to annotate a genome. MAKER can take protein and mRNA-seq/EST alignments as evidence as well as aligned repetitive elements for masking. It can also accept existing gene predictions. All of these data need to be in GFF3 format.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules [automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed]. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This

example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

Files

Genome assembly to be annotated in FASTA format, protein evidence alignments in GFF3 format, assembled mRNA-seq transcript alignments from the species of interest in GFF3 format, gene predictions for the genomic assembly you wish to annotate in GFF3 format, and repetitive elements to be masked in GFF3 format.

1. From the Unix command line, generate the MAKER control files:

```
% maker -CTL
```

This is the same as in Basic Protocol 1.

2. Edit the `maker_opts.ctl` file to add the genomic sequence and evidence, and specify the gene=finding method:

```
% emacs maker_opts.ctl
#-----Genome (these are always required)
genome=$PATH_TO_CBP_maker/maker_inputs/dpp_data/dpp
  _contig.fasta #genome sequence (fasta file or fasta
  embeded in GFF3 file)
organism_type=eukaryotic #eukaryotic or prokaryotic.
  Default is eukaryotic
#-----EST Evidence (for best results provide a file for
  at least one)
est_gff=$PATH_TO_CBP_maker/maker_inputs/dpp_data/
  mRNA_seq_evidence.gff
#aligned ESTs or mRNA-seq from an external GFF3 file
#-----Protein Homology Evidence (for best results
  provide a file for at least one)
protein_gff=$PATH_TO_CBP_maker/maker_inputs/dpp
  _data/protein_evidence.gff
#aligned protein homology evidence from an external GFF3
  file
#-----Repeat Masking (leave values blank to skip repeat
  masking)
rm_gff=$PATH_TO_CBP_maker/maker_inputs/dpp
  _data/repeats.gff
#pre-identified repeat elements from an external GFF3
  file
#-----Gene Prediction
pred_gff=$PATH_TO_CBP_maker/maker_inputs/dpp_data/
  snap_predictions.gff #ab-initio predictions from an
  external GFF3 file
```

MAKER is expecting alignments in the GFF3 file to be represented as `match/match_part` two-level features. Below is an example from the `mRNA_seq_evidence.gff` file. Importantly, MAKER assumes that evidence passed in as GFF3 represents the correct exon boundaries of transcripts; for best results, make sure that precomputed BLAST alignments have been aligned to the genome in a splice-aware fashion before passing them to MAKER in GFF3 format:

```
contig-dpp-500-500 est2genome expressed_sequence_match
  26786
```

```
  31656 14993 +.
  ID=contig-dpp-500-500:hit:53:3.2.0.0;Name=dpp-mRNA-5
contig-dpp-500-500 est2genome match_part 26786 26955
  14993 +. ID=contig-dpp-500-500:
  hsp:62:3.2.0.0;Parent=contig-dpp-500-
  500:hit:53:3.2.0.0;Target=dpp-mRNA-5 1 170
  +;Gap=M170
contig-dpp-500-500 est2genome match_part 27104 27985
  14993 +. ID=contig-dpp-500-500:
  hsp:63:3.2.0.0;Parent=contig-dpp-500-
  500:hit:53:3.2.0.0;Target=dpp-mRNA-5 171 1052
  +;Gap=M882
contig-dpp-500-500 est2genome match_part 29709 31656
  14993 +. ID=contig-dpp-500-500:hsp:
  64:3.2.0.0;Parent=contig-dpp-500-
  500:hit:53:3.2.0.0;Target=dpp-mRNA-5 1053 3000
  +;Gap=M1948
```

3. Run MAKER and check/collect the results as outlined in Basic Protocol 1, steps 3 to 5.

ALTERNATE PROTOCOL 2

PARALLELIZED DE NOVO GENOME ANNOTATION USING MPI

Users can dramatically decrease the time required for annotating a genome by spreading the computation out across multiple compute cores (CPUs). MAKER is fully MPI compliant, allowing users to parallelize their genome annotation efforts.

Necessary Resources

Hardware

Multicore server or cluster with a Linux-based operating system

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules [automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed]. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

OpenMPI or MPICH2

Files

Genome assembly to be annotated in FASTA format, protein evidence alignments in GFF3 format, assembled mRNA-seq transcript alignments from the species of interest in GFF3 format, gene predictions for the genomic assembly you wish to annotate in GFF3 format, and repetitive elements to be masked in GFF3 format.

1. Configure MAKER to run with MPI during the installation step of MAKER:

```
% cd $PATH_TO_MAKER/maker/src
% perl Build.PL
MAKER supports distributed parallelization via MPI.
  Would you like to configure MAKER for MPI (This
```

```
  requires that you have an MPI client installed)? [N]Y
Please specify the path to 'mpicc' on your system:
  [/usr/local/mpich2/bin/mpicc ]
Please specify the path to the directory containing
  'mpi.h': [/usr/local/mpich2/include ]
```

The text below the command lines is generated by MAKER and requires user input. The default input is printed in the brackets and can be accepted by pressing return/enter or changed by entering the requested information and pressing return/enter. These steps can be done when you install MAKER. In the above example, MAKER found `mpicc` and `mpi.h` in the path and gave them as the default response to the specify path request. If you would like to use another version/flavor of MPI, you can specify it at this point. In this example we are using MPICH2.

When installing MPICH2 or OpenMPI, it is important to compile them with shared libraries enabled. For OpenMPI, this may require the addition of a line similar to the one below to your `~/.bash_profile` or equivalent.

```
export LD_PRELOAD=/path/to/openmpi/lib/libmpi.so:$LD
  _PRELOAD
```

OpenMPI and MPICH2 exhibit very similar performance on jobs using less than 100 CPUs. When using more than 100 CPUs, the OpenMPI implementation of MPI is more stable.

2. Generate the MAKER control files and edit `maker_opts.ctl` as outlined in Basic Protocol 1, steps 1 and 2.

3. Run MAKER using mpiexec on the number of CPU cores you wish to utilize:

```
% mpiexec -n 26 maker
```

The first part of this command, `mpiexec`, is a standard way of starting an MPI job regardless of the MPI implementation. The `--n` argument to `mpiexec` is used to specify the number of processors (in this case 26). The next command is the `maker` executable. Please note that according to `mpiexec` documentation, in order to run this same command in the background or under control of `nohup`, you must also attach `/dev/null` to STDIN as demonstrated below:

```
% nohup mpiexec -n 26 maker < /dev/null &
```

Different cluster environments may also require additional command-line arguments for `mpiexec`; check with your cluster administrator and/or MPICH2 and OpenMPI documentation for additional details. For example, disabling OpenFabrics support may be required on Infiniband-based clusters for MAKER to work correctly with OpenMPI:

```
% mpiexec -mca btl ^openib -n 26 maker
```

4. Check/collect the results as outlined in Basic Protocol 1, steps 4 to 5.

PARALLELIZED DE NOVO GENOME ANNOTATION WITHOUT MPI

ALTERNATE PROTOCOL 3

If it is not possible to install MPICH2 or OpenMPI on the server or cluster where you wish to run MAKER, there is still a way to annotate your genome in parallel. This is done by starting multiple MAKER instances in the same directory. Each instance of MAKER will then use file locks together with the datastore index log to coordinate contig processing across multiple MAKER instances. If a datastore index log entry indicates that a contig

is being processed by a separate instance of MAKER, then that instance of MAKER will skip to the next contig in the FASTA. This checking and skipping process will continue until a given instance of MAKER finds an entry that has not been started.

Necessary Resources

Hardware

Multicore server or cluster with a Linux based operating system.

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules [automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed]. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

1. Generate the MAKER control files and edit the `maker_opts.ctl` as outlined in Basic Protocol, steps 1 and 2.
2. Start multiple instances of MAKER in the same directory (started as background processes):

```
% maker 2> maker1.error
% maker 2> maker2.error &
% maker 2> maker3.error &
```

3. Check/collect the results as outlined in Basic Protocol 1, steps 4 to 5.

SUPPORT PROTOCOL 1

TRAINING GENE FINDERS FOR USE WITH MAKER

Ab initio gene finders can achieve very high accuracies when well trained. Training data normally takes the form of a 'gold-standard' set of pre-existing gene annotations. Unfortunately, training data is usually not available for a newly sequenced organism. Here we outline a method for generating training data for a novel, never before annotated genome. The key is using MAKER in an iterative fashion. For more on this topic, see Holt and Yandell (2011). In this example, we train SNAP, but this method can be applied to other gene finders as well.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules [automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed]. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

Files

This example uses a larger data set than Basic Protocol 1 so as to generate enough gene models to train the gene finder. The file types are the same, with the exception of the SNAP species parameter/HMM file, which we are going to create. Here we are using a data set from *Pythium ultimum* (Lévesque et al., 2010). Note that the protein and EST evidence could also be given in GFF3 format (see Alternate Protocol 1).

1. Create MAKER control files as outlined in Basic Protocol 1, step 1.

2. Edit the `maker_opts.ctl` file to add the genomic sequence and evidence, and specify the gene finding method:

```
% emacs maker_opts.ctl
#-----Genome (these are always required)
genome=$PATH_TO_CBP_maker/maker_inputs/pyu_data/pyu-
  contig.fasta #genome sequence (fastafile or fasta
  embeded in GFF3 file)
organism_type=eukaryotic #eukaryotic or prokaryotic.
  Default is eukaryotic
#-----EST Evidence (for best results provide a file
  for at least one)
est=$PATH_TO_CBP_maker/maker_inputs/pyu_data/pyu-
  est.fasta
#set of ESTs or assembled mRNA-seq in fasta format
#-----Protein Homology Evidence (for best results
  provide a file for at least one)
protein=$PATH_TO_CBP_maker/maker_inputs/pyu_data/pyu-
  protein.fasta #protein sequence file in fasta format
  (i.e., from multiple organisms)
#-----Gene Prediction
est2genome=1
#infer gene predictions directly from ESTs, 1 = yes, 0
  = no
```

Note that the configuration shown above differs from that in Basic Protocol 1 in the Gene Prediction section. By setting `est2genome=1`*, MAKER will infer gene models directly from the EST/mRNA-seq evidence. Remember that if these data are given in GFF3 format, they must have been aligned to the genome in a splice-aware fashion. BLAST data will not suffice. If given in FASTA format, as in this example, MAKER will take care of the aligning and polishing. Given the nature of these data, many of the resulting gene models will be partial. However, there is usually enough information in these gene models for first-round training of a gene finder. Alternatively, you could also set* `protein2genome=1` *to derive gene models from splice-aware aligned protein evidence.*

3. Run MAKER with or without MPI (see Basic Protocol 1 and Alternate Protocols 2 and 3).

4. Check/collect the results as outlined in Basic Protocol 1, steps 4 to 5.

   For gene finder training, you only need to collect the GFF3 file for the genome.

