Multi-Genome Annotation with AUGUSTUS

Stefanie Nachtweide and Mario Stanke
University of Greifswald, Institute of Mathematics and Computer Science,
Walther-Rathenau-Straße 47, 17487 Greifswald, Germany, Phone +49-(0)3834-420-4642, Fax +49-(0)3834-420-4640, E-Mail mario.stanke@uni-greifswald.de

Abstract

Comparing multiple related genomes can help to improve their structural annotation. The accuracy and consistency of the predicted exon-intron structures of the protein coding genes can be higher when considering all genomes at once rather than annotating one genome at a time.

The comparative gene prediction algorithm of AUGUSTUS performs such a multi-genome annotation. A multiple alignment of genomes is used to exploit evolutionary clues to conservation and negative selection. Further, AUGUSTUS exploits the fact that orthologous genes typically have congruent exon-intron structures. Comparative AUGUSTUS simultaneously predicts the genes in all input genomes. In this chapter we walk the reader through a small example from eight vertebrate species, including the construction of an alignment of the input genomes and how to integrate RNA-Seq evidence from multiple species for gene finding.

Keywords

comparative genomics, genome annotation, gene prediction, protein-coding genes, AUGUSTUS, RNA-Seq

1 Introduction

Like almost every other gene finding method, AUGUSTUS can be used to predict genes in a single genome only, or on many genomes sequentially, processing one genome at a time. We here refer to such methods as *single-genome gene finding*. Single-genome gene finding with AUGUSTUS is described in the articles [1, 2, 3, 4]. The book chapter [5] gives practical protocols for genome annotation with AUGUSTUS akin to the ones described here but only for single-genome gene finding. A brief but gentle introduction to the theory behind single-genome gene finding can be found in yet another book chapter [6].

In between single-genome and multi-genome gene prediction methods are methods that predict genes in a single target genome only, but that use one or multiple other genomes as informants. These methods exploit evolutionary hints to the coding gene structure, e.g. conservation patterns, that are derived from a multiple genome alignment. To this class of *comparative gene prediction* methods belong CONTRAST [7] and N-SCAN [8], for example. A principle difference between comparative single- and multi-genome gene prediction is, that the former does not consider the feasibility or plausibility of homologous gene structures in the related genomes.

In this chapter we describe the more recent multi-genome gene prediction extension of AUGUSTUS, as implemented in the comparative gene prediction (CGP) algorithm [9]. It exploits the fact that orthologous and often otherwise homologous genes have broadly conserved gene structures: The number of exons is often the same and the positions within the transcripts, where the introns were spliced out, are often conserved as well. The conservation of gene structures is more far-reaching than the conservation of genome sequences. Figure 1 shows an example from human, mouse and chicken.

Comparative AUGUSTUS gets a multiple genome alignment as input, which itself consists of a (large) set of local multiple alignments. It analyzes these local alignments and finds tuples of regions – at most one from each genome – that appear to be homologous regions. These homologous region tuples

are internally referred to as *gene ranges*. Typically, aligned genomes have undergone many large-scale genome rearrangement events, such that very many gene ranges need to be constructed. In Subheading 3.2.2 we will demonstrate how comparative multi-genome gene prediction can be performed in parallel by initially splitting up the multiple genome alignment. AUGUSTUS simultaneously predicts on each region of a gene range the genes by running the comparative gene prediction algorithm described in [9]. This algorithm does not assume a reference genome. It searches for each of the regions of the different genomes a gene structure. We refer to such a set of gene structures as *joint gene structure*. A joint gene structure is scored, such that gene structures are preferred that in each genome match the genome-specific evidence. At the same time, gene structures are preferred that allow a parsimonious explanation of differences in gene structures across genomes, if any. Further, comparative features from the multiple alignment are used to score candidate exons. These features include the ratio of nonsynonymous and synonymous mutation rates (dN/dS) as evidence for purifying selection of codons and sequence conservation.

The evidence from spliced alignments of RNA-Seq data to the respective native genome is integrated via influencing the score that gene structure candidates obtain in a single-genome model. Thus, evidence on the structure of a gene G in one species can indirectly lead to a correct prediction of the structure of a homolog of G in another species. In Subheading 3.5 commands for a multi-genome annotation are given based on RNA-Seq data for a subset of the species.

2 Installation and Requirements

AUGUSTUS is open source software and was developed and tested in a Linux environment. The most recent release can be obtained from the project web site `http://bioinf.uni-greifswald.de/augustus` with

Bash input

```
$ wget http://bioinf.uni-greifswald.de/augustus/binaries/augustus.current.tar.gz
```

The most recent development version can be obtained from GitHub with

Bash input

```
$ git clone https://github.com/Gaius-Augustus/Augustus.git
```

Gene prediction with AUGUSTUS can in principle be performed on a single workstation, sequentially or in parallel. However, a large total genome input size may practically render a compute cluster necessary. Expect that multi-species gene predictions with AUGUSTUS take in the order of 20 CPU core days per Gb of total genome (= the sum of all genome lengths) on hardware from 2017. This number has to be taken as a very rough guidance. The actual running time may easily deviate, say 2 or 3 fold, depending on the hardware, actual input and options specified to AUGUSTUS. The required drive space is usually dominated by the storage requirements of the input - the genomes and RNA-Seq alignments or other extrinsic evidence, if any. In addition, running time to prepare extrinsic evidence is required, in particular to align RNA-Seq reads to the target genomes.

