Using OrthoMCL to Assign Proteins to OrthoMCL-DB Groups or to Cluster Proteomes Into New Ortholog Groups

Steve Fischer,[1,3] Brian P. Brunk,[1,2] Feng Chen,[4] Xin Gao,[1,2] Omar S. Harb,[1,2] John B. Iodice,[1,3] Dhanasekaran Shanmugam,[2] David S. Roos,[2] and Christian J. Stoeckert Jr.[1,3]

[1]Center for Bioinformatics, University of Pennsylvania, Philadelphia, Pennsylvania
[2]Department of Biology, University of Pennsylvania, Philadelphia, Pennsylvania
[3]Department of Genetics, University of Pennsylvania, Philadelphia, Pennsylvania
[4]Bayer Business and Technology Services, Bayer Healthcare Pharmaceuticals, Wayne, New Jersey

ABSTRACT

OrthoMCL is an algorithm for grouping proteins into ortholog groups based on their sequence similarity. OrthoMCL-DB is a public database that allows users to browse and view ortholog groups that were pre-computed using the OrthoMCL algorithm. Version 4 of this database contained 116,536 ortholog groups clustered from 1,270,853 proteins obtained from 88 eukaryotic genomes, 16 archaean genomes, and 34 bacterial genomes. Future versions of OrthoMCL-DB will include more proteomes as more genomes are sequenced. Here, we describe how you can group your proteins of interest into ortholog clusters using two different means provided by the OrthoMCL system. The OrthoMCL-DB Web site has a tool for uploading and grouping a set of protein sequences, typically representing a proteome. This method maps the uploaded proteins to existing groups in OrthoMCL-DB. Alternatively, if you have proteins from a set of genomes that need to be grouped, you can download, install, and run the stand-alone OrthoMCL software. *Curr. Protoc. Bioinform.* 35:6.12.1-6.12.19. © 2011 by John Wiley & Sons, Inc.

Keywords: OrthoMCL • ortholog groups • paralog • proteome • Markov clustering • reciprocal best hits • MCL

INTRODUCTION

An *OrthoMCL group* is a set of proteins across one or more species that represent putative orthologs and in-paralogs. Groups are formed by running the OrthoMCL (Li et al., 2003) software on the sequences from a set of proteomes (a *proteome* is the set of proteins that belong to a species; typically, its sequences are derived through the annotation of a complete genome). The input would be anywhere from a few to hundreds of proteomes.

OrthoMCL-DB (Chen et al., 2006) contains ortholog groups for most completely sequenced and annotated eukaryotes and for a number of completely sequenced and annotated prokaryotes. It provides a wealth of functionality, including domain architecture for each group, phyletic patterns for each group, and advanced querying, including phylogenetic pattern searches. Yet, it is in a sense limited by the genomes that are included.

Overview of algorithm

OrthoMCL groups proteins into "ortholog groups." That name is a little misleading because the groups contain *proteins* related by:

orthology (recent descent)
in-paralogy (recent duplication)
co-orthology (recent descent and duplication).

Current Protocols in Bioinformatics 6.12.1-6.12.19, September 2011
Published online September 2011 in Wiley Online Library (wileyonlinelibrary.com).
DOI: 10.1002/0471250953.bi0612s35

