ExonHunter — version 1.06
Installation and User Guide

Broňa Brejová, Tomáš Vinař

{brejova,vinar}@fmph.uniba.sk

Contents

1 Introduction 1

2 Pre-installation requirements 1

3 Installing and configuring ExonHunter 2
3.1 Downloading and installing files 2
3.2 Adapting eh.config 2
3.2.1 Global options 3
3.2.2 Evidence sources 3
3.2.3 Configuration of tasks 4
3.2.4 Paths to programs 4
3.2.5 Pipeline-specific parameters 5

4 Using ExonHunter 5
4.1 ExonHunter as ab initio gene finder 5
4.2 ExonHunter with repeat masking 5
4.3 Other sources of information 5
4.4 Custom sources of information 6

5 Problems and limitations 6
5.1 Temporary files and input sizes 6
5.2 Privileges to data files 7

6 Contributors 7

1 Introduction

ExonHunter is a eukaryotic gene finder that can use multiple sources of evidence to improve prediction accuracy. If you are using results from ExonHunter in a publication, please cite:

- B. Brejova, D. G. Brown, M. Li, T. Vinar. ExonHunter: A Comprehensive Approach to Gene Finding. Bioinformatics, 21(Suppl 1):i57-i65;. June 2005. ISMB 2005.

This program is distributed free of charge in a belief that it will be useful. Please, review the license agreement distributed with the package in a file license.txt. By using the program, you agree to the terms of the license agreement. If this license does not fit your needs, please contact us for alternate arrangements.

2 Pre-installation requirements

ExonHunter can be used as a standalone program to produce ab initio gene predictions on both masked and unmasked sequences. Perl (version at least 5.8) is required for supporting scripts.

The ExonHunter predictions improve significantly if additional evidence is used to support gene predictions. Pipeline collecting and supporting such additional evidence requires the following programs to be installed:

- **Common sequence repeats.** RepeatMasker (`http://www.repeatmasker.org/`) and corresponding repeat libraries from GIRI (`http://www.girinst.org/repbase`). RepeatMasker also requires either Cross Match (`http://www.phrap.org/`) or WUBlast (`http://blast.wustl.edu/`).
- **Expressed sequence tags.** BLAT (`http://genome.ucsc.edu/`). The main advantage of BLAT is the speed of aligning and the ability to perform protein alignments too.

  As an alternative you could use the SIM4 alignment program (`http://www.bx.psu.edu/miller_lab`). SIM4 is very slow. To speed up the alignments, you will need either PatternHunter homology search program (`http://www.bioinformaticssolutions.com/products/ph`) or WUBlast (`http://blast.wustl.edu/`).

  You will also need EST libraries in FASTA format for your organism and for other related organisms (see for example DFCI gene indices `http://compbio.dfci.harvard.edu/tgi/`).
- **Protein alignments.** BLAT (`http://genome.ucsc.edu`). The main advantage of BLAT is the speed of aligning and the ability to perform EST alignments too.

  A slower alternative is NCBI BLAST (`http://www.ncbi.nlm.nih.gov/BLAST`).

  You will also need a database of known proteins in FASTA format; you can obtain a curated database from SWISSPROT.
- **PET pairs.** Experimental support for PET pairs is included. Homology search program PatternHunter (`www.bioinformaticssolutions.com/products/ph`) is required.

3 Installing and configuring ExonHunter

3.1 Downloading and installing files

1. Download the file exonhunter-1.06.tgz and uncompress it.

   ```
   tar xzf exonhunter-1.06.tgz
   ```

   This will create an installation directory exonhunter-1.06
2. Change to the installation directory and run installation script as follows:

   ```
   make install
   ```

   By default, the script will install ExonHunter in subdirectories of /usr/local. This usually requires root privileges. You can change the destination by setting one or more of the following variables:

   prefix: use different directory instead of /usr/local; all other variables are set relative to the prefix, unless overridden by the user

   bindir: directory for user executables (default: $prefix/bin/)

   libdir: additional files required by ExonHunter (default: $prefix/lib/exonhunter)

   docdir: documentation (default: $prefix/share/doc/exonhunter)

   paramdir: parameter files (default: $prefix/share/exonhunter/param)

For example, `make install prefix=~/eh` will install ExonHunter files in subdirectory eh of your home directory. The user executables are placed in ~/eh/bin, and the directory for parameter files is ~/eh/share/exonhunter/param.