Example data can be found in the folder `examples/cgp` of the AUGUSTUS package. This folder contains short genomic regions from four vertebrates (human, mouse, cow, chicken) such that AUGUSTUS can run in seconds on a laptop. Additional documentation can be found in the file `README-cgp.txt` of the AUGUSTUS package and in the AUGUSTUS comparative gene finding tutorial that is available as part of the AUGUSTUS package on GitHub as well as under `http://bioinf.uni-greifswald.de/augustus/binaries/tutorial-cgp/`.

To run AUGUSTUS in comparative gene prediction (CGP) mode, it must be compiled with appropriate options set in the file `common.mk` in the main directory of the AUGUSTUS package. In particular, COMPGENEPRED and a database option need to be enabled. For the purpose of running the examples in this chapter, the corresponding lines could look like this:

File contents example: `common.mk`

```
COMPGENEPRED = true
...
SQLITE = true
```

After editing this file with any editor, (re)compile `augustus` by issuing (see Note 1)

Bash input

```
$ make clean all
```

3 AUGUSTUS Comparative Gene Prediction

To allow a quick start we present in the next subsection an example with prepared input data that can be run very quickly. In the subsequent subheadings 3.2 - 3.7 we will treat different scenarios for whole-genome annotation tasks.

3.1 A Tiny Flat File Example

For the purpose of testing the executable and getting familiar with the file formats, we recommend to first run the example from `examples/cgp` that does not require any database:

Bash input

```
$ cd examples/cgp
$ augustus --species=human --speciesfilenames=genomes.tbl \
$          --treefile=tree.nwk --alnfile=aln.maf --/CompPred/outdir=out
```

The backslash at the end of a line merely breaks up long commands into multiple lines for better readability. If above command produces an error message in which you are asked to "recompile with flag COMPGENEPRED", it means that the `augustus` executable was not compiled with this option enabled or another version of the `augustus` binary is also installed. In the latter case, you can consider specifying the full path to the binary rather than just calling '`augustus`'.

3.1.1 Input

With the mandatory parameter `species` a pre-trained parameter set is specified. The available species options are the subdirectory names under `config/species`. Note that a single parameter set is used for all aligned input genomes. We recommend to choose a species that best represents the whole clade, e.g. human in a vertebrate clade. The mandatory parameter `speciesfilenames` specifies the name of a tab-separated text file that contains a table with two columns. The first column contains identifiers for genomes or species. The second column holds the file names (including the relative paths) for the corresponding genome files. The genome files must be in FASTA format and may contain the sequences of multiple chromosomes/scaffolds.

File contents example: `genomes.tbl`

```
hg19     human.fa
mm9      mouse.fa
bosTau4  cow.fa
galGal3  chicken.fa
```

The mandatory parameter `treefile` specifies a phylogenetic tree of the species in Newick format, e.g.

File contents example: `tree.nwk`

```
(((hg19:0.163,mm9:0.353):0.021,bosTau4:0.219):0.438,galGal3:0.474);
```

All branch lengths are required and leaf nodes must be named as in `genomes.tbl`. Another valid format is

File contents example: `tree-alt.nwk`

```
begin trees;
        translate
                1       hg19,
                2       mm9,
                3       bosTau4,
                4       galGal3
                ;
tree con_50_majrule = [&U] (((1:0.163,2:0.353):0.021,3:0.219):0.438,4:0.474);
end;
```

The branch lengths in the tree should be the *expected number of codon mutations* in coding regions. Consequently, for closely related species, the sum of branch lengths of edges on the path from leaf A to leaf B in the tree is approximately the fraction of codons that are different in alignments of genes between genomes A and B. Frequently, phylogenetic trees are scaled differently, e.g. with respect to nucleotide or amino acid mutations. In such cases, and when a codon alignment is available, the alignment can be used to estimate the phylogenetic codon distance between any pair of species. The tree can then be rescaled internally using the parameter `/CompPred/scale_codontree`. Comparative AUGUSTUS is relatively robust with regards to wrong trees [10]. In cases where the phylogeny is not known, a star-like tree with uniform branch lengths might be used instead, e.g. `(hg19:0.01, mm9:0.01, bosTau4:0.01, galGal3:0.01);` which can be interpreted as consensus-finding on the ancestral exon and sequence states.

The mandatory parameter `alnfile` specifies a genome alignment file in MAF format. Such a file typically lists many short local alignments, each of different regions of the input genomes.