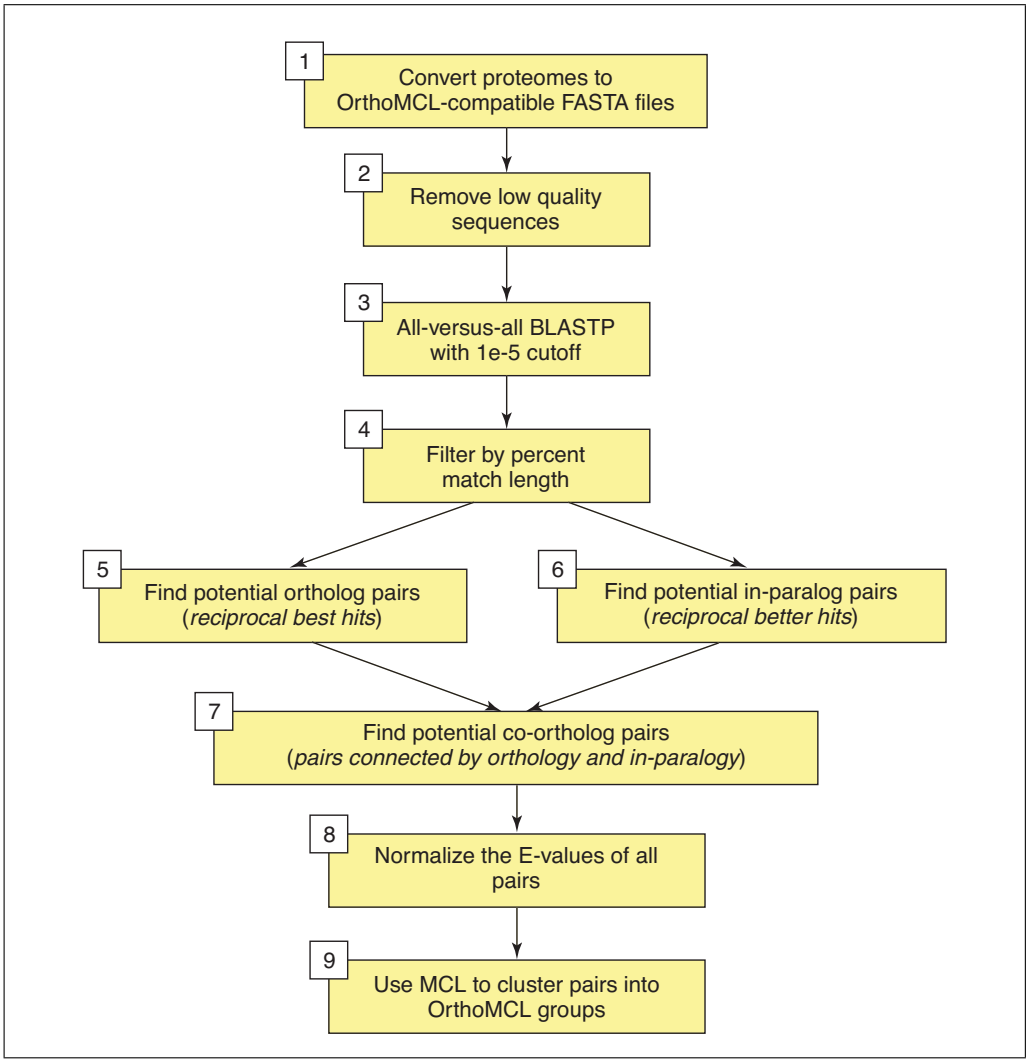

Figure 6.12.1 Overview of the OrthoMCL algorithm. (1) Proteomes must each be in FASTA format where the file name and definition lines comply with simple requirements. (2) The proteome files are filtered to remove low-quality sequences based on length and percent stop codons. (3) The proteomes are all compared to each other using BLASTP. They are masked with `seg` and an e-value cutoff of 1e-5 is applied. (4) For each pair of sequences that match, compute the "percent match length" score: count the number of amino acids in the shorter sequence that participate in any HSP, divide that by the length of the shorter sequence, and multiply by 100. Filter away matches with percent match $<$ 50%. (5) For all pairs of proteomes, find all pairs of proteins across them that have hits as good as or better than any other hits between these proteins and other proteins in those species. (6) Find all pairs of proteins within a species that have mutual e-values that are better than or equal to all of those proteins' hits to proteins in other species. (7) Find all pairs of proteins across two species that are connected through orthology and in-parology. (8) Normalize in-paralog e-values by averaging all *qualifying* in-paralog pairs in a genome and divide each pair by the average. Within a genome, in-paralog pairs qualify if either of the proteins in the pair has an ortholog in any genome. If no in-paralogs within a genome have any orthologs, all in-paralogs in that genome qualify. Normalize ortholog and co-ortholog pairs for any two species by averaging the e-values across them, and normalize using that average. (9) Pass on all ortholog, in-paralog, and co-ortholog pairs, with their normalized e-values, to the MCL program for clustering.

The computation is done on protein sequence because it is more sensitive than genomic sequence, but by definition the evolutionary relationships the algorithm attempts to discover exist in genome space. A complication of using proteins is that proteomes may contain alternative proteins (from alternative transcription). As discussed in Phase 1 below, this can be partially addressed by filtering.

There are a number of approaches to identifying groups of proteins with a common evolutionary history (Chen et al., 2007). These include methods based on phylogeny, evolutionary distance metrics, and sequence similarity (BLAST; Altschul et al., 1990; see *UNIT 3.4*). OrthoMCL uses the reciprocal best BLAST hit approach and makes an adjustment for species distance (normalization) to distinguish orthologs from in-paralogs. Groups are generated from the normalized BLAST scores between proteins using Markov clustering (Enright et al., 2002). Figure 6.12.1 provides an overview of the approach. Please refer to the OrthoMCL Algorithm Document (*https://docs.google.com/View?id=dd996jxg_1gsqsp6*) for details about how OrthoMCL-DB is created.

Overview of the protocols

The protocols described here will help you extend OrthoMCL-DB groups to include your own proteins (Basic Protocol 1) or to create your own groups using OrthoMCL software (Basic Protocol 2). In both cases, the result is a set of files with group assignments.

Using Basic Protocol 1, you map your proteins to existing OrthoMCL-DB groups. Once you have done so, you can turn to the OrthoMCL-DB Web site to learn about groups of interest and by extension your proteins that map to them. Both of these protocols, while running on OrthoMCL-DB servers, afford reasonable precautions to protect the privacy of your data, including removing it from the server when the job is done. Using Basic Protocol 2, you get groups independent of OrthoMCL-DB.

Choose Basic Protocol 1, *Assign a proteome to OrthoMCL-DB groups*, if you have one or a small number of proteomes that you would like to associate with OrthoMCL-DB groups. This is a quick way to approximate the groups that would result if your proteomes were included in the build of OrthoMCL-DB. You will also get groups of in-paralogs for your proteome. Also choose Basic Protocol 1 if you have a set of proteins that do not represent a single proteome, and would like to associate them with OrthoMCL-DB groups. In this case you will not get valid in-paralog groups. Note that Basic Protocol 1 will give you an approximation of including your proteins in an OrthoMCL-DB build, but there will be subtle differences because your proteins will not be submitted to the full OrthoMCL-DB build process (which is computationally huge). Choose Basic Protocol 2, *Create ortholog groups from your proteomes using the OrthoMCL software*, if (a) you have a small number of proteomes that are closely related and you do not need to leverage the grouping information in OrthoMCL-DB; (b) you have a small number of proteomes and would like to have them grouped but don't want the data to leave your facility for security reasons; or (c) you have a large number of proteomes.

STRATEGIC PLANNING

Alternative splice forms

Higher eukaryotes commonly have genes with alternative splice forms, leading to multiple proteins per gene. These "alternative proteins" can be either included or excluded in the input data (Basic Protocol 2, step 4).

There are theoretical and practical considerations when making the decision about including or excluding alternative proteins. OrthoMCL-DB Version 4 included only the longest splice form.

The theoretical reasons for excluding them are:

To avoid "pseudo-in-paralogs" which are alternative proteins within a group that therefore look like in-paralogs. If needed, these could be distinguished using the `gene—>protein` mapping files)

Group statistics (counts, coherence) are in terms of genes, and are not skewed by the presence of highly similar pseudo-in-paralogs. This helps avoids inaccuracy in terms of gene duplication events or paralogs.

The theoretical reasons against excluding them are:

Doing so may exclude important functional components of genes that are omitted in the longest variant.

Doing so may exclude distinct proteins that are omitted in the longest variant, for example, if an alternative splice form introduces a frame shift that leads to a very different protein.

The practical reasons for excluding them are:

Smaller input dataset, so requires less resources.

The practical reasons against excluding them are:

Requires a `gene-->protein` mapping file.