5. Make a directory for SNAP training and go to it:

```
% mkdir snap1
% cd snap1
```

6. Run `maker2zff`:

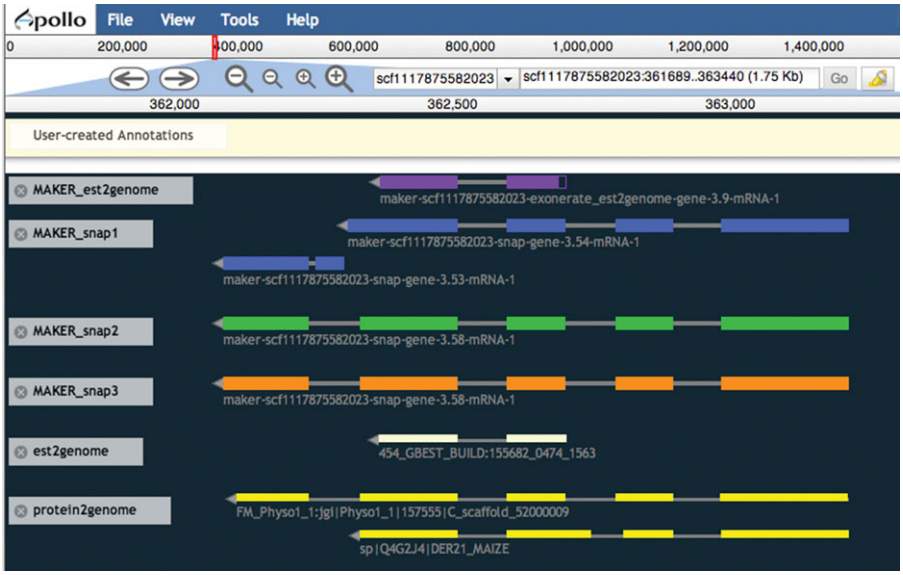

Figure 4.11.2 Iterative gene finder training improves gene annotations. Using only EST data and no gene finders, MAKER annotates a single two-exon gene at the locus `MAKER_est2genome` (purple). This annotation is consistent with the EST alignment (`est2genome`, beige), but is inconsistent with protein evidence data (`protein2genome`, yellow). After one round of SNAP training, MAKER annotates two models at this locus (`MAKER_snap1`, blue); these two models are more consistent with the protein evidence (`protein2genome`, yellow). An additional round of training yields a single MAKER annotation (`MAKER_snap2`, green) that is still more consistent with the protein evidence. Note that SNAP is not improved on with further training (`MAKER_snap3`, orange).

```
% maker2zff ../ pyu-contig.all.gff
genome.ann
genome.dna
```

`maker2zff` is an accessory script that comes with MAKER. It generates a ZFF-formatted file (`genome.ann`) and a FASTA file (`genome.dna`) that are required to train SNAP. To produce these files, the input GFF3 file must contain the genomic FASTA sequence appended to the end according to the GFF3 specification (this is the default used by `gff3_merge`). In order for a gene model to be considered suitable for training, it has to pass several quality filters imposed by the `maker2zff script`. By default, a gene model must have half of its splice sites confirmed by an EST/mRNA-seq alignment; half of its exons must overlap an EST/mRNA-seq alignment; and its annotation edit distance must be less than 0.5. All of these criteria can be modified on the command line.

7. Run `fathom with the categorize option` (part of SNAP package):

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

8. Run `fathom with the export option`:

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

9. 9 Run `forge` (part of SNAP package):

```
% forge export.ann export.dna
```

10. Run `hmm-assembler.pl` (part of SNAP package) to generate the final SNAP species parameter/HMM file and return to the MAKER working directory:

```
% hmm-assembler.pl pyu1 . > pyu1.hmm
% cd ..
```

11. Edit the maker_opts.ctl file to use the newly trained gene finder:

```
%emacs maker_opts.ctl
#-----Gene Prediction
snaphmm=./snap1/pyu1.hmm #SNAP HMM file
est2genome=0
#infer gene predictions directly from ESTs, 1 = yes, 0
  = no
```

12. Optional bootstrap training can be done by now repeating steps 3 to 10 and using the initial SNAP HMM file to seed the next round of SNAP training.

    Generally there is little further improvement after two rounds of bootstrap training with the same evidence, and you run the risk of overtraining (which can actually decrease SNAP's accuracy). See Figure 4.11.2. Once SNAP is trained, you can use the SNAP-derived annotations to train other gene finders following this same bootstrap procedure. When all of your gene finders are trained, you are ready to annotate your genome using Basic Protocol 1 or any of the alternate protocols above.

RENAMING GENES FOR GENBANK SUBMISSION

SUPPORT PROTOCOL 2

You can learn a lot about a MAKER gene annotation from the name assigned to the gene. Take for example the gene named `maker-contig-dpp-500-500-snap-gene-0.3`. Since it starts with `maker`, we know that that it is derived from a MAKER 'hint-based' prediction (for more information about how MAKER passes evidence derived 'hints' to the gene predictors, see Cantarel et al., 2008; Holt and Yandell, 2011). We have the name of the scaffold that the gene is on (`contig-dpp-500-500`) followed by the name of the gene finder used to generate the original and hint-based model (`snap`). The numbers following the gene predictor are used to make the ID unique.

Though useful, these IDs are not intended to be permanent. Once you have a registered genome prefix, you can use two of the accessory scripts distributed with MAKER to replace your MAKER gene names with NCBI-style gene IDs.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

`maker_map_ids, map_gff_ids, and map_fasta_ids` distributed with MAKER.

Files

MAKER generated GFF3 and FASTA files

1. Generate an id mapping file using `maker_map_ids`:

```
% maker_map_ids --prefix DMEL_ --justify 6\
dpp_contig.all.gff > dpp_contig.all.map
```

   This creates a two-column tab-delimited file with the original id in column 1 and the new id in column 2. The `--prefix` is where you give your registered genome prefix; the value following `--justify` determines the length of the number following the prefix (make sure that you allow adequate places for the number of genes in the annotation set, e.g., if you have 10,000 genes, `--justify` should be set to at least 5).

2. Look at the contents of the `contig-dpp-500-500.map` file.

```
% cat dpp_contig.all.map
maker-contig-dpp-500-500-snap-gene-0.3 DMEL_000001
maker-contig-dpp-500-500-snap-gene-0.3-mRNA-1
  DMEL_000001-RA
```

You will notice that the `.map` *files are simply two-column files showing the conversion of the existing gene/transcript ID (column 1) to a new ID (column 2).*

3. Use the map file created in step 1 to change the ids in the GFF3 and FASTA file

```
% map_gff_ids dpp_contig.all.map dpp_contig.all.gff
% map_fasta_ids dpp_contig.all.map
  dpp_contig.all.maker.proteins.fasta
% map_fasta_ids dpp_contig.all.map
  dpp_contig.all.maker.transcripts.fasta
% head --n 3 dpp_contig.all.gff
contig-dpp-500-500 . contig 1 32156 . . .
  ID=contig-dpp-500-500;Name=contig-dpp-500-500
contig-dpp-500-500 maker gene 23054 31656 . + .
  ID=DMEL_000001;Name=DMEL_000001;Alias=maker-contig-
  dpp-500-500-snap-gene-0.3;
```

Note that the above command lines do not redirect standard out (STDOUT). These scripts do an in-place edit of the file to save disk space. Therefore, it is important not to interrupt these scripts as they run, or the files can be corrupted/truncated. In this example, our long MAKER-generated gene ID `maker-contig-dpp-500-500-snap-gene-0.3` *was changed to* `DMEL_000001` *in both the GFF3 and FASTA files, with the original MAKER name kept as an alias.*

SUPPORT PROTOCOL 3

ASSIGNING PUTATIVE GENE FUNCTION

MAKER also provides support for functional annotation (i.e., identifying putative gene functions, protein domains, etc.). This protocol uses NCBI BLAST+ and the well-curated UniProt/Swiss-Prot set of proteins to assign putative functions to newly annotated genes.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

`NCBI BLAST+, maker_functional_gff, and maker_functional_fasta` (from MAKER)

Files

UniProt/SwissProt multi-FASTA file (*http://www.uniprot.org*), MAKER-generated GFF3 and FASTA files

1. Index the UniProt/Swiss-Prot multi-FASTA file using `makeblastdb`:

```
% makeblastdb -in uniprot_sprot.fasta -input_type fasta
  -dbtype prot
```

2. BLAST the MAKER-generated protein FASTA file to UniProt/SwissProt with BLASTP. Some command lines are longer than a single printed (displayed) line. These long commands include a '\' before the continued line, so that multiple lines are read as a single line:

```
% blastp -db uniprot_sprot.fasta\
-query contig-dpp-500-500.maker.proteins.fasta -out
  maker2uni.blastp -evalue\ .000001 -outfmt 6
  -num_alignments 1 -seg yes -soft_masking true\
-lcase_masking -max_hsps_per_subject 1
```

The key parts of this BLAST command line include the specification of the tabular format (`-outfmt 6`), and the `-num_alignments 1` and `-max_hsps_per_subject 1` flags which limit the hits returned for a given sequence to a single line in the BLAST report. The output for this BLAST search is:

```
DMEL_000001-RA sp|P07713|DECA_DROME 100.00 588 0 0 1 588
  1 588 0.0 1220
```

Tabular-formatted WUBLAST/ABBLAST output works as well for this protocol.