File contents example: `aln.maf`

```
a score=235085.000000
s hg19.chr21                      15725769 27 +  48129895 AGCTATTGCTGTTTATGTCTCAATTTC
s mm9.chr16                       75744509 27 +  98319150 AGCTCGCAGTGTTGATGCTTCAGTCTC
s bosTau4.chr1                   138520043 27 - 161106243 AGCTATTGATGTTTATGTCTTCATTTC
s galGal3.chr1                   101466793 21 + 200994015 AGCTCGAGAAG------AGCCATTATA

a score=128487.000000
s hg19.chr21                      15725796 32 +  48129895 CCAGAGGAGAGGGTTAGTACCAAATGCACCAA
s bosTau4.chr1                   138520070 30 - 161106243 CCAGAGGAGA--GTTCATATTGAGTGCACCAA
s mm9.chr16                       75744536 30 +  98319150 TCAGAGAAGA--ACTTGGACAAAGTGCACCCA
```

The sequence names (first field in an 's' line, up to the dot) must again be the genome identifiers as they appear in `genomes.tbl`. In Subheading 3.2 we describe how to obtain such a `.maf` formatted multiple genome alignment. Alignment rows of species that are not listed in `speciesfilenames` are ignored for convenience, such that the same multiple alignment can be used when annotating only a subset of the aligned genomes.

For running comparative AUGUSTUS on a target subset of genomes, simply delete all lines of non-target genomes in the file `genomes.tbl`. Neither the alignment nor the phylogenetic tree need modification if only a subset of genomes is used.

3.1.2 Output

The optional parameter `/CompPred/outdir` specifies an output directory. This is particularly useful when many comparative gene predictions are performed in parallel as done below. Upon completion

the output directory contains for each genome a separate file with the genome annotation in general feature format (GFF).

Bash output

```
> ls out
bosTau4.cgp.gff  galGal3.cgp.gff  hg19.cgp.gff  mm9.cgp.gff
```

A cross-species comparison of gene sets, e.g. to find orthologous genes and to compare their gene structures, can be performed with the tool `homGeneMapping`. For this, we refer the reader to step 6 of the AUGUSTUS CGP tutorial.

Above output only contains comparative gene predictions in regions of the genomes that are at least partially aligned to regions of other genomes. If whole-genome annotations are sought, one has to keep in mind that some genome regions may be unique to a genome or not be aligned otherwise. We therefore recommend to combine the comparative annotation with single-genome predictions with the tool `joingenes` as described in Subheading 3.6 below. The GFF output may have to be further processed depending on the aim that is pursued. Further processing is largely independent on the annotation method, however. E.g. it can be visualized in a browser or converted to another format, e.g. to GFF3 format:

Bash input

```
$ cd out
$ for s in bosTau4 galGal3 hg19 mm9;
$ do
$   gtf2gff.pl < $s.cgp.gff --gff3 --out=$s.cgp.gff3
$ done
```

In the following we will describe the practical steps of a multi-genome annotation in detail.

3.2 Creating a Whole-Genome Alignment

In some cases a whole-genome alignment may already be available to you. An important example is the multiple alignment of vertebrates by the UCSC Genome Browser group performed with MULTIZ [11], which is downloadable in MAF format from the UCSC download server [`http://hgdownload.soe.ucsc.edu/goldenPath/hg38/multiz100way/`]. In other cases, you may have to perform the alignment yourself. For this, we recommend the alignment program CACTUS [12]. From now on we use the example data from the folder `docs/tutorial-cgp` of the AUGUSTUS package (see Table 1). This directory includes a tutorial in HTML as well as example input and some output data. The example contains data from 8 vertebrates, which is still small but large enough to demonstrate the parallelization that would be appropriate for realistically-sized input. You may skip steps producing intermediate data by using the respective data from the tutorial, e.g. the alignment generation can be skipped by copying the folder `mafs/` from the `tutorial-cgp/results` into the `tutorial-cgp/data` directory.

3.2.1 Running the Aligner CACTUS

First, install `progressiveCactus` from GitHub or by following the respective installation instructions on `docs/tutorial-cgp/index.html`. Prepare the required input file for the CACTUS aligner, a text file with the species tree (Newick format) followed by a space-separated list of species names and location of the corresponding genome FASTA files (see Note 2). The commands

Bash input

```
$ cp tree.nwk vertebrates.txt
$ for f in $PWD/genomes/*.fa; do echo -ne "$(basename $f .fa)\t$f\n"; done >>vertebrates.txt
```

will produce a suitable file like this:

File contents example: `vertebrates.txt`

```
((monDom5:0.340786,(((hg38:0.035974,rheMac3:0.043601):0.109934,
  (mm10:0.084509,rn6:0.091589):0.271974):0.020593,(bosTau8:0.18908,
  canFam3:0.13303):0.032898):0.258392):0.181168,galGal4:0.559442);
bosTau8   /home/mario/Augustus/docs/tutorial-cgp/data/genomes/bosTau8.fa
canFam3   /home/mario/Augustus/docs/tutorial-cgp/data/genomes/canFam3.fa
galGal4   /home/mario/Augustus/docs/tutorial-cgp/data/genomes/galGal4.fa
hg38      /home/mario/Augustus/docs/tutorial-cgp/data/genomes/hg38.fa
mm10      /home/mario/Augustus/docs/tutorial-cgp/data/genomes/mm10.fa
monDom5   /home/mario/Augustus/docs/tutorial-cgp/data/genomes/monDom5.fa
rheMac3   /home/mario/Augustus/docs/tutorial-cgp/data/genomes/rheMac3.fa
rn6       /home/mario/Augustus/docs/tutorial-cgp/data/genomes/rn6.fa
```

Note that absolute path names are required. Now, the actual alignment can be performed with:

Bash input

```
$ runProgressiveCactus.sh vertebrates.txt cactusout\
$    vertebrates.hal --maxThreads=4 2>&1 > cactus.out
```

This command outputs a binary alignment file `vertebrates.hal`. For the example data it requires about 10 minutes. On large data, this command can be expected to be time-consuming and it may require to be run on a fast machine with many cores (increase `maxThreads` accordingly).