Many providers of protein sequence do not provide such a mapping. If you find one that does, it may not have the latest version of the protein sequences, so you may be forced into a tradeoff between getting the preferred protein sequences and getting a gene-protein mapping file.

Resources

Basic Protocol 2 requires access to a relational database system and significant computational resources. The OrthoMCL software uses a relational database as a set processing engine. It is compatible with Oracle and MySQL. You will need to create tables and, depending on the size of your data, you may need a significant amount of disk space (described below). It also requires computing resources adequate for doing an all-versus-all BLASTP (NCBI BLAST) of your set of proteomes. If you have a large number of proteomes, this will require a compute cluster.

Choosing proteomes

For Basic Protocol 2, your choice of proteomes will affect the granularity of the resulting ortholog groups. If the proteomes are closely related, members of a group will be more closely related and more distant relationships will be placed into separate groups. Another consideration is including one or more well annotated proteomes. These enable an internal validation for determining whether the protocol was successful and a means for annotating your (un-annotated) proteins based on the groups in which they reside.

BASIC PROTOCOL 1

ASSIGN A PROTEOME TO OrthoMCL-DB GROUPS

Use this tool to assign the proteins in a genome to OrthoMCL-DB Groups and to find groups of in-paralogs. You will upload your genome's proteins as a FASTA file to a service on the OrthoMCL-DB Web site that maps them to OrthoMCL-DB groups. The result is a set of files, as described below (your proteins are not incorporated into the OrthoMCL-DB site). Because your proteins are not actually included in an OrthoMCL-DB build process, which is a multi-week effort, the results will be an approximation to the result of including them in the OrthoMCL-DB build, but not identical. While the standard use of this service is to upload the complete set of proteins of a single proteome, you can also upload a set of proteins that are from a partial proteome or many proteomes. In both cases, you will get valid assignments of your proteins to OrthoMCL-DB groups.

However, in the former case, you will also get a valid set of in-paralog groupings, while in the latter you will not.

How the tool processes your proteins

This protocol maps your proteins to groups in OrthoMCL-DB. To understand how OrthoMCL-DB is created, please see the OrthoMCL-DB Methods page.

Phase 1

In this phase, the tool maps your proteins to groups in OrthoMCL-DB. It performs a BLASTP against all the proteins in OrthoMCL-DB, using a cutoff of 1e-5 and 50% match. Your protein is assigned to the group containing its best hit. If the best matching protein does not have a group, it is assigned to NO_GROUP.

Phase 2

In this phase, the tool processes all of your proteins that did not have an above-threshold match in phase 1. It uses the OrthoMCL-DB in-paralog algorithm described in the OrthoMCL Algorithm Document (*https://docs.google.com/View?id=dd996jxg_1gsqsp6*) to create pairs of potential in-paralogs. It then submits those pairs to the MCL program for clustering. The result is groups of your proteins that provisionally represent in-paralogs.

Necessary Resources

Hardware

A computer with an Internet connection

Software

An Internet browser, e.g., Internet Explorer (*http://www.microsoft.com/ie*), Firefox (*http://www.mozilla.org/firefox*), or Safari (*http://www.apple.com/safari*).
A program to unpack `.zip` files (these are standard on Macintoshes and Windows PCs)

Files

A FASTA file (see *APPENDIX 1B*) with a set of protein amino acid sequences. Typically the proteins will be from a single proteome, though this is not a requirement. The definition lines should have a unique identifier immediately following the > character, and followed by either a new-line or at least one space character.

1. From the OrthoMCL-DB home page (*http://www.orthomcl.org*; see Fig. 6.12.2) click on the Tools link at the top right, and choose "Assign your proteins to groups."
2. Provide the name of your FASTA file, or Browse to it.
3. Provide your e-mail address so you can get your results.
4. Optionally provide a job name. This is for you to distinguish different jobs you do.
5. Submit the job.
6. You will receive an e-mail within 5 min informing you that the job is on the queue.
7. You will receive an e-mail when the job is done, possibly many hours later, but less than 24 hr. Please retrieve your results within 48 hr of getting this e-mail.
8. Download your results. They will be in packaged in a `.zip` file. Use a standard `.zip` unpacker.

Figure 6.12.2 OrthoMCL-DB home page with the Tools link circled.

BASIC PROTOCOL 2

CREATE ORTHOLOG GROUPS FROM YOUR PROTEOMES USING THE OrthoMCL SOFTWARE

This protocol describes how to download, install, and run the OrthoMCL software. Use this software if you have at least two proteomes, and up to hundreds, and want to find ortholog groups. For details on the OrthoMCL algorithm, please read the OrthoMCL Algorithm Document (*https://docs.google.com/View?id=dd996jxg_1gsqsp6*).

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 in-paralogs as described in the OrthoMCL Algorithm Document (*https://docs.google.com/View?id=dd996jxg_1gsqsp6*). The `groups.txt` file contains the groups created by clustering the pairs with the MCL program.

How the software processes your proteins

In overview, there are four phases of processing by the software. The first phase is preparing your FASTA files (one per proteome). The second phase is running all-versus-all BLASTP (this document does not provide instructions for this; please see the NCBI BLAST documentation). The third phase is loading the BLASTP results into the relational database and running the OrthoMCL software to find significant pairs of proteins. The final phase is using the MCL software to cluster the pairs into groups.

The benchmark dataset

In this protocol we refer to a *benchmark dataset*. We have tested this set extensively. It had:

100 proteomes (across the tree of life, mostly eukaryotes)
1 million proteins
500 million significant similarities (BLAST hits).

The benchmark dataset took:

3 days to run all-versus-all BLAST on a 200 CPU compute cluster.
16 hr on a Linux server for the `orthmclPairs` processing to find pairs (using MySQL as the relational database)
2 hr on a Linux server for MCL to find the groups.

We base hardware requirements and time estimates on this benchmark dataset. The most significant predictor of resource/time requirements is the number of significant similarities. As this number changes, resource requirements will change nonlinearly.

