OrthoMCL User's Guide
Version 2.0

UNDER CONSTRUCTION:
  - mysql install documentation
  - mcl documentation

Introduction

For alternate documentation online, please read Unit 6.12 of the Current Protocols of Bioinformatics available at:
  http://onlinelibrary.wiley.com/doi/10.1002/0471250953.bi0612s35/full

For details on the orthomcl algorithm, please read the OrthoMCL Algorithm Document:
  https://docs.google.com/document/d/1RB-SqCjBmcpNq-YbOYdFxotHGuU7RK_wqxqDAMjyP_w/pub

The input to OrthoMCL is a set of proteomes.

The output is a set of files:
  pairs/
     potentialOrthologs.txt
     potentialCoorthologs.txt
     potentialInparalogs.txt
  groups.txt

The files in the pairs/ directory contain pairwise relationships between proteins, and their scores. They are categorized into potential orthologs, co-orthologs and inparalogs as described in the OrthoMCL Algorithm Document (see link above). The groups.txt file contains the groups created by clustering the pairs with the MCL program.

There are three overall stages:
  - all-v-all BLAST
  - the OrthoMCL Pairs program -- makes the pairs/ directory
  - the MCL program -- clusters the pairs to make the groups.txt file

These stages are executed in a series of thirteen steps detailed below. Most simply involve running a provided program. They are broken into steps for ease of backtracking and recoverability. Most are *very simple* to run, so don't be discouraged.

Benchmark Dataset

In the documentation for Orthomcl we refer to a Benchmark Dataset. We tested this set extensively. It had:
  - 100 proteomes (across the tree of life)
  - 1M proteins
  - 500M significant similarities (BLAST hits)

We base hardware requirements and time estimates on this benchmark dataset. The most significant predictor of resourc/time requirements is the number of significant similarities. Even so, as this number changes, resource requirements will change non-linearly.

Requirements

(1) UNIX

- The OrthoMCL Pairs program has only been tested on UNIX, specifically RedHat 5.8. Other UNIX variants may work, but we can't support them.
- The MCL program is UNIX compatible only

(2) BLAST
- we recommend NCBI BLAST for two reasons
  a) theoretically: XXXXXXX
  b) practically: NCBI BLAST supports a tab delimited output which we provide parsers for. See Step 7 below.
- for large datasets (e.g. 1M proteins) all-v-all BLAST will likely require a compute cluster. Otherwise, it could run for weeks or months.

(3) Relational Database
- The OrthoMCL Pairs program runs in a relational database. Supported vendors are:
  - Oracle
  - MySql
- If you don't already have one of these installed, install MySql, which can be done for free and without significant systems administration support. (Follow the instructions we provide below.)

We realize that it is a little inconvenient to require a relational database. However, using a relational database as the core technology for orthomclPairs provides speed, robustness and scalability that would have been very hard to acheive without it.

(4) Hardware
- the hardware requirements vary dramatically with the size of your dataset.
- for the Benchmark Dataset, we recommend:
  - memory: at least 4G
  - disk: 100G free space.
- you can estimate your disk space needs more accurately when you have completed Step 8 below. You will need at least 5 times the size of the blast results file produced in that step. 90% of the disk space required is to hold that file, load it into the database and index it in the database.

(5) Perl
- standard perl
- DBI libraries

(6) MCL program
- please see http://www.micans.org/mcl/sec_description1.html

(7) Time
- The Benchmark Dataset took:
  - 3 days to run all-v-all BLAST on a 500 cpu compute cluster.
  - 16 hours for the orthmclPairs processing to find pairs
  - 2 hours for MCL to find the groups

==========
========== Overview of steps ==========
==========

This is an overview of the thirteen steps to run orthomcl. Details are in the next sections.

All programs except mysql and mcl are provided as part of the OrthoMCL download. The provided programs all begin with 'orthomcl' and will print help if called with no arguments.

(1) install or get access to a supported relational database. If using MySql, certain configurations are required, so it may involve working with your MySql administrator or installing your own MySql. See the mysqlInstallationGuide.txt document provided with the orthomcl software.

(2) download and install the mcl program according to provided instructions.

(3) install and configure the OrthoMCL suite of programs.