3. Add the protein homology data to the MAKER GFF3 and FASTA files with `maker_functional_gff and maker_functional_fasta`.

```
% maker_functional_gff uniprot_sprot.fasta\
maker2uni.blastp dpp_contig.all.gff\
contig-dpp-500-500_functional_blast.gff
% maker_functional_fasta uniprot_sprot.fasta\
  maker2uni.blastp\ dpp_contig.all.maker.proteins.fasta\
dpp_contig.all.maker.proteins_functional_blast.fasta
% maker_functional_fasta uniprot_sprot.fasta\
  maker2uni.blastp\
dpp_contig.all.maker.transcripts.fasta\
dpp_contig.all.maker.transcripts_functional_blast
  .fasta
```

This procedure added `Note=Similar to dpp: Protein decapentaplegic (Drosophila melanogaster);` to column 9 of the gene and mRNA feature lines in the MAKER GFF3 file:

```
% head -n 4 dpp_contig.all.functional_blast.gff
##gff-version 3
contig-dpp-500-500 . contig 1 32156 . . .
  ID=contig-dpp-500-500;Name=contig-dpp-500-500
contig-dpp-500-500 maker gene 23054 31656 . + .
  ID=DMEL_000001;Name=DMEL_000001;Alias=maker-contig-
  dpp-500-500-snap-gene-0.3;Note=Similar to dpp:
  Protein decapentaplegic (Drosophila melanogaster);
contig-dpp-500-500 maker mRNA 23054 31656 . + .
  ID=DMEL_000001-
  RA;Parent=DMEL_000001;Name=DMEL_000001-
  RA;Alias=maker-contig-dpp-500-500-snap-gene-0.3-mRNA-
  1;_AED=0.13;_QI=1422|1|1|1|0.5|0.33|3|1049|588;
  _eAED=0.13;Note=Similar to dpp: Protein
  decapentaplegic (Drosophila melanogaster);
```

This also added “Name:” Similar to dpp Protein decapentaplegic (Drosophila melanogaster)“” to the definition lines of the FASTA entries.

```
% head -n 1
  dpp_contig.all.maker.proteins_functional_blast.fasta
```

```
DMEL_000001-RA protein Name:"Similar to dpp Protein
  decapentaplegic (Drosophila melanogaster)" AED:0.13
  eAED:0.13 QI:1422|1|1|1|0.5|0.33|3|1049|588
% head -n 1
  dpp_contig.all.maker.transcripts_functional_blast
  .fasta
DMEL_000001-RA transcript Name:"Similar to dpp Protein
  decapentaplegic (Drosophila melanogaster)" offset:1422
  AED:0.13 eAED:0.13 QI:1422|1|1|1|0.5|0.33|3|1049|588
```

A similar tool called `ipr_update_gff` *is also distributed with MAKER. This tool allows users to add functional annotations from InterProScan (Quevillon et al., 2005), including protein family domains to the MAKER GFF3 file. See Basic Protocol 5 for more on this point.*

SUPPORT PROTOCOL 4

LABELING EVIDENCE SOURCES FOR DISPLAY IN GENOME BROWSERS

Many genome annotation projects entail the use of multiple RNA-seq and protein datasets. For example, the RNA-seq datasets might come from multiple tissue types, stages of life, strains, accessions, and treatments, and the protein datasets might comprise the proteomes of related species. All of these data can be passed to MAKER as evidence in the form of a comma-separated list added to the `maker_opts.ctl` file. Additionally, each file can be given a tag that is moved forward to the MAKER GFF3 output to identify the source of any given evidence alignment. This tag can be very helpful when visualizing your data in a genome browser or when mining data from the MAKER-generated GFF3 file to use in other applications/protocols.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules (automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

Files

Genome assembly to be annotated in FASTA format

Protein evidence in FASTA format

Assembled mRNA-seq transcripts from the species of interest in FASTA format

Optional: a species parameter/HMM file for SNAP generated for the organism of interest or a closely related species. The process used to create a species parameter/HMM file is described in SNAP's internal documentation (Korf, 2004).

1. Create MAKER control files as outlined in Basic Protocol 1, step 1.
2. Edit the `maker_opts.ctl` file to add the genomic sequence and evidence, and specify the gene-finding method. Add the tags to the evidence at this time:

```
% emacs maker_opts.ctl
#-----Genome (these are always required)
genome=$PATH_TO_CBP_maker/maker_inputs/dpp_data/dpp
  _contig.fasta #genome sequence (fasta file or fasta
  embeded in GFF3 file)
organism_type=eukaryotic #eukaryotic or prokaryotic.
  Default is eukaryotic
#-----EST Evidence (for best results provide a file for
  at least one)
est=$PATH_TO_CBP_maker/maker_inputs/dpp_data/dpp_est
  .fasta:3instar
#set of ESTs or assembled mRNA-seq in fasta format
#-----Protein Homology Evidence (for best results
  provide a file for at least one)
protein=$PATH_TO_CBP_maker/maker_inputs/dpp_data/dpp
  _protein.fasta:Dsim #protein sequence file in fasta
  format (i.e., from multiple organisms)
#-----Gene Prediction
snaphmm=$PATH_TO_CBP_maker/maker_inputs/dpp_data/
  D.melanogaster.hmm #SNAP HMM file
```

3. Run MAKER and check/collect the results as outlined in Basic Protocol 1, steps 3 to 5.

The tags are added after the evidence dataset file name as a suffix consisting of a colon, followed by the identification tag. In the above example, the tag `3instar` was added to the `est` file and the tag `Dsim` was added to the protein file. In the final GFF3 output, the source column (column 2 in bold below) for the BLASTN alignments from the `dpp_est.fasta` file is changed from `blastn` to `blastn:3instar`. Similarly, as sources the GFF3 file contains `est2genome:3instar`, `blastx:Dsim`, and `protein2genome:Dsim`.

```
% grep blastn dpp_contig.all.gff | head -n 1
contig-dpp-500-500 blastn:3instar
  expressed_sequence_match 26786 31656 170 + .
  ID=contig-dpp-500-500:hit:48:3.2.0.0;Name=dpp-mRNA-5
% grep est2genome dpp_contig.all.gff | head -n 1
contig-dpp-500-500 est2genome:3instar
  expressed_sequence_match 26786 31656 14993 + .
  ID=contig-dpp-500-500:hit:53:3.2.0.0;Name=dpp-mRNA-5
% grep blastx dpp_contig.all.gff | head -n 1
contig-dpp-500-500 blastx:Dsim protein_match 27118
  30604 1482 + .
  ID=contig-dpp-500-500:hit:58:3.10.0.0;Name=dpp-CDS-5
% grep protein2genome dpp_contig.all.gff | head -n 1
contig-dpp-500-500 protein2genome:Dsim protein_match
  27118 30604 3062 + .
  ID=contig-dpp-500-500:hit:63:3.10.0.0;Name=dpp-CDS-5
```

This feature simplifies loading different lines of evidence into a genome browser as separate tracks.

UPDATING/COMBINING LEGACY ANNOTATION DATASETS IN LIGHT OF NEW EVIDENCE

MAKER provides means to employ new evidence to improve the accuracy of existing genome annotations without completely reannotating the genome. This allows MAKER users to rapidly update existing annotations in light of new mRNA-seq data sets and protein evidence. Note that the starting annotations need not have been produced using MAKER. The protocol outlined below assumes that a starting dataset of annotations is available in GFF3 format. If this is not available, see Basic Protocol 4, "Mapping annotations to a new assembly," which explains how to map pre-existing transcripts (produced by any annotation pipeline) to a genome assembly and produce a GFF3 file for later use with MAKER.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules (automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

Files

Genome assembly from the original annotation in FASTA format, new protein evidence in FASTA format, new assembled mRNA-seq transcripts from the species of interest in FASTA format, annotations to be updated/combined in GFF3 format

1. Create MAKER control files as outlined in Basic Protocol 1, step 1.
2. Edit the `maker_opts.ctl` file to add the genomic sequence, evidence, and gene models you wish to update:

```
% emacs maker_opts.ctl
#-----Genome (these are always required)
genome=$PATH_TO_CBP_maker/maker_inputs/legacy_data/
  legacy-contig.fasta #genome sequence (fasta file or
  fasta embeded in GFF3 file)
organism_type=eukaryotic #eukaryotic or prokaryotic.
  Default is eukaryotic
#-----EST Evidence (for best results provide a file for
  at least one)
est=$PATH_TO_CBP_maker/maker_inputs/legacy_data/
  legecy-new-mRNAseq.fasta #set of ESTs or assembled
  mRNA-seq in fasta format
#-----Protein Homology Evidence (for best results
  provide a file for at least one)
```

```
protein=$PATH_TO_CBP_maker/maker_inputs/legacy_data/
  legacy-new-protein.fasta #protein sequence file in
  fasta format (i.e., from multiple organisms)
#-----Gene Prediction
pred_gff=$PATH_TO_CBP_maker/maker_inputs/legacy
  _data/legacy-set1.gff,
  $PATH_TO_CBP_maker/maker_inputs/legacy_data/
legacy-set2.gff #ab-initio predictions from an external
  GFF3 file
#-----MAKER Behavior Options
keep_preds=0
#Concordance threshold to add unsupported gene
  prediction (bound by 0 and 1)
```

Passing gene models in GFF3 format to MAKER as `pred_gff` *allows MAKER to update the models in the light of new evidence by adding new 3′ and 5′ exons, additional UTR, and merging split models. This method will not change internal exons, nor will it entirely delete any existing gene model. When run in this mode with an additional gene finder turned on, MAKER will also create new annotations where new evidence suggest a gene but no corresponding model was previously present. In this example, two annotation sets (*`legacy-set1.gff`, `legacy-set2.gff`*) are being merged and updated. When two models are annotated at the same locus, MAKER will chose the model that best matches the evidence for inclusion in the final annotation set. Setting* `keep_preds=1` *will ensure that no gene models are lost from the legacy annotations. If* `keep_preds=0` *is set, gene models that are not supported by the evidence will not be included in the final MAKER annotation build. For this example, we have set* `keep_pres=0` *because we are concerned about false positives in the legacy annotations.*

3. Run MAKER and check/collect the results as outlined in Basic Protocol 1, steps 3 to 5.

   The original legacy annotation sets contained 237 and 203 gene annotations:

```
% grep -cP '\tgene\t' \
$PATH_TO_CBP_maker/maker_inputs/legacy_data/legacy-
  set1.gff
237
% grep -cP '\tgene\t' \
$PATH_TO_CBP_maker/maker_inputs/legacy_data/legacy-
  set2.gff
203
```

The combined annotation set contains 180 genes. This number of genes in the combined set is a result of adding genes from one set that were not annotated in the other set, merging or splitting genes, and discarding genes that are not supported by the evidence.

```
% grep -cP '\tgene\t' legacy-contig.all.gff
180
```

In addition to reconciling these two annotation sets based on the evidence, MAKER also added 3′ and 5′ UTR features that were supported by the new RNA evidence. Neither of the legacy annotation sets contained three or five prime UTR features.

```
% grep -cP '\t(three|five)_prime_UTR\t'
  legacy-contig.all.gff
47
```

ADDING MAKER'S QUALITY-CONTROL METRICS TO ANNOTATIONS FROM ANOTHER PIPELINE

The MAKER annotation pipeline strives to be transparent in its use of evidence for each gene annotation. To accomplish this transparency, MAKER does two things. First, all of the evidence alignments, repeat masked regions, ab initio gene predictions, etc, are included in MAKER's GFF3 output with its annotations. Second, MAKER generates a series of quality metrics for each annotated gene model. These metrics include (1) the MAKER mRNA Quality index (QI), and (2) an Annotation Edit Distance (AED). Both of these data types are attached to each MAKER transcript.