Necessary Resources

Hardware

Computer resources to run an all-versus-all BLASTP on the proteins from your proteomes.
A file system with space sufficient to house the results of the all-versus-all-BLAST.
A database server to host an Oracle or MySQL database that will do significant relational database processing. The server's requirements vary dramatically with the size of your dataset. For the Benchmark Dataset, we recommend:
memory: at least 4G
disk: 100 GB free space. You can estimate your disk space needs more accurately when you have completed step 7 of this protocol. 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 load that file into the database and index it in the database.
A Linux server to run the OrthoMCL software

Software

NCBI BLASTP (see *UNIT 3.4*). Note that The OrthoMCL software requires NCBI BLASTP output. Other BLASTP software versions do not have compatible output. The NCBI BLAST software is available at *ftp://ftp.ncbi.nih.gov/blast/*.
The Linux operating system. The `orthomclPairs` program has only been tested on Linux. The MCL program is Unix-compatible only.
The MCL software, available at *http://www.micans.org/mcl/*
Oracle or MySQL (see *UNIT 9.2*). The `orthomclPairs` program runs in a relational database. If you don't already have one available, install MySQL. You can do it for free and without significant systems administration support (OrthoMCL uses a relational database as its core technology for speed, robustness and scalability that would have been very hard to achieve otherwise).
Perl 5.8, including DBI libraries

Files

A set of FASTA files, one file per proteome. These must conform to the format (described below) expected by the OrthoMCL software.

1. Follow the Support Protocol to download, install, and configure the OrthoMCL programs.

2. Install and configure the relational database. If you are using Oracle, see the included `oracleConfigurationGuide.txt`. If you are using MySQL, see the included `mysqlConfigurationGuide.txt`. If you do not have either, see `mysqlInstallationGuide.txt` to install your own MySQL (see Support Protocol, step 2).

3. Download 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*.

4. Run the `orthomclAdjustFasta` on your proteome FASTA files to create a new set of FASTA files that are OrthoMCL compliant, and ready to use in step 5, below.

   Input:
   FASTA files as acquired from the genome resource.

   Output:
   the `my_orthomcl_dir/compliantFasta/` directory of OrthoMCL-compliant FASTA files.

   The FASTA-format specification requires that each sequence in a `.fasta` *file be separated by a line beginning with a* `>` *character. This line is typically called the definition line and contains information about the sequence. Step 5 below expects your FASTA files to conform to the following specific requirements: (1) they must be in a* `compliantFasta/` *directory which contains all of (and only) your proteome* `.fasta` *files, one file per proteome; (2) each* `.fasta` *file must have a name in the form* `xxxx.fasta`*, where* `xxxx` *is a three- or four-letter unique taxon code (e.g.,* `hsa.fasta` *or* `eco.fasta`*).*

   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.*

   Use `orthomclAdjustFasta` to convert your FASTA files to FASTA files that conform to those requirements. The FASTA files that are input to it must have definition lines (1) with one or more fields that are separated by white space or the `|` character (optionally surrounded by white space); and (2) that have the protein's unique ID in the same field for every protein.

   To use `orthomclAdjustFasta`:

   a. For any organism that has multiple protein FASTA files, combine them all into one single proteome FASTA file.
   b. Create an empty `my_orthomcl_dir/compliantFasta/` directory, and change to that directory.
   c. Run `orthomclAdjustFasta` with no arguments to get its help. This will tell you about the arguments you need to provide.
   d. Run `orthomclAdjustFasta` (with appropriate arguments) once for each input proteome FASTA file. It will produce a compliant file in the new directory.
   e. Check each file to ensure that the proteins all have proper IDs.

   Benchmark time: $\leq$ 1 min per genome.

5. Run the `orthomclFilterFasta` program to remove low-quality protein sequences from your FASTA files, and to combine your individual compliant proteome FASTA files into a single `goodProteins.fasta` file.

   Input:
   `my_orthomcl_dir/compliantFasta/` (See Step 4)
   optionally a `gene->protein` mapping file

   Output:
   `my_orthomcl_dir/goodProteins.fasta`
   `my_orthomcl_dir/poorProteins.fasta`
   a report of suspicious proteomes (> 10% poor proteins) on standard output.

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 *sequence length* and *percent stop codons*. You can adjust these values.

To run orthomclFilterFasta:

a. Change to `my_orthomcl_dir/`.
b. Run `orthomclFilterFasta` to get help, which will show you the options.
c. Run `orthomclFilterFasta` with appropriate arguments. Unless you are a power user, use the suggested values.

The program will print the name of any file containing more than 10% rejected proteins, along with the rejected percentage. You should consider removing these from your set of proteomes.

Benchmark time: 5 min.

6. Run all-versus-all BLASTP with `goodProteins.fasta` as the BLAST database and subject sequences.

Input:
`goodProteins.fasta`

Output:
`your_blast_results_in_tab_format`

This document does not provide assistance on the details of running BLASTP. For large datasets you should consider gaining access to a compute cluster. When you do so, you will need to: (1) use NCBI BLAST; (2) run with the `-m 8` option to provide tab delimited output required by step 8.

Use these options:

```
-F 'm S' -v 100000 -b 100000 -z db_size -e 1e-5 —m 8
```

where: `-F 'm S'` signifies "mask with Seg"; `-v 100000` is a "don't care" value; `-b 100000` is a "don't care" value; `-z db_size` is the number of proteins in the set (see "Incrementally add a genome" below); and `-e 1e-5` is the recommended e-value.

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 7.

If you are a super-power user, you can deviate from that, and also skip step 7. But you must be able to provide the exact format file created by that step as expected by step 8. The tricky part is computing percent match.

Benchmark time: 3 days

Incrementally add a genome: Once a large all-versus-all BLAST has been completed, you may need to "incrementally" add a new proteome, without re-running the large all-versus-all BLAST. To do so: (1) prepare the new proteome's FASTA file as you did for the previous ones (steps 4 and 5); (2) make a new BLAST database that includes all the previous proteins plus the new FASTA file; (3) use the `-z` argument of BLAST to simulate the size of the all database, so that the statistics and scoring are compatible with the original all-versus-all BLAST. Use the same `-z` value as was used in the original BLAST.