3.2.2 Exporting a HAL Alignment to MAF Alignments

The file `vertebrates.hal` contains the complete whole-genome multiple alignment. In order to allow for independent parallel executions of the gene prediction step, we recommend to split the global alignment into several overlapping alignment chunks and to simultaneously convert the chunks to MAF format:

Bash input

```
$ hal2maf_split.pl --halfile vertebrates.hal --refGenome hg38 \
$    --cpus 4 --chunksize 50000 --overlap 25000 --outdir mafs
```

`hal2maf_split.pl` builds upon the HAL tools, which is also part of the `progressiveCactus` package. Above script uses a single reference genome among the aligned genomes (here the human hg38) as a guide to split the alignment. The reference genome is conceptually divided into regions of size `chunksize` such that neighboring regions overlap by `overlap` base pairs. Then for each such region the local alignments of the input alignment for the given region are exported into one MAF alignment. To avoid undersized chunks on which only partial genes are predicted, we recommend to choose a reference species that has a high-quality, near-complete genome assembly with long scaffolds, if available. A sensible setting for `chunksize` in real data could be `1000000`. With a higher value each process requires more RAM and a larger run time. Above splitting command also introduces some fraction of truncated genes even when `chunksize` is large. In such cases and where truncated genes cannot be patched up with `joingenes` (see Subheading 3.4 below), choose a larger `overlap`. If the `progressiveCactus/submodules/hal/bin` directory is not in your global python path, use the parameter `hal_exec_dir` of `hal2maf_split.pl` to point to the directory that contains `hal2maf`.

This will generate the directory `mafs/` that contains the output alignment chunks in MAF format:

Bash output

```
> ls -1 mafs
chr16.0-49999.maf
chr16.100000-149999.maf
chr16.125000-174999.maf
chr16.150000-199999.maf
chr16.175000-210154.maf
chr16.25000-74999.maf
chr16.50000-99999.maf
chr16.75000-124999.maf
```

3.3 Loading Genomes into an SQLite Database

In *multi*-genome gene prediction, the genomes cannot all be processed in linear order. As all genomes together are often too large to be stored simultaneously in memory (as done in the example in Subheading 3.1), we implemented a database solution to obtain so-called *random access* to the regions. A single database for all species contains indices which allow to efficiently retrieve only those regions that are needed at the time or by the respective parallel job. SQLite3 and MySQL are supported. To avoid redundancy, the SQLite3 solution does not store the genomes themselves in the database, but only byte offsets into the respective FASTA genome files. We recommend the SQLite3 solution over the MySQL solution (see Note 3).

First, create a table of all genome names and sequence files (as for parameter `speciesfilenames` of `augustus`):

Bash input

```
$ for f in $PWD/genomes/*.fa; do echo -ne "$(basename $f .fa)\t$f\n"; done >genomes.tbl
```

With the example data from the tutorial, the generated `genomes.tbl` will be a table with 8 genomes. Now, load the genomes into an SQLite database

Bash input

```
$ while read line
$ do
$   species=$(echo "$line" | cut -f 1)
$   genome=$(echo "$line" | cut -f 2)
$   load2sqlitedb --noIdx --species=$species --dbaccess=vertebrates.db $genome
$ done < genomes.tbl
$ load2sqlitedb --makeIdx --dbaccess=vertebrates.db
```

This loop loads each genome into the database. It then creates indices on the tables that allow to access regions quickly. When genome regions are requested in subsequent steps, only small parts of `vertebrates.db` and the respective genomes files need to be read. `vertebrates.db` is now a (flat file) database that contains offsets into the 8 genomes.

You can check if loading was successful with following database query:

Bash input

```
$ sqlite3 -header -column vertebrates.db "\
$  SELECT speciesname, \
$   sum(end-start+1) AS 'genome length',\
$   count(*) AS '# chunks',\
$   count(distinct seqnr) AS '# seqs'\
$  FROM genomes natural join speciesnames\
$  GROUP BY speciesname;"
```

It returns a summary of the genomes in the database (see Note 4):

Bash output

```
speciesname  genome length  # chunks    # seqs
-----------  -------------  ----------  ----------
bosTau8      156091         4           1
canFam3      184728         4           1
galGal4      149999         3           1
hg38         210155         5           1
mm10         178393         4           1
monDom5      540519         11          1
rheMac3      220640         5           1
rn6          99944          2           1
```

Check if all genomes are in the database and the number of sequences and total genome size for each genome is correct.

3.4 *De Novo* Comparative Gene Finding

We next demonstrate an application, where only the naked genomes are available ('*de novo*' gene finding). A more typical example with RNA-Seq follows in the next section.

Create a new folder for the *de novo* experiments and therein softlinks to the MAF files.

Bash input

```
$ mkdir augCGP_denovo
$ cd augCGP_denovo
$ num=1
$ for f in ../mafs/*.maf; do ln -s $f $num.maf; ((num++)); done
```

Run comparative AUGUSTUS in so-called CGP mode on all alignment chunks in parallel.