3. Make sure that user executables are included in your path; by default, executables are placed in /usr/local/bin and are included in the system path.

4. If everything went well, you should be able to run executables: `exonhunter`, and `prepare-evidence`; running these executables should display brief help messages.

5. You can now remove the installation directory.

6. Download the parameter files for one or more species. These files will be named ehparam-species-date.tgz. Unpack them in the parameter files directory (by default, /usr/local/share/exonhunter/param) – there will be one subdirectory for each species.

7. Before you can run ExonHunter, you need to adapt configuration file in $paramdir/species/eh.config (for example: /usr/local/share/exonhunter/param/human/eh.config). This file needs to be configured for each species separately. See the next section.

3.2 Adapting eh.config

This configuration file contains information about your system, and about location of various data files and other tools needed to run ExonHunter.

Quick configuration tip: If you plan to use ExonHunter for ab initio prediction only, you can skip this section; no configuration is necessary.

3.2.1 Global options

Some pipelines that prepare alignments and other types of evidence can use multiple processors. **PROCESSORS** sets the number of processors that should be used. (Note, that the main prediction program will only use single processor.)

After the predictions are done, we assess how well the resulting predictions are supported by the evidence. A coding position is considered supported, if evidence suggests coding region in a correct frame at that position with score at least **SUPPORT_THRESHOLD** (this score is 1, if there is no evidence in support or against coding region; higher score means higher level of support).

Quick configuration tip: If you plan to use ExonHunter support for repeat masking, but do not plan to use alignments or other advisors, you only need to configure the following additional variables:

REPEATMASKER — points to the RepeatMasker executable

REPEATMASKER_OPTIONS — specifies the masking options

REPEATMASKER_SPECIES — specifies the database of repeats that should be used

In such case, you can skip rest of this section.

3.2.2 Evidence sources

For each source of evidence EVNAME, we need to specify how it should be processed. The query fasta sequence is first processed by one of the standard pipelines. The raw results of the pipeline is then turned into advisors for ExonHunter gene prediction.

Quick configuration tip: Most of the variables starting with EVIDENCE are already configured properly. Often you only need to change EVIDENCE_EVNAME_FILES to point to your copy of the database. If you do not plan to use a particular type of evidence, simply comment out variable EVIDENCE_EVNAME_TASK.

EVIDENCE_EVNAME_TASK — type of the evidence; at present the following values are supported in default configuration file: repeat, est, protein, and 35pairs (for PET pairs in some species), but advanced users can also create their own tasks.

EVIDENCE_EVNAME_USE_MASKED — which version of the query should be used in the pipeline? (true = masked, false = unmasked)

EVIDENCE_EVNAME_FILES — location of one or several files that contain additional evidence (such as EST database, protein database, etc.)

EVIDENCE_EVNAME_ABOUT — name of the recipe for post-processing the results of standard pipeline. Descriptions of available .about files is included in each parameter package, and is species specific.

EVIDENCE_EVNAME_BUCKETS — parameters required in order to turn alignments and other pieces of information into probabilities. Description of available bucket files is included in each parameter package, and is species specific.