7. Use the `orthomclBlastParser` program to convert the tab-delimited file returned by BLASTP (using the `—m 8` option) into a format ready for loading into the OrthoMCL schema in your relational database.

   Input:
   `my_blast_results`
   `my_orthomcl_dir/compliantFasta/`

   Output:
   `my_orthomcl_dir/similarSequences.txt`

   In addition to formatting, this program computes the *percent match* and *percent identity* of each hit:

   Percent identity: Taken directly from the BLAST result file, from the best HSP per hit.

   Percent match: (1) Select whichever is shorter, the query or subject sequence. Call that sequence S. (2) Count all amino acids in S that participate in any HSP. (3) Divide that count by the length of S and multiply by 100.

   Use the `orthomclBlastParser` program as shown below to do the conversion:

   ```
   % orthomclBlastParser my_blast_results
   my_orthomcl_dir/compliantFasta >> my_orthomcl_dir/
   similarSequences.txt
   ```

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

   Benchmark time: 10 min.

8. Use the `orthomclLoadBlast` program to load the `similarSequences.txt` file (made in step 7) into the relational database.

   Input:
   `my_orthomcl_dir/similarSequences.txt`
   `my_orthomcl_dir/orthomcl.config`

   Output:
   `SimilarSequences` table in the database
   Use the `orthomclLoadBlast` program for this.
   Benchmark time: 4 hr.

9. Run the `orthomclPairs` program to find pairs of proteins that are potentially orthologs, in-paralogs or co-orthologs.

   Input:
   `SimilarSequences` table in the database
   `my_orthomcl_dir/orthomcl.config`

   Output:
   `Orthologs` table
   `InParalogs` table
   `CoOrthologs` table

This is a computationally intensive step that finds protein pairs. It analyzes the BLAST results to find different types of reciprocal best hits (orthologs, co-ortholog, and in-paralog). To do so, it executes the algorithm described in the OrthoMCL Algorithm Document (*https://docs.google.com/View?id=dd996jxg_1gsqsp6*) using a relational database. The program proceeds through a series of about 20 internal steps, each creating an intermediate database table or index. Finally, it populates the three output tables.

In total, the intermediary tables are expected to be about 50 percent of the size of the `SimilarSequences` table.

Run the `orthomclPairs` program with no arguments to see its help.

There are two options to `orthomclPairs`:

`cleanup=` allows you to control the cleaning up of the intermediary tables.

`yes` drops the intermediary tables once they are no longer needed.
`no` keeps the intermediary tables in the database.
`only` runs the program but does no work except clean up intermediate tables
`all` is the same as `only` but also removes the three final output tables, and should only be done after step 10 (below) has dumped them to files.

`startAfter=` allows you to pick up where the program left off, if it stops for any reason. Look in the `orthomclPairs` 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 hr.

10. Use the `orthomclDumpPairsFiles` to create a set of result files from the results in the database made by `orthomclPairs`.

    Input:
    OrthoMCL database with populated pairs tables
    `my_orthomcl_dir/orthomcl.config`

    Output:
    `my_orthomcl_dir/pairs/`
    `my_orthomcl_dir/mclInput`
    Run the `orthomclDumpPairsFiles` with no arguments, to get its help.
    Then, change directory to `my_orthomcl_dir` and run:
    `% orthomclDumpPairsFiles orthomcl.config`

    Output files:

`pairs/orthologs.txt`	ortholog relationships
`pairs/inparalogs.txt`	in-paralog relationships
`pairs/coorthologs.txt`	co-ortholog relationships
`orthomclMclInput`	file required by the `mcl` program

The `pairs/` directory contains three files: `orthologs.txt`, `coorthologs.txt`, and `inparalogs.txt`. Each of these files describes pair relationships between proteins. They have three columns: (1) protein 1; (2) protein

2; (3) a normalized similarity score (See the OrthoMCL Algorithm Document for the normalization function).

These are candidate relationships (edges) that will subsequently be grouped (clustered) by the `mcl` program to form the OrthoMCL ortholog groups. These files contain *more sensitive* and *less selective* relationships than the final OrthoMCL groups. The OrthoMCL Algorithm Document (*https://docs.google.com/View?id=dd996jxg_1gsqsp6*) provides a formal definition of the relationships in these files. Here is a summary:

In-paralogs: all pairs of proteins within a species that have mutual hits that are better or equal to all of those proteins' hits to proteins in other species.

Orthologs: all pairs of proteins across two species that have hits as good as or better than any other hits between these proteins and other proteins in those species.

Co-orthologs: all pairs of proteins across two species that can be connected by following ortholog and in-parolog pairs. For any ortholog pair, the in-paralogs of each will form co-ortholog pairs with each other.

The `orthomclMclInput` file contains the identical information as the three pairs files but merged into a single file and in a format expected by the `mcl` program.

Benchmark time: 5 min.

11. Use the `mcl` program to cluster the pairs found in step 10 into the final OrthoMCL ortholog groups.

    Input:
    `my_orthomcl_dir/mclInput`

    Output:
    `my_orthomcl_dir/mclOutput`

    Use this command to run the mcl program:

    ```
    % mcl my_orthomcl_dir/mclInput —abc -I 1.5 -o
      my_orthomcl_dir/mclOutput
    ```

    Benchmark time: 3 hr.

12. Use the `orthomclMclToGroups` program to convert the file output by the `mcl` program into the final OrthoMCL-style groups file.

    Input:
    `my_orthomcl_dir/mclOutput`

    Output:
    `my_orthomcl_dir/groups.txt`
    Change to `my_orthomcl_dir/` and run:

    ```
    orthomclMclToGroups my_prefix 1000 < mclOutput >
      groups.txt
    ```

    where:
    `my_prefix` is a string to use as a prefix for your group IDs.
    `1000` is an arbitrary starting point for your group IDs.

    Benchmark time: 1 min.