Bash input

```
$ for ali in *.maf
$ do
$ id=${ali%.maf} # this will remove .maf suffix
$ augustus \
$   --species=human \
$   --softmasking=1 \
$   --treefile=../tree.nwk \
$   --alnfile=$ali \
$   --dbaccess=../vertebrates.db \
$   --speciesfilenames=../genomes.tbl \
$   --/CompPred/outdir=pred$id > aug$id.out 2> err$id.out &
$ done
```

This command starts one AUGUSTUS process for each of the 8 alignment files in the background using `&` and may take a few minutes (see Note 5). This simple parallelization approach is only for demonstration purposes. In real applications with several hundreds or thousands of alignment chunks, we recommend to run parallel jobs on a compute cluster. Set the option `-softmasking=1` in cases where the genomes are soft-masked.

This will generate the folders `pred1/`, ..., `pred8/`, one for each alignment chunk, that each contain GFF files with gene predictions for each input genome.

Bash output

```
> ls pred1/
bosTau8.cgp.gff  canFam3.cgp.gff  galGal4.cgp.gff  hg38.cgp.gff
mm10.cgp.gff  monDom5.cgp.gff  rheMac3.cgp.gff  rn6.cgp.gff
```

Merge gene predictions from parallel runs with

Bash input

```
$ mkdir joined_pred
$ while read line
$ do
$   species=$(echo "$line" | cut -f 1)
$   find pred* -name "${species}.cgp.gff" >${species}_gtfs.lst;
$   joingenes -f ${species}_gtfs.lst -o joined_pred/$species.gff
$ done < ../genomes.tbl
```

This will create the folder `joined_pred/` with the final gene predictions for each input genome in a single file.

3.5 RNA-Seq Based Comparative Gene Finding

We here demonstrate how RNA-Seq data can be incorporated into comparative AUGUSTUS. In general, the same types of extrinsic evidence can be incorporated as in single-species gene finding with AUGUSTUS (including RNA-Seq, cDNA, ESTs, protein sequences, etc). In the CGP mode, each piece of evidence is specific to a genome. The evidence can be incorporated for each genome or for any subset of genomes.

Note that RNA-Seq from different genomes or species can complement each other, e.g. tissues or conditions that were sampled in one species can also result in an improved accuracy for the genes expressed in that tissue or under that condition in other species. We recommend that RNA-Seq should only be aligned to the native genome of the RNA-Seq sample, or to the closest genome, otherwise, and not also to other genomes. On the one hand, the alignment to more than one genome is unnecessary, as evidence is shared between genomes. On the other hand, the alignment of RNA-Seq from one species to another species' genome is more error-prone.

RNA-Seq evidence is generated from the spliced alignments of individual reads against the respective genome. A typical sequence of steps could be to execute an aligner (e.g. STAR [13]) and then the quality filtering of alignments with `filterBAM`, the conversion to an AUGUSTUS-specific hints file with `bam2hints` for intron hints and `wig2hints.pl` for so-called exonpart hints. This step is not specific to applying *comparative* AUGUSTUS and e.g. described in the AUGUSTUS Wiki [`http://bioinf.uni-greifswald.de/bioinf/wiki/pmwiki.php?n=Augustus.Augustus`] and chapter [5]. We here assume that a so-called *hints* file for each genome is already available that summarizes the extrinsic evidence. The hints for this tutorial are in `tutorial-cgp/data/hints/`.

3.5.1 Loading RNA-Seq Hints into the SQLite Database

In this example, intron and exonpart (ep) hints for a subset of four of the eight species (human, mouse, chicken and macaque) are provided in the `hints` subdirectory. Prepare a text file with a list of species names and location of the corresponding hints files.

Bash input

```
$ for f in $PWD/hints/*.gff; do echo -ne "$(basename $f .hints.gff)\t$f\n"; done > hints.tbl
```

The file `hints.tbl` will now look similar to this:

File contents example: `hints.tbl`

```
galGal4 /home/mario/Augustus/docs/tutorial-cgp/data/hints/galGal4.hints.gff
hg38    /home/mario/Augustus/docs/tutorial-cgp/data/hints/hg38.hints.gff
mm10    /home/mario/Augustus/docs/tutorial-cgp/data/hints/mm10.hints.gff
rheMac3 /home/mario/Augustus/docs/tutorial-cgp/data/hints/rheMac3.hints.gff
```

Load the hints into the database `vertebrates.db` from above. You may want to make a backup copy of the database first. The backup is useful if you want to add different sets of hints to the same genome assemblies.