(4) run orthomclInstallSchema to install the required schema into the database.

(5) run orthomclAdjustFasta (or your own simple script) to generate protein fasta files in the required format.

(6) run orthomclFilterFasta to filter away poor quality proteins, and optionally remove alternative proteins. Creates a single large goodProteins.fasta file (and a poorProteins.fasta file)

(7) run all-v-all NCBI BLAST on goodProteins.fasta (output format is tab delimited text).

(8) run orthomclBlastParser on the NCBI BLAST tab output to create a file of similarities in the required format

(9) run orthomclLoadBlast to load the output of orthomclBlastParser into the database.

(10) run the orthomclPairs program to compute pairwise relationships.

(11) run the orthomclDumpPairsFiles program to dump the pairs/ directory from the database

(12) run the mcl program on the mcl_input.txt file created in Step 11.

(13) run orthomclMclToGroups to convert mcl output to groups.txt

We recommend you save the output of each step so that you can easily redo it if things go wrong.

==
============ Steps in detail =====================
==

========== Step 1: Install and configure the relational database ============
If you are using Oracle, please see the included oracleConfigurationGuide.txt

If you are using MySQL, please see the included mysqlConfigurationGuide.txt

If you do not have either, please see the mysqlInstallationGuide.txt to install your own mysql.

========== Step 2: install mcl ==========
Get the latest software from http://www.micans.org/mcl/src/mcl-latest.tar.gz

Follow the install instructions.

Also see: http://www.micans.org/mcl/sec_description1.html

========== Step 3: install and configure OrthoMCL programs ========
Input:
  - orthomclSoftware.tar
Output:
  - directory of executable programs
  - home directory for your run of orthomcl
  - orthomcl.config file

Use this command to unpack the software:

```
  tar -xf orthomclSoftware.tar

The result will be this:
  orthomclSoftware/
    bin/
     ...
    doc/
     UserGuide.txt
     orthomcl.config.template
    lib/

The bin/ directory has a set of programs.  To run the programs you will need to
either:
  a) include the orthomclSoftware/bin directory in your PATH
  b) call the programs using their full directory path

Make a directory to hold the data and results for your run of orthomcl.  In this
document we will call that directory "my_orthomcl_dir".

In the orthomclSoftware/doc/Main/OrthoMCLEngine directory is a file called
orthomcl.config.template.  Copy that file to my_orthomcl_dir/orthomcl.config, and edit
the new file according to the following instructions.

In the examples below, it is assumed that your MySql server has a database called
'orthomcl'.  You can either create one (go into the server and run 'create database
orthomcl') or use an existing database, and change the dbConnectString accordingly.

dbVendor=
  - either 'oracle' or 'mysql'
  - used by orthomclInstallSchema, orthomclLoadBlast, orthomclPairs
dbConnectString=
  - the string required by Perl DBI to find the database.
  - examples are:
     dbi:Oracle:orthomcl                (for an oracle database with service name
'orthomcl')
     dbi:MySql:orthomcl                 (for a centrally installed mysql server with
a database called 'orthomcl')
     dbi:MySql:orthomcl:localhost:3307  (for a user installed mysql server on port
3307 with a database called 'orthomcl')
  - used by orthomclInstallSchema, orthomclLoadBlast, orthomclPairs,
orthomclDumpPairsFiles
dbLogin=
  - your database login name
  - used by orthomclInstallSchema, orthomclLoadBlast, orthomclPairs,
orthomclDumpPairsFiles
dbPassword=
  - your database password
  - used by orthomclInstallSchema, orthomclLoadBlast, orthomclPairs,
orthomclDumpPairsFiles
similarSequencesTable=
  - the name to give the table that will be loaded with blast results by
orthomclLoadBlast. This is configurable for your flexibility.  It doesn't matter what
you call it.
  - used by orthomclInstallSchema, orthomclLoadBlast, orthomclPairs
orthologTable=
  - the name of the table that will hold potential ortholog pairs.  This is
configurable so that you can run orthomclPairs multiple times, and compare results.
  - used by orthomclInstallSchema, orthomclPairs, orthomclDumpPairsFiles
inParalogTable=InParalog
  - the name of the table that will hold potential inparalog pairs.  This is
configurable so that you can run orthomclPairs multiple times, and compare results.
  - used by orthomclInstallSchema, orthomclPairs, orthomclDumpPairsFiles
coOrthologTable=CoOrtholog
  - the name of the table that will hold potential coortholog pairs.  This is
configurable so that you can run orthomclPairs multiple times, and compare results.
```