13. Use the `orthomclSingletons` program to find proteins that are singletons, i.e., are not included in any group.

Input:
`my_orthomcl_dir/groups groups.txt`
`my_orthomcl_dir/groups goodProteins.txt`

Output:
`my_orthomcl_dir/groups singletons.txt`
Change directory to `my_orthomcl_dir/` and run this command:

```
% orthomclSingletons goodProteins.fasta groups.txt >>
  singletons.txt
```

Benchmark time: 1 min.

DOWNLOADING, INSTALLING, AND CONFIGURING THE OrthoMCL PROGRAMS

SUPPORT PROTOCOL

Use this support protocol to download, install and configure the OrthoMCL software.

1. Download the OrthoMCL software from *http://orthomcl.org/common/downloads/software/v2.0/*, and follow these instructions to install it.

   Input:
   `orthomclSoftware.tar`

   Output:
   a directory of executable programs
   a home directory for your run of OrthoMCL
   the `orthomcl.config` file.

2. Unpack the software using the following command:

   `tar -xf orthomclSoftware.tar`
   The result will look like this:

   ```
   orthomclSoftware/
   bin/
   ...
   doc/
     mysqlConfigurationGuide
     mysqlInstallGuide
     oracleConfigurationGuide
     UserGuide.txt
     orthomcl.config.template
   ```

3. Set the `PATH` as follows. The `orthomclSoftware/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`. If you don't know how to do this, ask your system administrator.

   b. call the programs using their full directory path.

4. The `orthomcl.config` file provides the OrthoMCL software with its configuration information. To edit the `orthomcl.config` file to set various values, start by creating a new directory to hold the configuration file (as well as the data and results for your run of OrthoMCL). In this document, we will call that directory `my_orthomcl_dir`.

5. In the directory `orthomclSoftware/doc/Main/OrthoMCLEngine` there is a file called `orthomcl.config.template`. Copy that file to `my_orthomcl_dir/orthomcl.config`.

6. If you are using a MySQL database, in the property examples below it is assumed that the MySQL server has a database called `orthomcl`. To accomplish this, either:

   a. create such a database by going into the server and running `create database orthomcl`.

   b. use an existing database, and in the `orthomcl.config` file change the `dbConnectString=` property (described below) accordingly.

7. Now you can edit the file, setting property values. Listed below are the properties you need to set (and suggested values when appropriate):

   `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=SimilarSequences`
   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=Ortholog`
   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 in-paralog 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 co-ortholog pairs. This is configurable so that you can run `orthomclPairs` multiple times, and compare results.
used by `orthomclInstallSchema`, `orthomclPairs`, `orthomclDumpPairsFiles`

`interTaxonMatchView=InterTaxonMatch`
`percentMatchCutoff=50`
BLAST similarities with percent match less than this value are ignored.
used by `orthomclPairs`

See Commentary, Basic Protocol 2, for more details.

`evalueExponentCutoff=-5`
BLAST similarities with e-value exponents greater than this value are ignored.
used by `orthomclPairs`

See Commentary, Basic Protocol 2, for more details.

`oracleIndexTblSpc=`
optional table space to house all oracle indexes, if required by your oracle server. Default is blank.

8. If you are using Oracle, see the included `oracleConfigurationGuide.txt`. If you are using MySQL, see the included `mysqlConfigurationGuide.txt`. If you do not have either, see the `mysqlInstallationGuide.txt` to install your own MySQL (see step 2).

9. 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*.

10. Run the `orthomclInstallSchema` program to install the OrthoMCL schema into your relational database.

    Input:
    a relational database
    `my_orthomcl_dir/orthomcl.config`

    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.

GUIDELINES FOR UNDERSTANDING RESULTS

Basic Protocol 1: Assign your proteome to OrthoMCL-DB groups

Your result will contain three files:

`orthologGroups`. This file is a mapping between your proteins and OrthoMCL-DB groups. It is tab-delimited text. (See Fig. 6.12.3 for a sample.) The columns are:

Your protein ID
The ID of the OrthoMCL-DB group it is provisionally mapped to
The ID of the protein sequence in OrthoMCL-DB that is its best hit.
The e-value mantissa of that hit

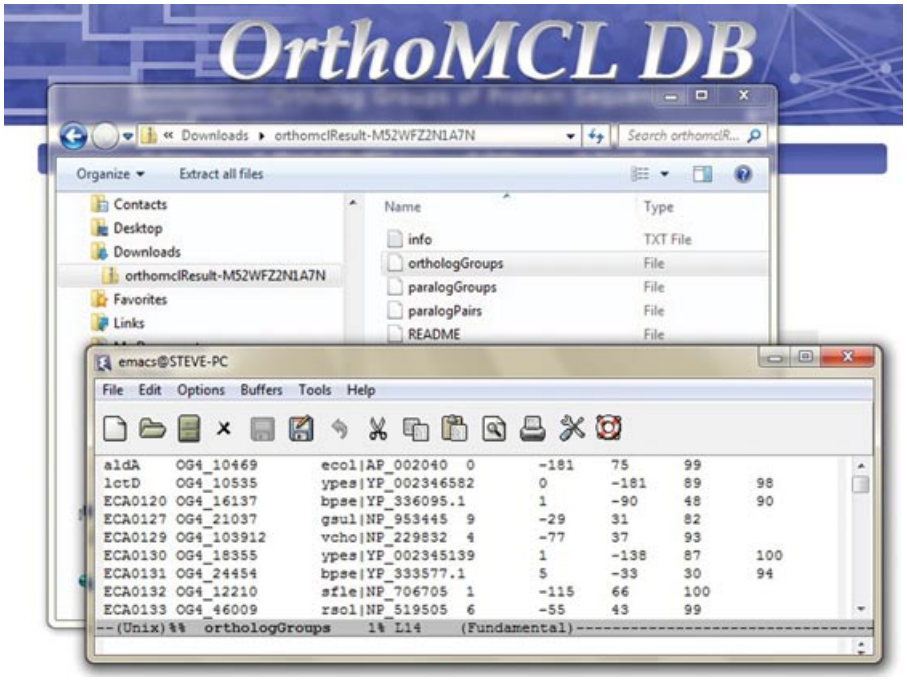

Figure 6.12.3 A proteome mapped to OrthoMCL-DB. The results are downloaded as a `.zip` file that contains five files. Shown here is the `orthologGroups` file obtained after submitting the *Erwinia carotovora* proteome (Bell et al., 2004).