The MAKER mRNA Quality index (QI) is a nine-dimensional summary of a transcript's key features and how they are supported by the data gathered by MAKER's compute pipeline. A typical QI might look as follows: `QI:0|0.77|0.68|1|0.77|0.78|19|462|824`. Table 4.11.2 provides a key for the QI data fields. Values are delimited by pipe symbols. Interpretation is easy. For example, the transcript with the QI string above has no 5′ UTR; 77% of its splice sites are confirmed by transcript data; 68% of its exons overlap transcript evidence; all of its exons overlap transcript or protein alignments; 77% of its splice sites are precisely confirmed by an ab initio gene prediction; 78% of its exons overlap an ab initio prediction; the transcript has 19 exons; the 3′ UTR is 462 base pairs long; and the protein it encodes is 824 amino acids in length. QI strings are easily parsed, and thus provide a good starting point for MAKER users seeking to write their own scripts for genome annotation quality control.

Also included in MAKER's GFF3 outputs is a second quality control measure called Annotation Edit Distance (AED; Eilbeck et al., 2009; Holt and Yandell, 2011; Yandell and Ence, 2012). MAKER and MAKER-P use AED to measure the goodness of fit of an annotation to the evidence supporting it. AED is a number between 0 and 1, with an AED of zero denoting perfect concordance with the available evidence and a value of one indicating a complete absence of support for the annotated gene model (Eilbeck et al., 2009). In other words, the AED score provides a measure of each annotated transcript's congruency with its supporting evidence. See (Yandell and Ence, 2012) for a further discussion of AED.

The protocol below adds QI tags and AED scores to gene models produced by other pipelines.

Table 4.11.1 MAKER Quality Index Summary (adapted from Cantarel et al., 2008)

Position	Definition
1	Length of the 5′ UTR
2	Fraction of splice sites confirmed by an EST/mRNA-seq alignment
3	Fraction of exons that match an EST/mRNA-seq alignment
4	Fraction of exons that overlap EST/mRNA-seq or protein alignments
5	Fraction of splice sites confirmed by ab initio gene prediction
6	Fraction of exons that overlap an ab initio gene prediction
7	Number of exons in the mRNA
8	Length of the 3′ UTR
9	Length of the protein sequence produced by the mRNA

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules [automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed]. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

Files

Genome assembly from the original annotation in FASTA format, protein evidence in FASTA format, assembled mRNA-seq transcripts from the species of interest in FASTA format, gene annotations in GFF3 format

1. Create MAKER control files as outlined in Basic Protocol 1, step 1.

2. Edit the `maker_opts.ctl` file to add the genomic sequence, evidence, and gene models you wish to add quality metrics to:

```
% emacs maker_opts.ctl
#-----Genome (these are always required)
genome=$PATH_TO_CBP_maker/maker_inputs/legacy_data/
  legacy-contig.fasta #genome sequence (fasta file or
  fasta embeded in GFF3 file)
organism_type=eukaryotic #eukaryotic or prokaryotic.
  Default is eukaryotic
#-----EST Evidence (for best results provide a file for
  at least one)
est=$PATH_TO_CBP_maker/maker_inputs/legacy_data/
  legecy-new-mRNAseq.fasta #set of ESTs or assembled
  mRNA-seq in fasta format
#-----Protein Homology Evidence (for best results
  provide a file for at least one)
protein=$PATH_TO_CBP_maker/maker_inputs/legacy_data/
  legacy-new-protein.fasta #protein sequence file in
  fasta format (i.e., from multiple organisms)
#-----Gene Prediction
model_gff=/home/mcampbell/project_links/CPB_maker/
  maker_inputs/legacy_data/legacy-set1.gff
#annotated gene models from an external GFF3 file
  (annotation pass-through)
```

   Annotations given to MAKER as `model_gff` *remain unchanged in the final MAKER output and are kept regardless of evidence support.*

3. Run MAKER and check/collect the results as outlined in Basic Protocol 1, steps 3 to 5. Then repeat this procedure for `legacy-set2.gff`.

   This gives us a GFF3 for each annotation set. They are renamed below for simplicity:

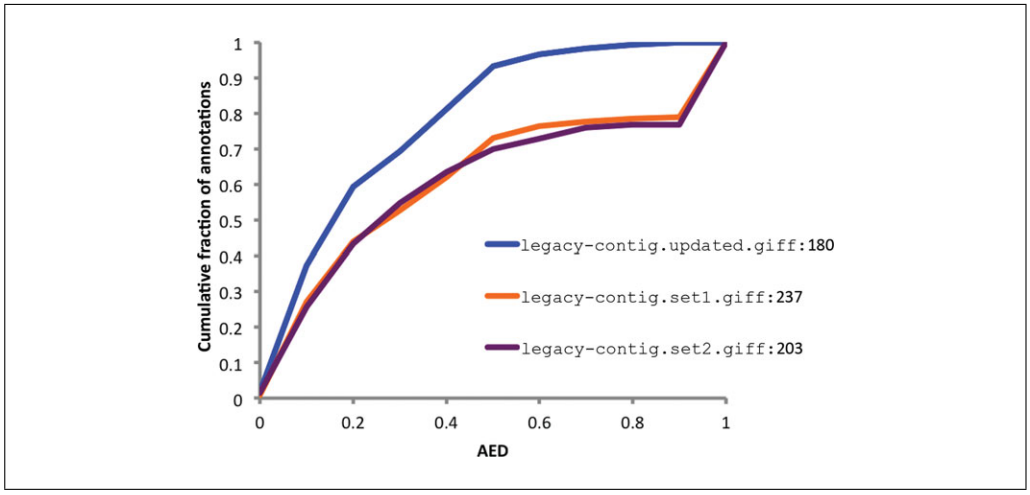

Figure 4.11.3 Adding quality metrics to legacy annotations facilitates comparison between annotation sets. Shown on the *y* axis is the cumulative distribution of AED for each dataset. The two legacy annotation sets are of comparable quality, with approximately 70% of their annotations having AEDs of less than 0.5 (orange and purple lines). Combining and updating legacy annotations results in a much improved annotation build (blue line), in which greater than 90% of the annotations have and AED less than 0.5.

```
legacy-contig.set1.gff
legacy-contig.set2.gff
```

Now that we have quality metrics for all of the annotations, we can compare them. A cumulative distribution function curve based on AED is a simple way to visually compare annotation sets. An example using the above annotation sets as well as the results from Basic Protocol 2, where we combined and updated these legacy annotations, is shown in Figure 4.11.3.

BASIC PROTOCOL 4

MAPPING ANNOTATIONS TO A NEW ASSEMBLY

Genome assemblies can change over time for a variety of reasons. Removing contaminants and improving assemblies with new genomic sequence data are two common reasons. Changes in the reference sequence make it necessary to also alter the beginning and ending coordinates of annotated genes. The simplest way to fix this is to use MAKER to map existing annotations forward onto the new assembly. The protocol below explains this process. Assembly changes can also invalidate pre-existing gene models, requiring structural revisions. In cases where the assembly has changed substantially. Basic Protocol 2, "Updating/combining legacy annotation datasets in light of new evidence," provides an easy means to simultaneously remap and update the existing gene annotations.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules [automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed]. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This

example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

Files

The new genome assembly in FASTA format, and the manually curated transcripts in FASTA format. This example uses the transcripts generated by MAKER in Basic Protocol 1, and a version of the genomic sequence with 60 bases removed from the first intron of the MAKER annotated gene.

1. Create MAKER control files as outlined in Basic Protocol 1, step 1.
2. Edit the `maker_opts.ctl` file to add the changed genomic sequence and the transcripts you wish to map forward:

```
% emacs maker_opts.ctl
#-----Genome (these are always required)
genome=$PATH_TO_CBP_maker/maker_inputs/new_assembly/
  new_assembly.fasta #genome sequence (fasta file or
  fasta embeded in GFF3 file)
organism_type=eukaryotic #eukaryotic or prokaryotic.
  Default is eukaryotic
#-----EST Evidence (for best results provide a file for
  at least one)
est=$PATH_TO_CBP_maker/maker_inputs/new_assembly/
  manualy_curated_transcript.fasta #set of ESTs or
  assembled mRNA-seq in fasta format
#-----Gene Prediction
est2genome=1 #infer gene predictions directly from
  ESTs, 1 = yes, 0 = no
```

MAKER will align the manually curated transcripts to the genome. By setting `est2genome=1`*, MAKER will create gene models directly from those alignments.*

3. Manually add the following line to the `maker_opts.ctl` file:

```
est_forward=1
```

By setting this hidden option in MAKER, the sequence id from the FASTA header will maintained as part of the gene name in the GFF3 output.

4. Run MAKER and check/collect the results as outlined in Basic Protocol 1, steps 3 to 5.

Shown below are the lines for exon two in the GFF3 file. The coordinates for exon 2 have shifted by 60 bp. Further exploration of the MAKER outputs will also show that the final transcript and protein outputs have not changed between the two assemblies.

```
Original assembly exon 2
contig-dpp-500-500 maker exon 27104 27985
Updated assembly exon 2
contig-dpp-500-500 maker exon 27044 27925
```

THE MAKER GENE BUILD/ RESCUING REJECTED GENE MODELS

BASIC PROTOCOL 5

MAKER users can decide which gene models to include in their final annotation build. This is accomplished using the MAKER tools and procedures described below. The resulting datasets are termed either *default*, *standard*, or *max*. The MAKER *default build* includes only those gene models that are supported by the evidence (i.e., AED <1.0). The MAKER-P *standard build* includes every gene model in the default build, plus

every ab initio gene prediction that encodes a Pfam domain as detected by InterProScan (Quevillon et al., 2005), and does not overlap an annotation in the MAKER default set. The MAKER *max build* includes every gene-model in the default build plus every ab initio gene prediction that does not overlap an annotation in the MAKER default set, regardless of whether or not it encodes a Pfam domain. We recommend that users choose the standard build, as previous work (Holt and Yandell, 2011; Campbell et al., 2014) has shown that this build procedure has the best overall accuracy. Nevertheless some users may prefer specificity to sensitivity, choosing the default build, whereas others may wish to include every possible gene model by using the max build procedure.

Necessary Resources

Hardware

Computer with a Unix-based operating system (e.g., Linux, Mac OS X)

Software

MAKER and MAKER-P are available for download at *yandell-lab.org*. Installation instructions are included in the tarball. For brevity's sake, the following protocols describe MAKER, but apply to MAKER-P as well.

MAKER will identify and download all of its necessary external dependencies including BLAST, Exonerate, RepeatMasker, and a number of Perl modules [automatic download and installation of Perl modules requires CPAN (*https://metacpan.org/pod/CPAN*) to be installed]. MAKER will also install a number of additional programs such as SNAP, Augustus, and MPICH2. This example uses a version of MAKER installed with NCBI BLAST+, Exonerate, RepeatMasker, with optional RepBase libraries, and SNAP.

InterProScan

Files

Genome assembly to be annotated in FASTA format

Protein evidence in FASTA format

Assembled mRNA-seq transcripts from the species of interest in FASTA format

Optional: a species parameter/HMM file for SNAP generated for the organism of interest or a closely related species. The process used to create a species parameter/HMM file is described in SNAP's internal documentation (Korf, 2004).