Examples: The following example specifies source of evidence "repeat" for repeat-masking the query sequence.

```
EVIDENCE_REPEAT_FILES =
EVIDENCE_REPEAT_TASK = repeat
EVIDENCE_REPEAT_ABOUT = repeat.about
EVIDENCE_REPEAT_BUCKETS = repeat
EVIDENCE_REPEAT_USE_MASKED = false
```

The following example specifies source of evidence "human-est" for aligning database of human ESTs to human query sequence. The database is located in file /data/human-est.fasta, is processed through task "est", and is post-processed and filtered using stringent criteria defined in file "est-strong.about".

```
EVIDENCE_HUMAN-EST_FILES = /data/human-est.fasta
EVIDENCE_HUMAN-EST_TASK = est
EVIDENCE_HUMAN-EST_ABOUT = est-strong.about
EVIDENCE_HUMAN-EST_BUCKETS = human-est
EVIDENCE_HUMAN-EST_USE_MASKED = true
```

3.2.3 Configuration of tasks

For each task TASKNAME, we need to specify pipeline that should be used:

```
PROGRAM_TASK = PIPELINE
```

Quick configuration tip: Often, you don't need to change this, unless you are missing some of the programs required by the default pipeline.

Currently supported pipelines are:

- **For repeat masking:** repeatmasker
- **For EST databases:** blat, sim4, phsim4 (uses PatternHunter to pre-filter the database to speed up the alignments), wublastsim4 (uses WU-BLAST to pre-filter the database to speed up the alignments)

- **For protein databases:** blatp (uses BLAT for protein alignment), blastx (uses NCBI blast)
- **For PET pair alignment:** ph, wublast (standard nucleotide alignments)
- **To "mask" gaps in the sequence:** find_gaps
- **To use custom created gtf files:** none (see Section 4.4).

(Note: In this version of ExonHunter, there are no pipelines that can be used for genome-genome alignments.)

Example: In the following lines, we define which pipelines should be used for three tasks: "protein", "est", and "repeat".

```
PROGRAM_PROTEIN = blatp
PROGRAM_EST = blat
PROGRAM_REPEAT = repeatmasker
```

3.2.4 Paths to programs

In this section, you need to set the paths to various programs that are used by the pipelines. Follow the examples in the configuration file.

Example:

```
PH = /usr/bin/java -Xmx1200m -jar /opt/patternhunter/phn.jar
BLAST = /usr/local/bin/blastall
FORMATDB = /usr/local/bin/formatdb
REPEATMASKER = /usr/local/bin/RepeatMasker
SIM4 = /usr/local/bin/sim4
BLAT = /usr/local/bin/blat
```

3.2.5 Pipeline-specific parameters

Specific parameters used by each combination of pipeline PIPELINENAME and task TASKNAME. These parameters are of the form TASKNAME_PIPELINENAME_OPTIONNAME, and are specific to each pipeline. TASKNAME can be omitted, in which case the option is applied to all tasks using this pipeline.

Quick configuration tip: You really don't need to worry about these, unless you are a really advanced user.

4 Using ExonHunter

4.1 ExonHunter as ab initio gene finder

To run ExonHunter as an ab initio gene finder on a file *filename.fasta* which originates in species *spname* (e.g., human), simply run the following command:

```
exonhunter --output filename.gtf filename.fasta spname
```

This command will output predictions to *filename.gtf* in standard GTF format (see `genes.cs.wustl.edu/GTF2.html` for specification). Other options of the exonhunter are described in a help screen that is displayed when you run it without any parameters.

4.2 ExonHunter with repeat masking

ExonHunter will predict many genes in repetitive sequences, unless you use repeat masking. We recommend that instead of using masked sequences, you use the internal ExonHunter pipeline for masking repeats. Provided that RepeatMasker was correctly configured in eh.config file, simply run the following commands:

```
prepare-evidence filename.fasta spname repeat
exonhunter --output filename.gtf filename.fasta spname
```

4.3 Other sources of information

Using other sources of information (EST alignments, protein alignments, etc.) is simple, as long as file eh.config was configured properly. To use all available sources of information, use the following sequence of commands:

```
prepare-evidence filename.fasta spname ALL
exonhunter --output filename.gtf filename.fast spname
```

You can also use only selected sources of additional evidence. To get list of all configured sources, run the following command:

```
prepare-evidence filename.fasta spname -
```

Then you can choose from the list of evidence sources and prepare evidence for ExonHunter by running script 'prepare-evidence' for each of the selected sources separately. The only restriction is that you must run 'repeat' first, because other pipelines often require repeat masked sequence.