The e-value exponent of that hit
The percent identity of that hit
The percent match of that hit

`paralogPairs`. This file contains reciprocal best hits (from an all-versus-all BLASTP of your proteins) among those proteins in your genome *that were not mapped* to OrthoMCL groups. The scores are normalized (see the OrthoMCL Algorithm Document).

`paralogGroups`. This file is a grouping of your proteins into provisional in-paralog groups. Only proteins that appear in the `paralogPairs` file are considered. Those pairs are passed to the MCL program, and this file is the result. Each row represents a provisional in-paralog group.

The `orthologGroups` and `paralogGroups` file represent *approximate* assignments to OrthoMCL-DB groups because the full OrthoMCL algorithm has not been run. Group assignments are made based only on the best BLAST hit. In most cases, particularly if the hit characteristics are good, this assignment is identical to what would be found if the full OrthoMCL algorithm were run. However, for weaker hits, the assignment is less reliable. A small percentage (<10%) of assignments may be different if the full algorithm were run.

Basic Protocol 2: Create ortholog groups from your proteomes using the OrthoMCL software

The primary results file for Basic Protocol 2 is the `groups.txt` file. The columns in this file are described in Guidelines for Understanding Results, Basic Protocol 1, above. The proteins in a group should not be thought of strictly as "orthologs." They are proteins that are *likely to be more closely related evolutionarily to each other than to proteins*

in other groups (or singletons). Pairs of proteins within a group may have an ortholog relationship, an in-paralog relationship, or a co-ortholog relationship. Refer to the files in the `pairs/` directory to learn about pairwise relationships before clustering into groups. These pairs relationships are more sensitive and less selective then the relationships within groups. The pairs in the `orthologs.txt` file are similar to reciprocal best hits across organisms. The pairs in the `paralogs.txt` file are similar to reciprocal best hits within an organism (where there are no better hits across organisms; these are also called "reciprocal better hits").

Clustering will be affected by separation in the (similarity) distances between proteins. For example, if you include only proteomes from evolutionarily distant organisms, distantly related proteins might end up in the same ortholog group. As you add more proteomes, more distantly related proteins would be expected to be split up among multiple groups. To a certain degree, this phenomenon is driven by the presence of multiple domains in proteins. Ortholog groups created based on sharing one protein domain may be split into groups sharing additional domains as more closely related proteomes are added.

In-paralogs in proteomes that are not closely related may be put into separate groups if there is a closely related proteome that also has that set of paralogs. For example, if the proteome set used is heavily biased phylogenetically, e.g., containing many bacteria and few eukaryotes, and if the chosen eukaryotes are distantly related, then many of the eukaryotic genes in the proteome set may remain ungrouped (singletons) or be grouped as in-paralogs. This is well illustrated in OrthoMCL-DB Version 4. It includes only one ciliate—*Tetrahymena thermophila*. When another ciliate proteome, *Paramecium tetraurelia*, is grouped (using Basic Protocol 1), many of the previously singleton or in-paralogous *T. thermophila* proteins form new groups with *P. tetraurelia* proteins.

COMMENTARY

Background Information

Basic Protocols 1 and 2 use the same fundamental approach, which is the OrthoMCL algorithm for finding protein pairs. In Basic Protocol 1, this algorithm is used to produce OrthoMCL-DB, to which your proteins are mapped. It is also used to find in-paralog pairs in your proteins. In Basic Protocol 2, this algorithm is used to find orthologs, in-paralog, and co-ortholog pairs in your proteomes.

Critical Parameters and Troubleshooting

Basic Protocol 1

A troubleshooting guide for Basic Protocol 1 is presented in Table 6.12.1.

Basic Protocol 2

There are two `orthomclPairs` parameters (see Steps 1 and 9) that can affect how distantly related proteins are included in groups. The first parameter is the `evalueExponentCutoff`. The recommended value is `−5`, which means that all pairs found by the program will have a BLAST e-value score of 1e-5 or less (in both directions). This may seem inclusive, but there are other stringent requirements for pairs (see the OrthoMCL Algorithm Document). Also, the MCL program recognizes the difference between weak and strong pairs, and avoids combining groups unnecessarily. The main effect of the weak threshold is to avoid early elimination of distant relationships. The second `orthomclPairs` parameter is `percentMatchCutoff`, with a recommended setting of `50`. This is not the same as percent identity. It requires that for two proteins to form a pair, the number of amino acids from the shorter protein that were included in any of the pair's HSPs must be at least 50% of the length of that protein. This ensures that the majority of the shortest protein is involved in the match, increasing the likelihood that the two proteins will share domains. You might consider altering either or both of these parameters to compare the resulting groups, but our experience is that these settings work well.

Advanced Parameters

Basic Protocol 2

In step 11, you specify a `−I` argument to the `mcl` program. This is the *inflation* parameter for the Markov clustering algorithm

Table 6.12.1 Troubleshooting Guide for Basic Protocol 1

Problem	Possible cause	Solution
Receiving mail that says your job failed	You submitted an invalid FASTA file	Read the FASTA format description at *http://blast.ncbi.nlm.nih.gov/blastcgihelp.shtml*. Submit a valid FASTA file
	You submitted a FASTA file with non-unique IDs per sequence	Give each sequence a unique ID, and resubmit.

and it ranges from `1` to `4`. Lower inflation values result in the inclusion of more sequences in fewer groups. The impact of inflation choice is discussed in the original OrthoMCL paper (Li et al., 2003). Tight clustering tends to prevent sequences with different functions from being clustered together, but may also separate true orthologs. OrthoMCL uses an inflation of ~1.5 to balance sensitivity and selectivity based on grouping of enzymes and their E.C. numbers.

Suggestions for Further Analysis

Basic Protocol 1

Your proteins are now mapped to OrthoMCL-DB groups. Each group has its own page at the OrthoMCL-DB Web site. To find out more about a particular protein, go the OrthoMCL-DB site and review the page for that protein's group.