1. Create MAKER control files as outlined in Basic Protocol 1, step 1.

2. Edit the `maker_opts.ctl` file to add the genomic sequence, evidence, and specify the gene finding method(s):

```
% emacs maker_opts.ctl
#-----Genome (these are always required)
genome=$PATH_TO_CBP_maker/maker_inputs/pyu_data/pyu-
  contig.fasta #genome sequence (fasta file or fasta
  embeded in GFF3 file)
organism_type=eukaryotic #eukaryotic or prokaryotic.
  Default is eukaryotic
#-----EST Evidence (for best results provide a file for
  at least one)
est=$PATH_TO_CBP_maker/maker_inputs/pyu_data/pyu-
  est.fasta
#set of ESTs or assembled mRNA-seq in fasta format
#-----Protein Homology Evidence (for best results
  provide a file for at least one)
```

```
protein=$PATH_TO_CBP_maker/maker_inputs/pyu_data/pyu-
  protein.fasta #protein sequence file in fasta format
  (i.e., from multiple organisms)
#-----Gene Prediction
snaphmm=./pyu3.hmm #SNAP HMM file
#-----MAKER Behavior Options
keep_preds=1
#Concordance threshold to add unsupported gene
  prediction (bound by 0 and 1)
```

This step is very similar to step 2 in Basic Protocol 1. The key difference is setting `keep_preds=1` *in the MAKER behavior options section. Setting* `keep_preds=1` *prevents MAKER from rejecting unsupported gene models. The* `pyu3.hmm` *file was made using Support Protocol 1 and is found in the* `CPB_maker` *tarball described previously.*

3. Follow steps 3 to 5 in Basic Protocol 1.

   Instructing MAKER to retain unsupported gene models trades specificity for sensitivity. See Yandell and Ence (2012) for discussion of annotation specificity and sensitivity issues. For small, very compact genomes such as those of many fungi, this approach often works quite well; i.e., the sensitivity/specificity trade-off is minimal. However, for larger eukaryotic genomes, such as large animal and plant genomes, setting `keep_preds=1` *can result in thousands of false-positive gene models, so further filtering is necessary. One of the simplest ways to identify true positives is to run InterProScan (Quevillon et al., 2005) on the MAKER annotations. The idea is that a gene model without EST or protein homology that encodes a known protein domain is likely to be a true positive.*

4. Run InterProScan on the MAKER generated proteins to identify proteins with known functional domains:

```
% interproscan.sh -appl PfamA -iprlookup -goterms -f
  tsv\
-i pyu-contig.all.fasta
```

   The above example uses the stand-alone version of InterProScan and limits the search to Pfam domains. InterProScan can be run multiple ways and any of them that output a `.tsv` *file will work.*

5. Update the MAKER generated GFF3 file with the InterProScan results using `ipr_update_gff`:

```
% ipr_update_gff contig-dpp-500-500.gff\
pyu-contig.all.maker.proteins.fasta.tsv\
pyu-contig.max.functional_ipr.gff
```

   This procedure added a `Dbxref` *tag to column nine of the gene and mRNA features that have Pfam domains identified by InterProScan in the GFF3 file. The value for this tag contains InterPro and Pfam ids as well as the Gene Ontology ids associated with the identified domains, and looks like this:* `Dbxref=InterPro:IPR001300,Pfam:PF00648;Ontology_term=GO:0004198,GO:0005622,GO:0006508`. *The resulting GFF3 file from this command serves as the MAKER max build containing all gene models regardless of evidence support.*

6. Use the `quality_filter.pl` script distributed with MAKER to filter the gene models based on domain content and evidence support. Start by running `quality_filter.pl` without any options to see the usage:

```
% quality_filter.pl
quality_filter.pl: generates default and standard
```

```
 gene builds from a maker generated gff3_file with
 iprscan data pushed onto column 9 using
 ipr_update_gff.
USAGE: quality_filter.pl -[options] <gff3_file>
OPTIONS: -d Prints transcripts with an AED <1 (MAKER
 default)
-s Prints transcripts with an AED <1 and/or Pfam
 domain if in gff3 (MAKER Standard)
-a <number between 0 and 1> Prints transcripts
 with an AED < the given value
```

We can generate the MAKER default build and the MAKER standard build using the `-d` and `-s` options respectively:

```
% quality_filter.pl -d pyu-
  contig.max.functional_ipr.gff\
> pyu-contig.default.functional_ipr.gff
% quality_filter.pl -s pyu-
  contig.max.functional_ipr.gff\
> pyu-contig.standard.functional_ipr.gff
```

When we count the number of genes in these two files, we can see that we were able to rescue 161 genes that were not annotated due to lack of evidence but are supported by Pfam domain content:

```
% grep -cP '\tgene\t'\
pyu-contig.default.functional_ipr.gff
404
% grep -cP '\tgene\t'\
pyu-contig.standard.functional_ipr.gff
565
```

This procedure was used in the MAKER-P paper for benchmarking MAKER-P on the Arabidopsis genome. When the gene models with Pfam domain support were included, sensitivity improved at the expense of specificity, but the best accuracy was obtained using the TAIR10 annotations as truth (Campbell et al., 2014).

GUIDELINES FOR UNDERSTANDING RESULTS

MAKER and MAKER-P are designed with three general use-case scenarios in mind. These are (1) de novo annotation of new genomes; (2) updating annotations to reflect assembly changes and/or new evidence; and (3) quality control of genome annotations. Classic model-organism genomes such mouse (Waterston et al., 2002), *C. elegans* (Press et al., 1998), and *Drosophila melanogaster* (Adams et al., 2000) benefited from pre-existing gold-standard gene annotations. These were used to train gene finders and to evaluate the accuracy of genome annotations. In contrast, the genomes being sequenced today are novel, and their contents are unknown. Thus, evidence, in the form of transcript and protein alignments, must be used as a surrogate for gold-standard annotations. Accordingly, MAKER and MAKER-P provide means for employing transcript and protein alignments to train gene finders and for evaluating the accuracy of the genome annotations, i.e., quality control. These operations are primarily accomplished using Annotation Edit Distance (AED). AED is a distance measure that summarizes the congruency of each annotation with its supporting evidence. A value of 0 indicates that the annotation matches the evidence perfectly, while a value of 1 indicates that the annotation has no evidence support. See Yandell and Ence (2012) for more discussion on this topic; also see Cantarel et al. (2008); Eilbeck et al. (2009; Holt and Yandell (2011).

Protein domain content provides another means to judge the quality of de novo protein coding annotations. Previous work (Holt and Yandell, 2011) has shown that somewhere between 55% to 65% of the proteins comprising a well annotated eukaryotic proteome will contain a recognizable domain. See Basic Protocol 5 for more on how to employ MAKER and MAKER-P to carry out domain-based analyses of annotations.

Together, AED and proteome domain content provide two simple summary statistics with which to globally compare one genome's annotations to another's (Holt and Yandell, 2011). As a rule of thumb, a genome annotation build where 90% of the annotations have an AED less than 0.5, and over 50% of its proteome contains a recognizable domain, can be considered well annotated (Holt and Yandell, 2011; Yandell and Ence, 2012; Campbell et al., 2014).

Gene number is a third important summary statistic for evaluating the overall quality of a genome annotation build. Clearly, a build comprising only a handful of genes is hardly a satisfactory result, no matter how domain-rich their proteins, or how well they agree with the transcript and protein alignment evidence. Unfortunately, there is no sure way to determine gene number for a genome. Some guidance, however, can be had from considering gene numbers from model organism genomes. Generally—and biology is full of exceptions—MAKER users should expect to see somewhere around 10,000 protein-coding annotations for fungal genome, between 12,000 and 20,000 for an invertebrate genome, and around 20,000 to 30,000 for a vertebrate genome. Plant gene numbers are even more difficult estimate because whole-genome duplications are common in plant evolution, but somewhere between 20,000 and 40,000 protein-coding genes are a good first guess. Consider too that fragmented assemblies will inflate these numbers, as a gene will often be split across multiple scaffolds. Again, keep in mind that these are ballpark figures. Biology is all about exceptions to the rule. This is one reason that MAKER and MAKER-P offer three different annotation build protocols: default, standard, and max. Generally, the MAKER default build provides a useful lower bound of well annotated genes with which to estimate gene numbers, the max build an upper bound, and the standard build a best first estimate for gene number.

COMMENTARY

Background Information

MAKER was developed as an easy-to-use annotation pipeline for emerging model organism genomes (Cantarel et al., 2008). The overarching goal of MAKER was to enable small, independent research groups without extensive bioinformatics expertise or resources to annotate genomes.

MAKER 2 is a backwardly compatible extension of MAKER (Holt and Yandell, 2011). MAKER2 improved MAKER's gene-finding capabilities, offering improved, dynamic means to inform gene predictors, and provided new means for quality control using AED, as well as means for updating legacy annotations in light of new transcript and protein evidence.

MAKER-P is designed to address the needs of the plant genome community. MAKER-P provides means for annotation of complex plant genomes, and for automated revision, quality control, and management of existing genome annotations. MAKER-P also provides means for annotation of ncRNA genes and pseudogene annotation. MAKER-P is dramatically faster than other genome-annotation pipelines, including the original MAKER2, allowing it to scale to even the largest plant genomes. Recent work, for example, has shown that the version of MAKER-P available within the iPlant Cyberinfrastructure can re-annotate the entire maize genome in less than 3 hr (Campbell et al., 2014), and that it can carry out the complete de novo annotation of the 17.83-GB draft loblolly pine genome in less than 24 hr (Neale et al., 2014; Wegrzyn et al., 2014). MAKER-P can be used to annotate any genome, not just plants, and is now the main production release of the MAKER pipeline.

MAKER, MAKER2, and MAKER-P are available for download at *http://www.yandell-lab.org*. In addition, MAKER-P is available on the iPlant Cyberinfrastructure. Instructions for using MAKER-P on iPlant can be found at *https://pods.iplantcollaborative.org/wiki/display/sciplant/MAKER-P+at+iPlant*.

Critical Parameters

Critical parameters are defined here as parameters that will have global effects on the de novo annotation of a genome. These fall into four broad classes: repeat masking, evidence alignment, gene prediction, and MAKER behavior.

Repeat masking. Good repeat masking is essential in producing high-quality gene annotations. When not adequately masked, portions of transposable elements can be erroneously included in annotations of neighboring protein-coding genes. Species-specific repeat libraries will provide the best masking, especially for organisms that are phylogenetically distant from those currently found in RepBase (Jurka et al., 2005). See *http://weatherby.genetics.utah.edu/MAKER/wiki/index.php/Repeat_Library_Construction--Basic* and *http://weatherby.genetics.utah.edu/MAKER/wiki/index.php/Repeat_Library_Construction--Advanced* for basic and advanced protocols for generating species specific repeat libraries, respectively.