Resulting files are stored in a temporary directory. When you run exonhunter command (in the same way as above), ExonHunter will find and use all sources of evidence that were prepared by 'prepare-evidence' scripts.

Additional switches for the 'prepare-evidence' script are described on a help screen that is displayed when you run it without any parameters.

4.4 Custom sources of information

Sometimes you may not wish to run alignments using ExonHunter pipelines, since you may have pre-computed alignments for your sequences. It is possible to use your custom gtf files with coordinates of alignments instead. You can also combine your custom information with output of ExonHunter pipelines. To use this option, follow these steps:

1. Create a temporary directory that will store all the supplementary files as follows:

   ```
   exonhunter --gtfonly filename.fasta spname
   ```

2. (Optional) Run `prepare-evidence` script for any sources of information that you wish to process with ExonHunter pipelines.

3. For each additional custom source of evidence, prepare two files and store them in the temporary directory created by ExonHunter:

   - source.gtf — contains the information (e.g., intervals corresponding to alignments) acquired from this source of information
   - source.about — file specifying how ExonHunter should use this information; for more information, see example .about files for ExonHunter pipelines in the parameter directories.

4. Finally, run ExonHunter; it will use all information described in all .about files in the temporary directory.

   ```
   exonhunter --output filename.gtf filename.fasta spname
   ```

Alternatively, you can place custom .about file together with other .about files to the parameter directory and add a section to eh.config specifying pipeline none as follows (replace “custom” with the name of your custom source:

```
EVIDENCE_CUSTOM_FILES = path_to_gtf_file
EVIDENCE_CUSTOM_TASK = custom
EVIDENCE_CUSTOM_ABOUT = custom.about
EVIDENCE_CUSTOM_BUCKETS = custom
EVIDENCE_CUSTOM_USE_MASKED = false
PROGRAM_CUSTOM = none
```

If no file is specified in EVIDENCE_CUSTOM_FILES, you need to manually place the gtf file to the ehtemp directory. Afterward run prepare-evidence for your custom source, which will create about file in the ehtemp directory automatically. If a file is specified, it will be copied to the ehtemp directory by the prepare-evidence script.

To rerun ExonHunter with different subset of sources of information, or to add more information, you do not need to recreate all the pipelines. Simply delete unwanted .about files, or add new sources of information with prepare-evidence or as custom evidence, and rerun the final exonhunter command.

5 Problems and limitations

5.1 Temporary files and input sizes

Exonhunter creates a temporary directory for each fasta file. This directory contains a subdirectory for each source of evidence. By default, prepare-evidence deletes these temporary subdirectories once it is finished, but the temporary directory as such is preserved. It contains gtf files for individual sources of evidence and their filtered versions after post-processing specified in the .about file. These files may be useful for subsequent visualization of results.

Temporary directory also contains two very big files: super.advisor gives the strength of evidence at each site, external-signals.result contains information about signals such splice site, start and stop codons. These files are very large and should be deleted if space is a concern.

ExonHunter can handle fasta files containing multiple sequences. Each sequence should be under 10Mbp.

5.2 Privileges to data files

When you are using additional databases, such as EST or protein database, you must ensure that you have read privileges to these files, as well as **write privileges to the directory where these files are located.** ExonHunter needs to use `formatdb` (for NCBI BLAST) or `xdformat` (for WU-BLAST). These files create index files in the same directories as the databases are located.

6 Contributors

ExonHunter was developed by Broňa Brejová and Tomáš Vinař under the guidance of their doctoral supervisors Ming Li and Daniel G. Brown at the University of Waterloo in Canada. Some parts of the code were written by David Pál at the University of Waterloo and Peter Kováč at Comenius University in Bratislava, Slovakia.