Broccoli version 1.0

Manual

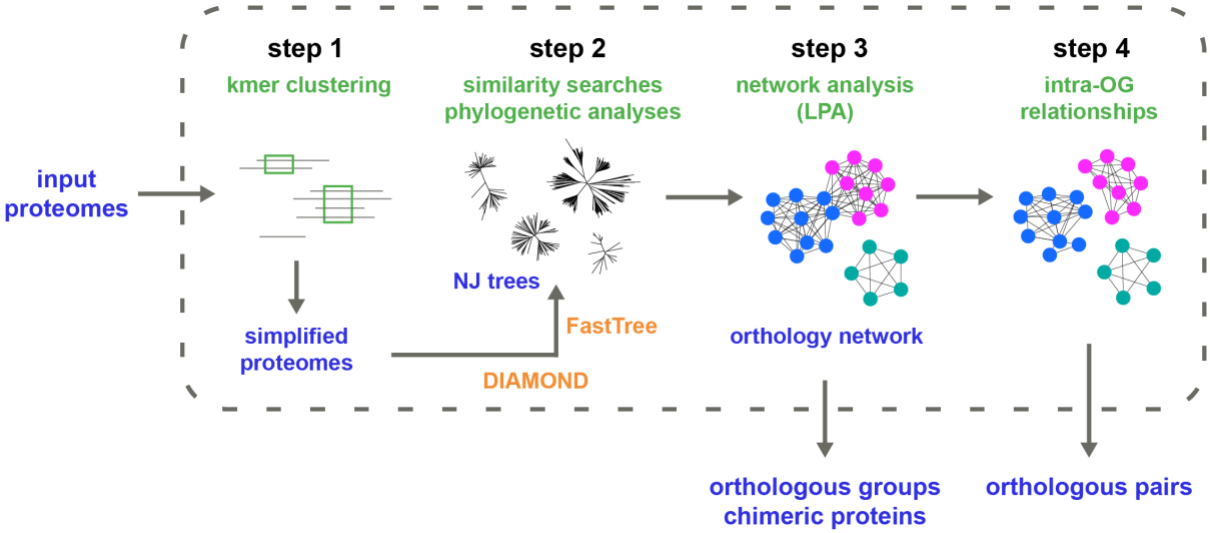
contact:
romain.derelle@gmail.com

Overview

Broccoli, a user-friendly pipeline designed to infer with high precision orthologous groups and pairs of proteins using a phylogeny-based approach. Briefly, Broccoli performs ultra-fast phylogenetic analyses on most proteins and builds a network of orthologous relationships. Orthologous groups are then identified from the network using a parameter-free machine learning algorithm (label propagation). Broccoli is also able to detect chimeric proteins resulting from gene-fusion events and to assign these proteins to the corresponding orthologous groups.

INSERT REFERENCE (not yet submitted)

Here is an overview of the 4 steps of the pipeline:



Requirements

Hardware

Broccoli is fully parallelised. The more CPUs you through at it, the faster your analyses will be. My recommendations are:

- 1 to *n* CPUs, where *n* corresponds to the number of proteomes

nb: there is virtually no additional gain to run Broccoli on more CPUs than the number of proteomes to be analysed

- 2 to 3 GB of RAM memory per CPU (depending on how many proteomes are analysed, and their sizes)

Software

To run Broccoli, you will need:

- a Unix system (MacOS or Linux)
- Python version 3.6 or above
- the ete3 library
- Diamond version 0.9.25 or above
- FastTree2

Installation

You can either install individually each component, or alternatively create a conda environment containing everything required by Broccoli:

1. install Conda in your home directory
2. create a conda environment named 'env-broccoli'

```
conda create -n env-broccoli python=3.6 ete3 diamond fasttree
```

3. activate the environment

```
conda activate env-broccoli
```

you are now ready to use Broccoli using this Terminal window.

Please note that Diamond and FastTree can be installed locally (see Options step 2).

Input data

The input of Broccoli consists of a set of proteome files and grouped in a directory (step 1). Their format should be as follows:

- one proteome file per species
- sequences in FASTA format
- the protein names extracted by Broccoli correspond to the string of characters located before the 1st space. These protein names should be unique (a warning message is printed if not the case).

Output data

Broccoli stores the temporary and output files in 4 directories named ‘dir_step1’ to ‘dir_step4’ corresponding to each of the 4 steps.

The main output files are:

- orthologous_groups.txt (dir_step3)
- chimeric_proteins.txt (dir_step3)
- orthologous_pairs.txt (dir_step4)

Broccoli also generates at each step a log file in its corresponding directory, and a few statistic files are available in ‘dir_step3’.