Evidence. It is crucial that MAKER has access to as complete an evidence dataset as possible. Ideally these data will include assembled RNA-seq transcripts from several tissues and developmental time points, as well as the complete proteomes of both a closely related organism and of an outgroup to account for lineage-specific gene loss. It is also advisable to include an omnibus protein dataset such as UniProt/Swiss-Prot. If RNA-seq data is not available, high-quality gene annotations can still be obtained from protein data alone, but they will lack untranslated regions (UTR), and MAKER may miss genes specific to the organism at hand. Remember that, by default, MAKER will not annotate genes that have no evidence support, so incomplete evidence datasets can lead to lower overall gene counts.

Gene prediction. It is important to understand that MAKER does not predict genes; rather, the gene finders you select in the control files predict the genes (SNAP, Augustus, etc.). Poorly trained gene finders will result in lower-quality final annotations. The gene finders will perform better inside of MAKER than they would have on their own because of evidence-derived hints being passed to them by MAKER (see Holt and Yandell, 2011, for more on this point), but the better trained the gene finder, the better this process will work. See Support Protocol 1 for directions for using MAKER to train gene finders. Consider too that not all gene finders perform well on every organism. A gene finder that performs well on fungi may not perform as well on plants and animals. Don't be afraid to remove a poorly performing gene finder from your analysis. Poor performance from multiple gene predictors likely indicates other problems such as insufficient repeat masking. You may want to build a species-specific repeat library for use with MAKER. A Web tutorial outlining this process is available at *http://weatherby.genetics.utah.edu/MAKER/wiki/index.php/Repeat_Library_Construction--Advanced*. There may also be widespread assembly errors or assembly fragmentation problems (these break open reading frames and erase potential splice sites, making it impossible to generate accurate annotations). Programs such as CEGMA (Parra et al., 2007) can be used to estimate what fraction of gene content will be recoverable from your genome assembly.

MAKER behavior. Important MAKER options that should be kept in mind include `split_hit`, `max_dna_len`, and `single_exon`. These options are set in the `maker_opts.ctl` file. The value of `split_hit` can be thought of as the longest intron that you expect in your genome. As a rule of thumb, 20 kb for vertebrates and 40 kb for mammals are reasonable values to try first. The default 10 kb works for many plants and most invertebrates and fungi. However, you may want to set it even lower for gene-dense genomes with short introns. Setting this value too low will result in truncated annotations, while setting it too high can result in concatenated genes (as evidence alignments will extend across neighboring paralogs). The `max_dna_len` parameter controls the window size for the genomic blocks MAKER will operate on at a time (larger values increase memory usage). This value must be set to at least three times the `split_hit` value to avoid issues with very large genes extending across multiple windows.

The `single_exon` parameter controls whether or not MAKER will consider single-exon EST alignments when generating hints for gene predictors. It is turned off by default. Setting `single_exon=1` will allow MAKER to annotate single-exon genes based on unspliced EST/mRNA-seq data, but will also greatly increase the false-positive rate for gene annotation. Single-exon alignments often result from spurious alignments, library contamination, background transcription of the genome, pseudogenes, and repeat elements. These facts should be considered carefully before enabling the `single_exon` parameter.

Nevertheless, for intron-poor genomes, you may want to turn this option on. If you choose to do so, the `single_length` parameter can be used to set a minimum size for single-exon alignments to accept. Shorter alignments are more likely to be spurious than longer alignments; 250 base pairs is a good minimum value for this parameter.

Troubleshooting

MAKER users should subscribe to the `MAKER_dev` mailing list (*http://yandell-lab.org/mailman/listinfo/maker-devel_yandell-lab.org*). Answers to common MAKER use errors can be found by searching the archived posts from the MAKER mailing list found at *https://groups.google.com/forum/#!forum/maker-devel*.

Advanced Parameters

MAKER has a large number of options and parameters. For a full list of the MAKER control file options including descriptions, see Table 4.11.2.

Acknowledgments

This work was supported by NSF IOS-1126998 to MY. A portion of M.C.'s efforts were supported by NIH 1R01GM104390-01 to MY.

Literature Cited

Adams, M.D., Celniker, S.E., Holt, R.A., Evans, C.A., Gocayne, J. D., Amanatides, P. G., and Venter, J.C. 2000. The genome sequence of *Drosophila melanogaster*. *Science* 287:2185-2195.

Bradnam, K.R., Fass, J.N., Alexandrov, A., Baranay, P., Bechner, M., Birol, I., Boisvert, S., Chapman, J.A., Chapuis, G., Chikhi, R., Chitsaz, H., Chou, W.C., et al. 2013. Assemblathon 2: Evaluating de novo methods of genome assembly in three vertebrate species. *GigaScience* 2:10. doi:10.1186/2047-217X-2-10.

Campbell, M., Law, M., Holt, C., Stein, J., Moghe, G., Hufnagel, D., Lei, J., Achawanantakun, R., Jiao, D., Lawrence, C.J., Ware, D., Shiu, S.H., Childs, K.L., Sun, Y., Jiang, N., and Yandell, M. 2013. MAKER-P: A tool-kit for the rapid creation, management, and quality control of plant genome annotations. *Plant Physiol.* 164:513-524.

Cantarel, B.L., Korf, I., Robb, S.M.C., Parra, G., Ross, E., Moore, B., Holt, C., Sanchez Alvarado, A., and Yandell, M. 2008. MAKER: An easy-to-use annotation pipeline designed for emerging model organism genomes. 18:188-196.

Eilbeck, K., Moore, B., Holt, C., and Yandell, M. 2009. Quantitative measures for the management and comparison of annotated genomes. *BMC Bioinformatics* 10:67.

Goff, S.A., Vaughn, M., McKay, S., Lyons, E., Stapleton, A.E., Gessler, D., Matasci, N., Wang, L., Hanlon, M., Lenards, A., Muir, A., Merchant, N., et al. 2011. The iPlant collaborative: Cyberinfrastructure for plant biology. *Front. Plant Sci.* 2:34.

Holt, C. and Yandell, M. 2011. MAKER2: An annotation pipeline and genome-database management tool for second-generation genome projects. *BMC Bioinformatics* 12:491.

Jurka, J., Kapitonov, V.V, Pavlicek, A., Klonowski, P., Kohany, O., and Walichiewicz, J. 2005. Repbase Update, a database of eukaryotic repetitive elements. *Cytogenet. Genome Res.* 110:462-467.

Korf, I. 2004. Gene finding in novel genomes. *BMC Bioinformatics* 5:59.

Law, M., Childs, K.L., Campbell, M.S., Stein, J.C., Holt, C., Olson, A.J., Holt, C., Lei, J., Jiao, D. Andorf, C.M., Ware, D. Shiu, S.-H., Sun, Y. Jiang, N., and Yandell, M. 2014. Automated update, revision and quality control of the *Zea mays* genome annotations using MAKER-P improves the B73 RefGen_v3 gene models and identifies new genes. *Plant Physiol.* In press.

Lévesque, C.A., Brouwer, H., Cano, L., Hamilton, J.P., Holt, C., Huitema, E., Raffaele, S., Robideau, G.P., Thines, M., Win, J., Zerillo, M.M. Beakes, G.W., et al. 2010. Genome sequence of the necrotrophic plant pathogen *Pythium ultimum* reveals original pathogenicity mechanisms and effector repertoire. *Genome Biol.* 11:R73.

Lipman, D.J. and Pearson, W.R. 1985. Rapid and sensitive protein similarity searches. *Science* 227:1435-1441.

Lowe, T.M. 1999. A computational screen for methylation guide snoRNAs in yeast. *Science* 283:1168-1171.

Lowe, T.M. and Eddy, S.R. 1997. tRNAscan-SE: A program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Res.* 25:955-964.

Campbell, M.S., Law, M., Holt, C., Stein, J.C., Moghe, G.D., Hufnagel, D.E., Lei, J., Achawanantakun, R., Jiao, D., Lawrence, C.J., Ware, D., Shiu, S.H., Childs, K.L., Sun, Y., Jiang, N., and Yandell, M. 2014. MAKER-P: A tool kit for the rapid creation, management, and quality control of plant genome annotations. *Plant Physiol.* 164:513-524.

Neale, D.B., Wegrzyn, J.L., Stevens, K.A., Zimin, A.V., Puiu, D., Crepeau, M.W., Cardeno, C., Koriabine, M., Holtz-Morris, A.E., Liechty, J.D., Martínez-García, P.J., Vasquez-Gross, H.A., et al. 2014. Decoding the massive genome of loblolly pine using haploid DNA and novel assembly strategies. *Genome Biol.* 15:R59.

Parra, G., Bradnam, K., and Korf, I. 2007. CEGMA: A pipeline to accurately annotate core genes in eukaryotic genomes. *Bioinformatics* 23:1061-1067.

Press, H., York, N., and Nw, A. 1998. Genome sequence of the nematode *C. elegans*: A platform for investigating biology. *Science* 282:2012-2018.

Quevillon, E., Silventoinen, V., Pillai, S., Harte, N., Mulder, N., Apweiler, R., and Lopez, R. 2005. InterProScan: Protein domains identifier. *Nucleic Acids Res.* 33:W116-W120.

Waterston, R.H., Lindblad-Toh, K., Birney, E., Rogers, J., Abril, J.F., Agarwal, P., Agarwala, R., Ainscough, R., Alexandersson, M., An, P., Antonarakis, S.E., Attwood, J., et al. 2002. Initial sequencing and comparative analysis of the mouse genome. *Nature* 420:520-562.

Wegrzyn, J.L., Liechty, J.D., Stevens, K.A., Wu, L.-S., Loopstra, C.A., Vasquez-Gross, H.A., Dougherty, W.M., Lin, B.Y., Zieve, J.J., Martínez-García, P.J., Holt, C., Yandell, M., Zimin, A.V., Yorke, J.A., Crepeau, M.W., Puiu, D., Salzberg, S.L., Dejong, P.J., Mockaitis, K., Main, D., Langley, C.H., and Neale, D.B. 2014. Unique features of the loblolly pine (*Pinus taeda* L.) megagenome revealed through sequence annotation. *Genetics* 196:891-909.

Yandell, M. and Ence, D. 2012. A beginner's guide to eukaryotic genome annotation. *Nat. Rev. Genet.* 13:329-342.

Zimin, A., Stevens, K.A., Crepeau, M.W., Holtz-Morris, A., Koriabine, M., Marçais, G., Puiu, D., Roberts, M., Wegrzyn, J.L., de Jong, P.J., Neale, D.B., Salzberg, S.L., Yorke, J.A., and Langley, C.H. 2014. Sequencing and assembly of the 22-gb loblolly pine genome. *Genetics* 196:875-890.