Bash input

```
$ while read line
$ do
$   species=$(echo "$line" | cut -f 1)
$   hints=$(echo "$line" | cut -f 2)
$   load2sqlitedb --noIdx --species=$species --dbaccess=vertebrates.db $hints
$ done < hints.tbl
$ load2sqlitedb --makeIdx --dbaccess=vertebrates.db
```

Check if loading was successful and the content is plausible with following database query:

Bash input

```
$ sqlite3 -header -column vertebrates.db "\
$   SELECT count(*) AS '#hints',typename,speciesname\
$   FROM (hints as H join featuretypes as F on H.type=F.typeid)\
$         natural join speciesnames\
$   GROUP BY speciesid,typename;"
```

This returns a summary of how many hints of each type are in the database for each species (see Note 6):

Bash output

```
#hints      typename    speciesname
----------  ----------  -----------
3368        exonpart    galGal4
129         intron      galGal4
7905        exonpart    hg38
267         intron      hg38
7930        exonpart    mm10
378         intron      mm10
11050       exonpart    rheMac3
265         intron      rheMac3
```

3.5.2 Preparing an Extrinsic Configuration File

Extrinsic evidence can be more or less trustworthy depending on the source. E.g. an intron present in a reference annotation of one of the genomes may be trusted completely, while an intron inferred from RNA-Seq alignments or a spliced alignment of a protein homolog has some chance to be wrong. Such parameters are set in a text file we refer to as *extrinsic config file*. Start its creation by copying an existing extrinsic config file:

Bash input

```
$ cp ${AUGUSTUS_CONFIG_PATH}/extrinsic/extrinsic-cgp.cfg extrinsic-rnaseq.cfg
```

Open `extrinsic-rnaseq.cfg` file with a text editor, go to the first `[GROUP]` section and replace the next line

File contents example: `extrinsic-rnaseq.cfg`

```
[GROUP] # replace 'none' by the names of genomes with src=W and src=E hints in the database
none
```

as instructed by the space-separated list of names of genomes with RNA-Seq hints, i.e.

File contents example: `extrinsic-rnaseq.cfg`

```
[GROUP]
hg38 mm10 rheMac3 galGal4
```

In comparative mode of AUGUSTUS, hints can be integrated for multiple species. The configuration file allows to specify the extrinsic parameters individually for each species or – often more conveniently – for groups of species. For example, for two genomes hints from existing annotations may be available, one more trustworthy than the other. Another genome may have RNA-Seq evidence and for yet another genome no evidence may be available. For instructions on changing specific parameters we refer the reader to the bottom of the files `extrinsic.cfg` and `extrinsic-cgp.cfg` in the folder `config/extrinsic` of the AUGUSTUS package or to the chapter on single-genome gene prediction with AUGUSTUS [5].

3.5.3 Running AUGUSTUS-CGP with RNA-Seq hints

Create a new folder for these experiments and switch to the new directory

Bash input

```
$ mkdir augCGP_rnaseq
$ cd augCGP_rnaseq
$ # create here softlinks to the alignment chunks for convenience
$ num=1; for f in ../mafs/*.maf; do ln -s $f $num.maf; ((num++)); done
```

Next, run comparative AUGUSTUS in parallel on the alignment chunks:

Bash input

```
$ for ali in *.maf
$ do
$  id=${ali%.maf} # remove .maf suffix
$  augustus \
$   --species=human \
$   --softmasking=1 \
$   --treefile=../tree.nwk \
$   --alnfile=$ali \
$   --dbaccess=../vertebrates.db \
$   --speciesfilenames=../genomes.tbl \
$   --alternatives-from-evidence=0 \
$   --dbhints=1 \
$   --UTR=1 \
$   --allow_hinted_splicesites=atac \
$   --extrinsicCfgFile=../extrinsic-rnaseq.cfg \
$   --/CompPred/outdir=pred$id > aug$id.out 2> err$id.out &
done
```

The option `UTR=1` enables the model for untranslated regions and is recommended whenever 'exon-part' hints are incorporated. `dbhints=1` enables the retrieval of hints from the database. The option `allow_hinted_splicesites=atac` enables the prediction of the rare AT-AC splice sites, when evidenced by hints. This is in addition to the default GT-AG and GC-AG splice sites (first and last two intronic bases). Above command will generate a folder for each alignment chunk that contains GFF files with gene predictions for each input genome. Finally, merge the gene predictions from parallel runs with the command in the last box from Subheading 3.4.

3.6 Joining with Single-Genome Gene Predictions

Comparative AUGUSTUS delivers in principle an incomplete genome annotation because it only annotates significantly alignable regions. Other genome regions, e.g. genome regions that are unique

to one input genome, are not annotated. In addition, the gene ranges can sometimes break up genes, e.g. when some genome assemblies are fragmented or wrong. Therefore, we recommend to supplement the comparative annotation with a single-genome annotation. For demonstration purposes, we run regular AUGUSTUS independently on each genome, using RNA-Seq where available:

Bash input

```
$ mkdir aug_rnaseq
$ cd aug_rnaseq
$ for assembly in bosTau8 canFam3 monDom5 rn6; do
$   # make ab initio predictions for genomes without hints
$   augustus --species=human ../genomes/$assembly.fa --softmasking=1 > $assembly.gff &
$ done
$ for assembly in hg38 mm10 rheMac3 galGal4; do
$   # make RNA-Seq based predictions on other genomes
$   augustus --species=human ../genomes/$assembly.fa --hintsfile=../hints/$assembly.hints.gff \
$       --UTR=on --allow_hinted_splicesites=atac --extrinsicCfgFile=extrinsic.M.RM.E.W.cfg \
$       --softmasking=on --alternatives-from-evidence=off > $assembly.gff &
done
```

If `extrinsicCfgFile` is not a path to a file, a file with that name, here `extrinsic.M.RM.E.W.cfg`, is searched in the `config/extrinsic` folder. Now, we have for each genome two annotations, the comparative and the non-comparative, which we merge using `joingenes`. When doing this, we give the comparative annotation a higher priority. In the `data` directory issue

Bash input

```
$ mkdir aug_joined
$ cd aug_joined
$ for assembly in hg38 mm10 rheMac3 galGal4 bosTau8 canFam3 monDom5 rn6; do
$   joingenes -g ../augCGP_rnaseq/joined_pred/$assembly.gff,../aug_rnaseq/$assembly.gff \
$      --priorities=2,1 --output=$assembly.jg.gff
done
```

`joingenes` does not only select the higher priority transcripts – here comparative – from conflicting transcript versions, but also sometimes extends comparatively predicted transcripts that appear to be truncated using overlapping predictions of the same gene set or from single-genome predictions.