Running Broccoli

To test Broccoli with the small example dataset present in the directory example_dataset (30 sec to 1mn):

```
python broccoli.py -dir example_dataset
```

Broccoli will store the temporary and output files in 4 directories named dir_step1 to dir_step4 (one for each step) located in the current directory. In this test run, Broccoli should identify 227 orthologous groups, 1 chimeric protein and 863 orthologous pairs.

Here is an example of command line to perform all 4 steps on a dataset present in the directory ‘data’, on 8 threads and with the Diamond and FastTree programs installed locally in the directory ‘pgms’:

```
python broccoli.py -dir ./data -threads 8 -path_diamond ./pgms/diamond
-path_fasttree ./pgms/FastTree
```

There are two things to keep in mind before starting Broccoli:

1. to perform a given step, Broccoli requires all directories generated at the precedent steps

ex: step 3 requires ‘dir_step1’ and ‘dir_step2’ and all their files to be present

nb: it shouldn’t be a problem if you don’t move the directories

2. each step first deletes its corresponding directory if it already exists (to avoid using temporary files from a precedent run)

Options

general options

-steps	steps to be performed, comma separated (default = '1,2,3,4')
-threads	number of threads [default = 1]
-h, -help	show this help message and exit

The option -steps determines which step(s) of the pipeline will be executed. By default, all 4 steps are executed.
If you only need the orthologous groups as output, and not the orthologous pairs, you can limit the analysis to the three first steps: '-steps 1,2,3'.

nb: the steps have to be consecutive (the option '-steps 1,4' is not valide).

The -threads option is used to indicate how many CPUs will be used by the pipeline (all selected steps will run using this number of CPUs).

step 1

-dir	name of the directory containing the proteome files [required]
-extension	extension of proteome files (default = '.fasta')
-min_length	min. length of sequences [default = 10]
-kmer_size	length of kmers [default = 100]
-kmer_min_aa	min. nb of different aa a kmer should have [default = 15]

The input is controlled by the 2 parameters -dir (full or relative path of the directory) and -extension. Files with other extensions present in the directory will be ignored.

Ultra-short sequences with length inferior to -min_length will be ignored and therefore not included in the clustering.

The option -kmer_size corresponds to the length of kmers, while -kmer_min_aa corresponds to the minimum number of different amino-acids a kmer should have (in order to avoid highly repetitive sequences).

step 2

-path_diamond	path of DIAMOND with filename [default = 'diamond']
-path_fasttree	path of FastTree with filename [default = 'fasttree']
-e_value	e-value for similarity search [default = 0.001]
-nb_hits	max. nb of hits per species [default = 5]
-max_gap	max. fraction of gap per position [default = 0.7]

By default, the 2 programs should be named 'diamond' and 'fasttree' and be present in your path. If installed locally, the relative/full path of these 2 programs should be given to Broccoli using the options -path_diamond and path_fasttree.

The -e_value and -nb_hits options corresponds to --evalue and --max-target-seqs of DIAMOND respectively.

Finally, -max_gap governs the trimming of alignments by allowing a certain fraction of gap per position.

step 3

-min_nb_hits	minimum number of hits belonging to the OG [default = 2]
-fusion_shared	minimum fraction of connected nodes in each OG [default = 0.5]
-fusion_nb_sp	minimum nb of species in OGs involved in gene-fusions [default = 2]
-fusion_overlap	maximum overlap between OGs in amino-acids [default = 10]

All these options control the behaviour of the 2 corrections applied after the identification of orthologous groups:

- spurious hits: -min_nb_hits
- chimeric proteins: -fusion_shared, -fusion_nb_sp and -fusion_overlap

step 4

-ratio_ortho	limit ratio ortho/total [default = 0.5]
-not_same_sp	ignore ortho relationships between proteins of the same species

The option -ratio_ortho controls the precision/recall of orthologous pairs: increasing the ratio improve precision while decreasing the ratio improve the recall of orthologous pairs.

Finally, -not_same_sp discards orthologous pairs of proteins from the same species (to be used for the Quest for Orthologs benchmark datasets).

FAQ

How to make Broccoli faster?

Broccoli is already fairly fast. But if you really need to reduce its runtimes, there are 2 parameters you can play with:

- step1: reducing the kmer size allows to simplify further the proteomes and therefore reduce the overall computational time, **but at the obvious risk of clustering paralogs together (be careful)**.

nb: reducing the kmer size has nearly no effect on the simplification of prokaryotic proteomes

nb2: in some species, many unrelated proteins share identical fragments (e.g. viral domains in *Trichomonas vaginalis*)

- step2: reducing the number of hits per species reduces the number of sequences in alignments and trees, **but potentially at the cost of a loss of precision**

How to cite Broccoli?

If you use Broccoli in your research, please cite this paper:
INSERT REFERENCE (not submitted yet)

Please also cite the article corresponding to the programs Diamond and FastTree.