Zou, C., Lehti-Shiu, M.D., Thibaud-Nissen, F., Prakash, T., Buell, C.R., and Shiu, S.-H. 2009. Evolutionary and expression signatures of pseudogenes in Arabidopsis and rice. *Plant Physiol.* 151:3-15.

Internet Resources

http://www.sequenceontology.org/gff3.shtml

Generic Feature Format version 3.

https://metacpan.org/pod/CPAN

CPAN Web site (A. König, developer).

Table 4.11.2 MAKER Control File Options Found in the `maker_opts.ctl` and `maker_bopts.ctl` Files

Option	Comments
`maker_opts.ctl`	
`#-----Genome (these are always required)`	Headings for sections in the control files are marked by a pound sign and five dashes. These headings are not actually used by MAKER but are helpful when trying to find a specific option or parameter.
`genome= #genome sequence (fasta file or fasta embeded in GFF3 file)`	This is a single multifasta file that contains the assembled genome. Both absolute and relative file paths are allowed. It is also important to note that though there are a large number of characters accepted by FASTA format to represent nucleotides, many of them are not supported by some of the tools MAKER calls, so make sure that FASTA sequence contains only A, T, C, G, and N.
`organism_type=eukaryotic #eukaryotic or prokaryotic. Default is eukaryotic`	MAKER's default is eukaryotic. Setting this to prokaryotic changes some of MAKER's behavior options automatically (such as turning off repeat masking).
`#-----Re-annotation Using MAKER Derived GFF3`	This section was developed as a convenience method for using the output of a previous MAKER run as the evidence to a new MAKER run.
`maker_gff= #MAKER derived GFF3 file`	Path to the MAKER generated GFF3 file.
`est_pass=0 #use ESTs in maker_gff: 1 = yes, 0 = no`	Set to `1` to use the EST/mRNA-Seq alignments from the MAKER file. See `est=` below for details.
`altest_pass=0 #use alternate organism ESTs in maker_gff: 1 = yes, 0 = no`	Set to `1` to use the alternative EST/mRNA-seq alignments from the MAKER file. See `altest=` below for details.

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
`maker_opts.ctl`	
`protein_pass=0 #use protein alignments in maker_gff: 1 = yes, 0 = no`	Set to 1 to use the protein alignments from the MAKER file. See `protein=` below for details.
`rm_pass=0 #use repeats in maker_gff: 1 = yes, 0 = no`	Set to 1 to use the repeat masking data from the MAKER file. See the `#-----Repeat Masking` section below for details.
`model_pass=0 #use gene models in maker_gff: 1 = yes, 0 = no`	Set to `1` to use the gene models from the MAKER file. See `model_gff=` below for details.
`pred_pass=0 #use ab-initio predictions in maker_gff: 1 = yes, 0 = no`	Set to `1` to use the gene predictions from the MAKER file. See `pred_gff=` below for details.
`other_pass=0 #passthrough anyything else in maker_gff: 1 = yes, 0 = no`	Set to 1 to pass any other features through from a previous MAKER file. See other`_gff=`, below for details.
`#-----EST Evidence (for best results provide a file for at least one)`	This section contains options pertaining to Transcript Evidence., e.g., EST, mRNA-seq and assembled full length cDNAs. These are assumed to be correctly assembled and they will be aligned in a splice aware fashion (MAKER uses Exonerate to do this). MAKER can use these alignments to infer gene models directly when the est2genome option is turned on. MAKER also uses them as support for intron/exon boundaries in hints sent to the gene finders, and for AED calculations. MAKER also uses these data to infer alternate splice forms and UTR regions. How these alignments cluster with other evidence (protein, for example) will help MAKER infer gene boundaries in some cases.
`est= #set of ESTs or assembled mRNA-seq in fasta format`	Specifies files containing assembled mRNA-Seq transcripts, ESTs, or full-length cDNAs. You may provide multiple files in a comma-separated list.
`altest= #EST/cDNA sequence file in fasta format from an alternate organism`	Specifies files containing assembled mRNA-Seq transcripts, ESTs, or full-length cDNAs from *another* related organism. This option is useful when there is no transcript evidence available for the genome at hand, but this data is available for a closely related species. However, these alignments are done using tblastx, which makes these data very expensive computationally. Use protein evidence from a relate species if at all possible before using transcript evidence. You may provide multiple files in a comma-separated list.
`est_gff= #aligned ESTs or mRNA-seq from an external GFF3 file`	These are prealigned transcripts from the organism being annotated in GFF3 format. The most common sources of these kinds of data are alignment based transcript assemblers such as cufflinks, or outputs from a previous MAKER run. You may provide multiple files in a comma-separated list.

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
`maker_opts.ctl`	
`altest_gff= #aligned ESTs from a closly relate species in GFF3 format`	These are prealigned transcripts from a related species in GFF3 format. The most common source of these kinds of data is output from a previous MAKER run. You may provide multiple files in a comma-separated list.
`#-----Protein Homology Evidence (for best results provide a file for at least one)`	This section of the control file covers options controlling the use of protein homology evidence. Protein homology evidence helps MAKER locate coding regions and gene boundaries. These alignments will also be used to generate hints for the gene finders and as part of the AED calculation.
`protein= #protein sequence file in fasta format (i.e., from mutiple oransisms)`	This is a collection of protein sequences (usually from related species) in FASTA format. A minimum of one full proteome from a related species should be used. Multiple files in a comma-separated list are allowed.
`protein_gff= #aligned protein homology evidence from an external GFF3 file`	These are pre-aligned proteins in GFF3 format. The most common source of these data is a previous MAKER run. Multiple files in a comma-separated list are allowed.
`#-----Repeat Masking (leave values blank to skip repeat masking)`	Repeats will be masked to stop EST and proteins from aligning to repetitive regions and to keep gene prediction algorithms from being allowed to call exons in those regions.
`model_org=all #select a model organism for RepBase masking in RepeatMasker`	Specifies the model organism to use for RepeatMasker when RepBase libraries are installed. Common values are mammal, grass, primate, fungi, etc. The genus and species can also be used so long as they are bound by double quotes e.g., "`drosophila melanogaster`". See RepeatMasker documentation for valid entries. You can also use the value `simple` to specify only low complexity repeats (this option is MAKER specific). If you have gone to the trouble of making a custom repeat library (i.e., `rmlib=`) you do not need to use RepBase and may leave this option blank.
`rmlib= #provide an organism specific repeat library in fasta format for RepeatMasker`	Specifies the location of a custom repeat library. The file should be in FASTA format.
`repeat_protein= #provide a fasta file of transposable element proteins for RepeatRunner`	Specify transposable element proteins in FASTA format. A default file comes packaged with MAKER. These are aligned in protein space to help mask known transposable elements that have diverged over time.
`rm_gff= #pre-identified repeat elements from an external GFF3 file`	These are pre-aligned repeats in GFF3 format. The most common source of these data is a previous MAKER run, but these are also available from some organism databases.

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
`maker_opts.ctl`	
`prok_rm=0 #forces MAKER to repeatmask prokaryotes (no reason to change this), 1 = yes, 0 = no`	As a general rule, masking a prokaryotic genome is unnecessary and can lead to truncated gene models.
`softmask=1 #use soft-masking rather than hard-masking in BLAST (i.e., seg and dust filtering)`	Soft-masking in BLAST prevents alignments from seeding in regions of low complexity but allows alignments to extend through these regions.
`#-----Gene Prediction`	This section covers the gene finders used by MAKER. Unless gene finders are specified, MAKER will not annotate any genes. MAKER will run each gene predictor without hints once (ab initio predictions) and once with hints. Models produced by the gene finder will only be maintained in the final annotation set if there is some form of evidence supporting their structure. If multiple models overlap, only the one with the lowest AED (best evidence match) will be maintained in the final annotation set.
`snaphmm= #SNAP HMM file`	Specifies the location of the HMM file required to run SNAP. Always use an HMM specific for the genome at hand if at all possible, although a related species can be used to generate models that can then be used for training for the genome at hand. Multiple files in a comma-separated list are allowed.
`gmhmm= #GeneMark HMM file`	Specify an HMM file for GeneMark. Multiple files in a comma-separated list are allowed.
`augustus_species= #Augustus gene prediction species model`	Specify the species model to use for Augustus. This is just a name and not a file path. To get a list of valid options, look in the `../augustus/config/species` directory. Multiple files in a comma- separated list are allowed.
`fgenesh_par_file= #FGENESH parameter file`	Location of an FGENESH parameter file. Multiple files in a comma- separated list are allowed.
`pred_gff= #ab-initio predictions from an external GFF3 file`	Predictions from any gene finder can be used in MAKER, so long as the gene finder's output has been converted to GFF3 format. Multiple files in a comma- separated list are allowed.
`model_gff= #annotated gene models from an external GFF3 file (annotation pass-through)`	These are assumed to be high-confidence gene models usually from a previous annotation of the genome. Because these models are considered high confidence, they will be used to merge evidence clusters around existing loci. This clustering will slightly bias MAKER towards keeping rather than replacing previous models for borderline cases. MAKER is only allowed to keep or replace these models and cannot modify them, although if `map_forward=1` is set, their names will be mapped forward onto whatever model replaces them. If no evidence supports these models,

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
maker_opts.ctl	
	MAKER will still keep them because they are assumed to be high confidence (but MAKER will tag them with an AED score of 1). Multiple files in a comma- separated list are allowed.
est2genome=0 #infer gene predictions directly from ESTs, 1 = yes, 0 = no	This option is used to create gene models directly from the transcript evidence. This option is useful when no gene predictor is trained on your organism or there is not a training file available from a closely related organism. The gene models from this option will be fragmented and incomplete because of the nature of transcript data (especially mRNA-Seq). These gene models are most useful for first round training of gene finders. Once there is a trained gene predictor, turn this option off.
protein2genome=0 #infer predictions from protein homology, 1 = yes, 0 = no	Similar to est2genome. This option will make gene models directly from protein alignments. Like est2genome this option is most useful for training gene predictors and should be turned off afterwards.
trna=0 #find tRNAs with tRNAscan, 1 = yes, 0 = no	Set to 1 to use tRNAscan-SE to annotate tRNAs.
snoscan_rrna= #rRNA file to have Snoscan find snoRNAs	Specify a FASTA file containing rRNAs that will be used by snoscan to annotate snoRNAs.
unmask=0 #also run ab-initio prediction	This option lets the gene finders run on the unmasked sequence as well as the masked sequence.
programs on unmasked sequence, 1 = yes, 0 = no	This allows repetitive regions to be included in gene models (does not affect evidence alignment), which may be useful in cases where over masking of repeats is suspected.
#-----Other Annotation Feature Types (features MAKER doesn't recognize)	This section covers parameters that allow users to add additional annotations to MAKER's set.
other_gff= #extra features to pass-through to final MAKER generated GFF3 file	These are GFF3 lines you just want MAKER to add to your files. These are things MAKER does not annotate: promoter/enhancer regions, CpG islands, restrictions sites, etc. MAKER will not attempt to validate the features, but will just pass them through "as is" into the final GFF3 file. Multiple files in a comma-separated list are allowed.
#-----External Application Behavior Options	These options are passed to external programs like BLAST and can usually be left as default, especially if you are running MAKER with MPI.
alt_peptide=C #amino acid used to replace non-standard amino acids in BLAST databases	This option allows the user to specify amino acid codes that will be used to replace non-standard amino acids in protein alignment databases used by BLAST and Exonerate. Cysteine (C) is the default because it has the lowest overall substitution penalty of all of the amino acids in the BLOSUM matrix.