  - used by orthomclInstallSchema, orthomclPairs, orthomclDumpPairsFiles
interTaxonMatchView=InterTaxonMatch
percentMatchCutoff=
  - blast similarities with percent match less than this value are ignored.
  - used by orthomclPairs
evalueExponentCutoff=
  - blast similarities with evalue Exponents greather than this value are ignored.
  - used by orthomclPairs
oracleIndexTblSpc=
  - optional table space to house all oracle indexes, if required by your oracle
server. default is blank.

========== Step 4: orthomclInstallSchema ========
Input:
  - database
Output:
  - database with schema installed

Run the orthmclInstallSchema program to install the schema. (Run the program with no arguments to get help. This is true of all following orthomcl programs.)

Benchmark time: < 1 min

========== Step 5: orthomclAdjustFasta ==========
Input:
  - fasta files as acquired from the genome resource.
Output:
  - the my_orthomcl_dir/compliantFasta/ directory of orthomcl-compliant fasta files
(see Step 6)

Use orthomclAdjustFasta to produce a compliant file from any input file that conforms to the following pattern (for other files, provide your own script to produce complaint fasta files):
  - has one or more fields that are separated by white space or the '|' character
(optionally surrounded by white space)
  - has the unique ID in the same field of every protein.

First, for any organism that has multiple protein fasta files, combine them all into one single proteome fasta file

Then, create an empty my_orthomcl_dir/compliantFasta/ directory, and change to that directory. Run orthomclAdjustFasta once for each input proteome fasta file. It will produce a compliant file in the new directory. Check each file to ensure that the proteins all have proper IDs.

Benchmark time: < 1 min per genome

======== Step 6: orthomclFilterFasta ===========
Input:
  - my_orthomcl_dir/compliantFasta/
  - optionally a gene->protein mapping file
Output:
  - my_orthomcl_dir/goodProteins.fasta
  - my_orthomcl_dir/poorProteins.fasta
  - report of suspicious proteomes (> 10% poor proteins)

This step produces a single goodProteins.fasta file to run BLAST on. It filters away poor-quality sequences (placing them in poorProteins.fasta). The filter is based on length and percent stop codons. You can adjust these values.

The input requirements are:

  1) a compliantFasta/ directory which contains all and only the proteome .fasta files, one file per proteome.
  1) each .fasta file must have a name in the form 'xxxx.fasta' where xxxx is a three or four letter unique taxon code. For example: hsa.fasta or eco.fasta
  2) each protein in those files must have a definition line in the following format:
     >xxxx|yyyyyyyy
     where xxxx is the three or four letter taxon code and yyyyyyy is a sequence identifier unique within that taxon.

Change dir to my_orthomcl_dir/ and run orthomclFilterFasta.

Benchmark time: 5 min

========= Step 7: All-v-all BLAST =========
Input:
  - goodProteins.fasta
Output:
  - your_blast_results_in_tab_format

You must run your own BLAST. For large datasets you should consider gaining access to a compute cluster.

We expect you to:
  - use NCBI BLAST
  - run with the -m 8 option to provide tab delimited output required by Step 8
  - for IMPORTANT DETAILS about other BLAST arguments, see:
    the OrthoMCL Algorithm Document (https://docs.google.com/document/d/1RB-SqCjBmcpNq-YbOYdFxotHGuU7RK_wqxqDAMjyP_w/pub)

If you are a power user you can deviate from this, so long as you can ultimately provide output in exactly the format provided by NCBI BLAST using the -m 8 option, and expected by Step 8.

If you are a super-power user you can deviate from that, and also skip Step 8.   But you must be able to provide the exact format file created by that step as expected by Step 9. The tricky part is computing percent match.