3.7 Annotation Mapping

An important special case of comparative gene prediction with extrinsic evidence is the setting of annotation mapping. Here, one or several genomes already have (a) trusted annotation(s), while further, usually new genomes of related species require annotation. The existing annotations shall be leveraged so that ortholog gene structures of previously annotated genes shall be lifted or *mapped* to the other genomes, where possible. This can be done with comparative AUGUSTUS using a variant of the protocol in Subheading 3.5 where RNA-Seq evidence is replaced with the evidence from the existing annotation(s). We refer the reader to Exercise 4 of the CGP tutorial in the AUGUSTUS package. Such an approach has recently been performed when annotating 16 *de novo* assembled mouse strains [14] and is also implemented in the comparative annotation toolkit CAT [15].

4 Notes

1. If issues with the installation or basic running arise, first consult the instructions in the files `README-cgp.txt` and `README.TXT`. Problems that appear to result from a bug (e.g. a segmentation fault) can be reported on the GitHub page [`https://github.com/Gaius-Augustus/Augustus`].

2. If your clade is very diverse, such that a whole-genome alignment is difficult and very gappy, e.g. when considering divergent vertebrates species, then it may be preferable to run comparative AUGUSTUS (if necessary several times) on less diverse subclades, e.g. on mammals or primates only. If you believe that a whole-genome-duplication has taken place in some genomes with regards to others in your data set, then use a duplicated genome as reference for `hal2maf_split.pl`. Otherwise, some homology information could be systematically ignored in comparative AUGUSTUS.
3. If `sqlite3` is not available on your system, use the alternative with MySQL. However, we have experienced performance issues with MySQL, when accessing the database from ∼1000 comparative AUGUSTUS processes simultaneously, not so with the `sqlite3` file-based database.
4. Make sure that genome versions and genome names match. If an error "failed retrieving sequence" occurs, a reason could be that an assembly version in the database is not identical to the one that was aligned. In such a case the program `getSeq` that retrieves sequence ranges from the database can be helpful to find the discordance. If unsure, you may want to reimport the correct genome assemblies.
5. If some jobs require too much memory for the computing nodes, consider decreasing the parameter `chunksize` of `hal2maf_split.pl`, or, rerun the 'stragglers' on a node with more memory.
6. Sometimes individual RNA-Seq libraries can worsen the predictions when included in addition to all the other available RNA-Seq libraries, e.g. if not poly-A selected. Such problematic libraries could be identified using a genome browser and then excluded before running AUGUSTUS again.

5 Acknowledgement

This chapter is based on research that was funded partially by Deutsche Forschungsgemeinschaft grant STA 1009/10-1 to MS and by a scholarship of the Studienstiftung des deutschen Volkes to SN.

6 References

[1] M. Stanke and S. Waack. Gene prediction with a hidden Markov model and new intron submodel. *Bioinformatics*, 19 Suppl. 2:ii215–ii225, 2003.

[2] M. Stanke, M. Diekhans, R. Baertsch, and D. Haussler. Using native and syntenically mapped cDNA alignments to improve *de novo* gene finding. *Bioinformatics*, 24(5):637–644, 2008.

[3] O. Keller, M. Kollmar, M. Stanke, and S. Waack. A novel hybrid gene prediction method employing protein multiple sequence alignments. *Bioinformatics*, 27(6):757–763, 2011.

[4] K.J. Hoff and M. Stanke. WebAUGUSTUS – a web service for training AUGUSTUS and predicting genes in eukaryotes. *Nucleic Acids Research*, 41(W1):W123–W128, 2013.

[5] K.J. Hoff and M. Stanke. Predicting genes in single genomes with AUGUSTUS. *Current Protocols in Bioinformatics*, 2018. accepted.

[6] M. Stanke and M. Borodovsky. *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins*, chapter Genome Structural Annotation. Wiley, 4th edition, 2018. submitted.

[7] S. Gross, C. Do, M. Sirota, and S. Batzoglou. CONTRAST: a discriminative, phylogeny-free approach to multiple informant de novo gene prediction. *Genome Biology*, 8(12):R269, 2007.

[8] Samuel S Gross and Michael R Brent. Using multiple alignments to improve gene prediction. *Journal of computational biology*, 13(2):379–393, 2006.

[9] Stefanie König, Lars W. Romoth, Lizzy Gerischer, and Mario Stanke. Simultaneous gene finding in multiple genomes. *Bioinformatics*, 32(22):3388–3395, 2016.