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
`maker_opts.ctl`	
`cpus=1 #max number of CPUs to use in BLAST and RepeatMasker (not for MPI, leave 1 when using MPI)`	Specifies the number of CPUs to use when running BLAST. If using MAKER with MPI, leave this as 1 or it will act like a multiplier to the CPUs already specified by mpiexec and can overburden your job.
`#-----MAKER Behavior Options`	These options affect internal MAKER behavior. They can be tuned to help MAKER run more effectively.
`max_dna_len=100000 #length for dividing up contigs into chunks (increases/decreases memory usage)`	Affects the window size used by MAKER for looking at blocks of the genome. Larger values use more memory. It is important that this parameter be at least three times the expected maximum intron size, or genes can bridge multiple windows and performance will suffer. 300,000 is a good `max_dna_len` on large vertebrate genomes if memory is not a limiting factor.
`min_contig=1 #skip genome contigs below this length (under 10kb are often useless)`	Causes MAKER to skip short contigs without attempting to annotate them. For large, repeat-rich genomes, setting this option to 10,000 can decrease run time without sacrificing annotation quality because contigs shorter than this are usually un-annotatable (they are too short to contain a full gene).
`pred_flank=200 #flank for extending evidence clusters sent to gene predictors`	Gene finders require flanking sequence on either side of a gene to correctly find start and stop locations. This parameter adds flanking sequence to evidence clusters to ensure the required flanking sequence is there. This option also affects how close evidence islands must be before clustering together. If you are annotating a genome with a sparse/fragmented evidence set increasing this value can capture exons missing from your evidence. Decreasing this value can help decrease gene mergers in organisms with high gene density.
`pred_stats=0 #report AED and QI statistics for all predictions as well as models`	Adds AED and QI statistics to the reference ab initio models in the GFF3. This can be computationally expensive, but can be useful when evaluating rejected gene models.
`AED_threshold=1 #Maximum Annotation Edit Distance allowed (bound by 0 and 1)`	Restricts the final gene models to have at least a given threshold of evidence support. Setting this option to a value lower than 1 will result in a final annotation set with fewer gene models but they will be better supported by the evidence.
`min_protein=0 #require at least this many amino acids in predicted proteins`	Sometimes gene predictors can generate very short predictions, especially on fragmented genomes with very short contigs. Setting this option can filter them out from the final annotation set.
`alt_splice=0 #Take extra steps to try and find alternative splicing, 1 = yes, 0 = no`	When this parameter is set to `0` MAKER will generate a single transcript for each gene that best matches the evidence. When set to `1`, MAKER will separate the transcript evidence into mutually

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
`maker_opts.ctl`	
	exclusive intron/exon sets. The information from each evidence set is then independently given to the gene finders as hints. If the gene finder predicts an alternative transcript using the alternate evidence set, then it is kept as an isoform in the final GFF3 output. Be careful when using this feature of MAKER in conjunction with noisy RNA-seq data, as this can result in an excess of alternative transcripts being predicted.
`always_complete=0 #extra steps to force start and stop codons, 1 = yes, 0 = no`	Will extend or truncate gene models to try and force canonical start/stop codons even if they are not biologically correct.
`map_forward=0 #map names and attributes forward from old GFF3 genes, 1 = yes, 0 = no`	When a gene from `model_gff` input is replaced with a new model, that new model will inherit the name from the model it replaced. Allows for naming conservation when re-annotating a genome.
`keep_preds=0 #Concordance threshold to add unsupported gene prediction (bound by 0 and 1)`	This is used when you want an annotation set with maximum sensitivity. As a general rule, gene finders tend to over-predict on novel genomes, so MAKER rejects models that do not have at least some form of evidence support. This flag removes the evidence support requirement. On some genomes with high gene density where over-prediction is modest (fungi, oomycetes, etc.), setting this parameter to `1` can be beneficial. However, doing so on larger plant and animal genomes can lead to false-positive gene calls, outnumbering true gene models by an order of magnitude or more.
`split_hit=10000 #length for the splitting of hits (expected max intron size for evidence alignments)`	This option is currently used to keep BLAST from aligning transcripts and proteins with exons unreasonably far apart, which can causes false merging of neighboring paralogs or spurious alignment of terminal exons.
`single_exon=0 #consider single exon EST evidence when generating annotations, 1 = yes, 0 = no`	By default MAKER does not use single exon transcript alignments as supporting evidence for gene models. Single exon alignments overwhelmingly represent spurious alignments, library contamination, background transcription of the genome, pseudogenes, and repeat elements. This somewhat decreases the sensitivity of MAKER, but greatly improves the specificity and overall accuracy. Turn this parameter on if the genome contains many single exon genes.
`single_length=250 #min length required for single exon ESTs if 'single_exon is enabled'`	If `single_exon` is set to `1`, this option filters out the shortest alignments, because spurious alignments and contamination tend to be biased toward shorter sequences.

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
`maker_opts.ctl`	
`correct_est_fusion=0 #limits use of ESTs in annotation to avoid fusion genes`	This option helps prevent merging of gene models because of overlapping UTRs (common in fungal genomes) or because of falsely merged RNA-seq assemblies (e.g., you did not turn on the Jaccardian clip option when running Trinity). If you see gene models where transcript evidence is causing a neighboring gene model to be merged into the UTR, or you see gene models that are being rejected only because they slightly overlap the UTR of a neighboring gene, then turn this option on. It will trim back the low confidence UTRs on both genes to allow both models into the final annotation set.
`tries=2 #number of times to try a contig if there is a failure for some reason`	Sets the maximum number of retries before MAKER considers an assembly contig to have failed. Large computes especially in cluster environments can be hindered by random failures caused by the network or I/O performance. This option gets past such failures by just trying again. It will not, however, get around systematic failures caused by errors in your dataset.
`clean_try=0 #remove all data from previous run before retrying, 1 = yes, 0 = no`	MAKER tries to recover from failures before trying a contig again, and it starts off where it left off in the analysis. However, some failures can result in irrecoverable file corruption that MAKER cannot fix. In those cases, it is better to just delete all files from the contig and start again from scratch. This is the best way to get around stubborn random failures caused by slow or unreliable NFS file system implementations.
`clean_up=0 #removes theVoid directory with individual analysis files, 1 = yes, 0 = no`	This option will help save disk space by deleting individual raw results files (such as BLAST, Exonerate, and gene predictor outputs) once they are no longer needed. If you have the disk space it is usually best to keep this set to `0`. Having those files around will make rerunning MAKER much faster if it's ever necessary.
`TMP= #specify a directory other than the system default temporary directory for temporary files`	Many programs MAKER uses create temporary files, and some programs need fast I/O performance or non-NFS storage to run correctly. MAKER uses `/tmp` or whatever your system's temporary directory is by default; however, you may specify an alternate location. Never specify an NFS-mounted location, however, or MAKER will fail in a very ugly way.
`maker_bopts.ctl`	
`blast_type=ncbi+ #set to 'ncbi+', 'ncbi' or 'wublast'`	MAKER can use three of the major BLAST engines. Choosing a BLAST engine is more likely to be influenced by what flavor of BLAST is installed on the system rather than performance of one over the other.

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
maker_opts.ctl	
pcov_blastn=0.8 #Blastn Percent Coverage Threhold EST-Genome Alignments	Sets the required percent coverage (end-to-end) for an EST/mRNA-seq alignment to be maintained as evidence.
pid_blastn=0.85 #Blastn Percent Identity Threshold EST-Genome Aligments	Sets the required percent identity for an EST/mRNA-seq alignment to be maintained as evidence.
eval_blastn=1e-10 #Blastn eval cutoff	Sets the required BLAST e-value cutoff for an EST/mRNA-seq alignment to be maintained as evidence.
bit_blastn=40 #Blastn bit cutoff	Sets the required BLAST bit value cutoff for an EST/mRNA-seq alignment to be maintained as evidence.
depth_blastn=0 #Blastn depth cutoff (0 to disable cutoff)	Allows the user to limit the number of BLAST alignments that are kept and used for annotation. Setting this to a non-zero number will save memory and improve runtime for large evidence datasets.
pcov_blastx=0.5 #Blastx Percent Coverage Threhold Protein-Genome Alignments	These options are analogous to the BLASTN options above but are applied to protein evidence.
pid_blastx=0.4 #Blastx Percent Identity Threshold Protein-Genome Aligments	
eval_blastx=1e-06 #Blastx eval cutoff	
bit_blastx=30 #Blastx bit cutoff	
depth_blastx=0 #Blastx depth cutoff (0 to disable cutoff)	
pcov_tblastx=0.8 #tBlastx Percent Coverage Threhold alt-EST-Genome Alignments	These options are analogous to the BLASTN options above but are applied to alt_ests (EST/mRNA-seq data from a closely related species) evidence.
pid_tblastx=0.85 #tBlastx Percent Identity Threshold alt-EST-Genome Aligments	
eval_tblastx=1e-10 #tBlastx eval cutoff	
bit_tblastx=40 #tBlastx bit cutoff	
depth_tblastx=0 #tBlastx depth cutoff (0 to disable cutoff)	
pcov_rm_blastx=0.5 #Blastx Percent Coverage Threhold For Transposable Element Masking	These options are analogous to the BLASTN options above but are applied to transposon protein alignments used in repeat masking.

continued

Table 4.11.2 MAKER Control File Options, *continued*

Option	Comments
`maker_opts.ctl`	
`pid_rm_blastx=0.4 #Blastx Percent Identity Threshold For Transposbale Element Masking`	
`eval_rm_blastx=1e-06 #Blastx eval cutoff for transposable element masking`	
`bit_rm_blastx=30 #Blastx bit cutoff for transposable element masking`	
`ep_score_limit=20 #Exonerate protein percent of maximal score threshold`	Setting this higher will require polished protein to have a higher exonerate score to be maintained as evidence.
`en_score_limit=20 #Exonerate nucleotide percent of maximal score threshold`	Same as above but for Polished EST/mRNA-seq alignments.