Time estimate: highly dependent on your data and hardware

========= Step 8: orthomclBlastParser ========
Input:
  - your_blast_results_in_tab_format
  - my_orthomcl_dir/compliantFasta/
Output:
  - my_orthomcl_dir/similarSequences.txt

This step parses NCBI BLAST -m 8 output into the format that can be loaded into the orthomcl database.

Use the orthomclBlastParser program for this.   In addition to formatting, it computes the percent match of each hit, which is tricky (see the perl code if you are a super-power user.)

orthomclBlastParser my_blast_results compliantFasta >> similarSequences.txt

IMPORTANT NOTE: the size of this file determines the disk space required by the relational database. You will need 5x the size of this file. Please see the oracleConfigGuide or mysqlConfigGuide now that you know the size of this file.

Benchmark time: 10 min

========= Step 9: orthomclLoadBlast ===========
Input:

  - similarSequences.txt
Output:
  - SimilarSequences table in the database

This step loads the BLAST results into the orthomcl database.

Use the orthomclLoadBlast program for this.

NOTE: You might get the following error when you run this command:

  "The used command is not allowed with this MySQL version."

The SQL that causes this is LOAD DATA LOCAL INFILE. MySql needs specific configuration to enable this command. See these two pages:

http://dev.mysql.com/doc/refman/5.1/en/load-data-local.html
http://dev.mysql.com/doc/refman/5.0/en/loading-tables.html

In sum:
  1) you need to start the MySql server with the option: --local_infile=1
  2) during installation, MySql needs to be have been compiled with: --enable-local-infile.

It is possible that your MySql was compiled with that flag. It is included by default in some distributions of MySql. The only way we know of to find out what compile flags were used is to try the mysqlbug command. This command opens an email so you can report a bug. Apparently at the bottom of the mail is a list of compile flags. Once you see them you can abort the mail. If your mysql was not compiled with that flag it will need to be reinstalled.

Benchmark time: 4 hours

========= Step 10: orthomclPairs =========
Input:
  - SimilarSequences table in the database
Output:
  - PotentialOrthologs table
  - PotentialInParalogs table
  - PotentialCoOrthologs table

This is a computationally major step that finds protein pairs. It executes the algorithm described in the OrthoMCL Algorithm Document (docs.google.com/Doc?id=dd996jxg_1gsqsp6), using a relational database. The program proceeds through a series of internal steps, each creating an intermediate database table or index. There are about 20 such tables created. Finally, it populates the output tables.

The cleanup= option allows you to control the cleaning up of the intermediary tables. The 'yes' option drops the intermediary tables once they are no longer needed. The 'no' option keeps the intermediary tables in the database. In total, they are expected to be about 50 percent of the SimilarSequences table. They are useful mostly for power users or developers who would like to query them. They can be removed afterwards with the 'only' or 'all' options. The latter also removes the final tables, and should only be done after Step 11 below has dumped them to files.

The startAfter= option allows you to pick up where the program left off, if it stops for any reason. Look in the log to find the last completed step, and use its tag as the value for startAfter=

Because this program will run for many hours, we recommend you run it using the UNIX 'screen' program, so that it does not abort in the middle. (If it does, use startAfter=).

Benchmark time: 16 hours

========== Step 11: orthomclDumpPairsFiles ========
Input:
  - database with populated pairs tables
Output
  - pairs/ directory.
  - mclInput file

Run the orthomclDumpPairsFiles.

The pairs/ directory contains three files: ortholog.txt, coortholog.txt,
inparalog.txt. Each of these has three columns:
   - protein A
   - protein B
   - their normalized score (See the Orthomcl Algorithm Document).

Benchmark time: 5 min

========== Step 12: mcl ========
Input:
  - mclInput file
Output:
  - mclOutput file

mcl my_orthomcl_dir/mclInput --abc -I 1.5 -o my_orthomcl_dir/mclOutput

Benchmark time: 3 hours

========== Step 13: orthomclMclToGroups ==========
Input:
  - mclOutput file
Output:
  - groups.txt

Change to my_orthomcl_dir and run:
  orthomclMclToGroups my_prefix 1000 < mclOutput > groups.txt

my_prefix is an arbitrary string to use as a prefix for your group IDs.

1000 is an arbitrary starting point for your group IDs.

Benchmark time: 1 min