Ortholog grouping can provide information regarding the evolutionary origin and functional conservation of proteins. Using the phyletic pattern searches (under the Search menu) in OrthoMCL-DB, it is possible to map the phyletic profile of the various ortholog groups that have been mapped to the proteins or proteomes of interest. The easiest way to do this is by uploading the text file containing the list of ortholog groups (obtained using Basic Protocols 1 and 2) using the upload tool in the history page for group searches. Once these uploaded ortholog groups are retrieved as a query result in the history page, then they can be combined with results from various phyletic pattern searches. This approach can be useful in identifying genes acquired by horizontal gene transfers in species of interest. For example, when a protein from a species of interest has orthologs in distantly related organisms but not in more recent ancestral species, it would suggest that the gene was acquired by the species of interest through horizontal gene transfer rather than by vertical inheritance as a result of speciation. This sort of information could be useful for identifying druggable targets in parasitic organisms.

Functional annotation based on ortholog grouping is enhanced when information regarding Pfam (*UNIT 2.5*; Bateman et al., 2004) domains and Enzyme Commission (E.C.) numbers (Webb and International Union of Biochemistry and Molecular Biology, 1992) are mapped to individual groups. In OrthoMCL-DB Version 4, a text file providing the mapping of all Pfam domains for all ortholog groups is available on the download site (accessed from the Data menu). This file also gives the frequency of occurrence of any given domain (a value of `1` indicates that all proteins in that group have that domain) which can be used to guide functional annotation of new proteomes. Although there are no data mapping EC numbers in OrthoMCL-DB Version 4, a similar mapping strategy to that of Pfam domain mapping can be implemented.

Basic Protocol 2

For a particular group of interest, create a FASTA file with that group's protein sequences. Submit the FASTA file to a multiple sequence alignment program or to Pfam analysis (see Internet Resources, below, and *UNIT* 2.5). To visualize the connectedness of the group, download and install the Biolayout program (see Internet Resources, below). To get a sense of what these tools will do, go the OrthoMCL-DB Web site and select a group of interest. These tools will be available on that group's page.

You can also use OrthoMCL groups to assign functional annotation to un-annotated proteins. If some of your proteomes have good functional assignments, you can use those assignments to assess the consistency of groups. When a group has consistent annotations, you can assign those functions to other members of the group with confidence. E.C. number (Li et al., 2003; Chen et al., 2006) and Gene Ontology (GO; see *UNIT 7.2*; also see The Gene Ontology Consortium, 2000) function assignments are robust for this sort of analysis since these ontologies are directed acyclic graphs

with less specific assignments higher up in the graph.

Acknowledgments

This work was supported by the National Institute of Allergy and Infectious Diseases at the National Institutes of Health [Award NO1-AI900038C Contract No. HHSN272200900038C to D.S.R., C.J.S. and J.C.K.]. The authors thank Deborah F. Pinney for her contributions in populating the OrthoMCL-DB database used in Basic Protocol 1.

Literature Cited

Altschul, S.F., Gish, W., Miller, W., Myers, E.W., and Lipman, D.J. 1990. Basic local alignment search tool. *J. Mol. Biol.* 215:403-410.

Bateman, A., Coin, L., Durbin, R., Finn, R.D., Hollich, V., Griffiths-Jones, S., Khanna, A., Marshall, M., Moxon, S., Sonnhammer, E.L., Studholme, D.J., Yeats, C., and Eddy, S.R. 2004. The Pfam protein families database. *Nucleic Acids Res.* 32:D138-D141.

Bell, K.S., Sebaihia, M., Pritchard, L., Holden, M.T., Hyman, L.J., Holeva, M.C., Thomson, N.R., Bentley, S.D., Churcher, L.J., Mungall, K., Atkin, R., Bason, N., Brooks, K., Chillingworth, T., Clark, K., Doggett, J., Fraser, A., Hance, Z., Hauser, H., Jagels, K., Moule, S., Norbertczak, H., Ormond, D., Price, C., Quail, M.A., Sanders, M., Walker, D., Whitehead, S., Salmond, G.P., Birch, P.R., Parkhill, J., and Toth, I.K. 2004. Genome sequence of the enterobacterial phytopathogen *Erwinia carotovora* subsp. atroseptica and characterization of virulence factors. *Proc. Natl. Acad. Sci. U.S.A.* 101:11105-11110.

Chen, F., Mackey, A.J., Stoeckert, C.J. Jr., and Roos, D.S. 2006. OrthoMCL-DB: Querying a comprehensive multi-species collection of ortholog groups. *Nucleic Acids Res.* 34:D363-D368.

Chen, F., Mackey, A.J., Vermunt, J.K., and Roos, D.S. 2007. Assessing performance of orthology detection strategies applied to eukaryotic genomes. *PLoS One*. 2:e383.

Enright, A.J., Van Dongen, S., and Ouzounis, C.A. 2002. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res.* 30:1575-1584.

The Gene Ontology Consortium. 2000. Gene ontology: Tool for the unification of biology. *Nat. Genet.* 25:25-29.

Li, L., Stoeckert, C.J. Jr., and Roos, D.S. 2003. OrthoMCL: Identification of ortholog groups for eukaryotic genomes. *Genome Res.* 13:2178-2189.

Webb, E., and International Union of Biochemistry and Molecular Biology. Enzyme nomenclature 1992: Recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology on the nomenclature and classification of enzymes. 1984th ed. Academic Press, New York.

Key References

Li et al., 2003. See above.
The original paper describing the OrthoMCL algorithm.

Chen et al., 2006. See above.
A paper describing the OrthoMCL-DB.

Chen et al., 2007. See above.
A paper comparing OrthoMCL to other approaches.

Internet Resources

http://orthomcl.org
The OrthoMCL-Db site

http://pfam.sanger.ac.uk/search#tabview=tab1
Submit a set of proteins to find Pfam domains

http://www.ebi.ac.uk/Tools/msa/clustalw2/
Submit a set of proteins for multiple sequence alignment

http://www.biolayout.org/
Download software to visualize groups using Biolayout.