[10] Stefanie Nachtweide. *The Simultaneous Identification of Genes in Related Species*. doctoral thesis, 2018.

[11] Kate R Rosenbloom, Joel Armstrong, Galt P Barber, Jonathan Casper, Hiram Clawson, Mark Diekhans, Timothy R Dreszer, Pauline A Fujita, Luvina Guruvadoo, Maximilian Haeussler, et al. The ucsc genome browser database: 2015 update. *Nucleic Acids Research*, 43(D1):D670–D681, 2014.

[12] Benedict Paten, Dent Earl, Ngan Nguyen, Mark Diekhans, Daniel Zerbino, and David Haussler. Cactus: Algorithms for genome multiple sequence alignment. *Genome Research*, 21(9):1512–1528, 2011.

[13] A. Dobin, C.A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, and T.R. Gingeras. STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, 29(1):15–21, 2013.

[14] Jingtao Lilue, Anthony G Doran, Ian T Fiddes, Monica Abrudan, Joel Armstrong, Ruth Bennett, William Chow, Joanna Collins, Stephan Collins, Anne Czechanski, Petr Danecek, Mark Diekhans, Dirk-Dominic Dolle, Matt Dunn, Richard Durbin, Dent Earl, Anne Ferguson-Smith, Paul Flicek, Jonathan Flint, Adam Frankish, Beiyuan Fu, Mark Gerstein, James Gilbert, Leo Goodstadt, Jennifer Harrow, Kerstin Howe, Mikhail Kolmogorov, Stefanie Koenig, Chris Lelliott, Jane Loveland, Richard Mott, Paul Muir, Fabio Navarro, Duncan Odom, Naomi Park, Sarah Pelan, Son K Phan, Michael Quail, Laura Reinholdt, Lars Romoth, Lesley Shirley, Cristina Sisu, Marcela Sjoberg-Herrera, Mario Stanke, Charles Steward, Mark Thomas, Glen Threadgold, David Thybert, James Torrance, Kim Wong, Jonathan Wood, Fengtang Yang, David J Adams, Benedict Paten, and Thomas M Keane. Sixteen diverse laboratory mouse reference genomes define strain-specific haplotypes and novel functional loci. *Nature Genetics*, 2018.

[15] Ian T. Fiddes, Joel Armstrong, Mark Diekhans, Stefanie Nachtweide, Zev N. Kronenberg, Jason G. Underwood, David Gordon, Dent Earl, Thomas Keane, Evan E. Eichler, David Haussler, Mario Stanke, and Benedict Paten. Comparative Annotation Toolkit (CAT) – simultaneous clade and personal genome annotation. *Genome Research*, 2018. doi:10.1101/gr.233460.117.

7 Tables

species	assembly	genomic region
human	hg38	chr16:186964-397118
mouse	mm10	chr17:26104939-26283331
rat	rn6	chr10:15470071-15570014
cow	bosTau8	chr25:224163-380253
dog	canFam3	chr6:40126702-40311429
rhesus	rheMac3	chr20:149129-369768
rabbit	monDom5	chr6:149454308-149994826
chicken	galGal4	chr14:12108253-12258251

Table 1: The example data set covers a 2 megabase syntenic region in 8 vertebrates.

8 Figures

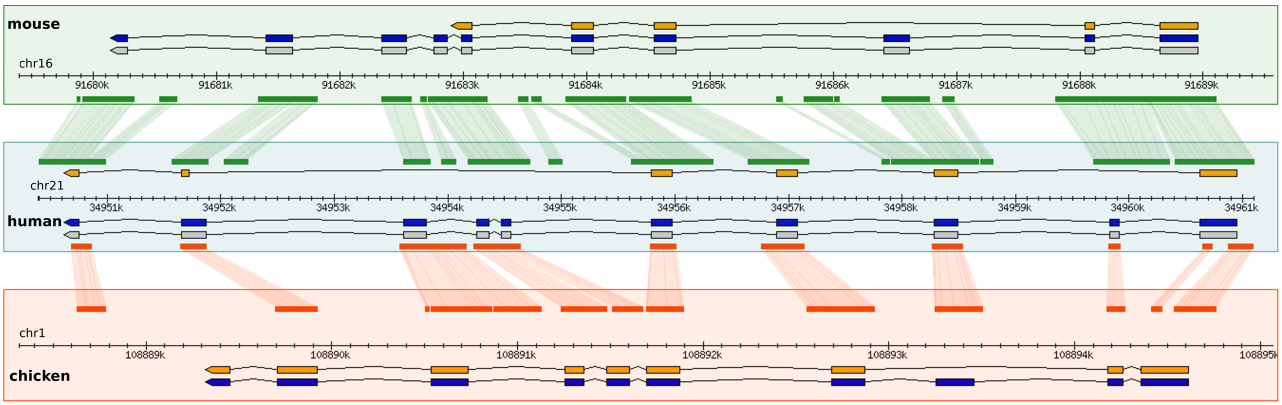

Figure 1: Homologous genes from mouse, human and chicken with alignable genome regions (green and red trapezoids). The gray gene structure in mouse and human is from RefSeq. The orange gene structure is from an *ab initio* single-genome prediction with AUGUSTUS. The blue structures were predicted *de novo* with comparative AUGUSTUS, and are more consistent and more accurate in this example.