USING SYNTENY IN PHYLOGENOMICS ALGORITHMS TO CLUSTER PROTEIN SEQUENCES

Christine Houry Kehyayan

A thesis
in
The Department
of
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of Doctor of Philosophy (Computer Science)
Concordia University
Montréal, Québec, Canada

September 2014

Concordia University

School of Graduate Studies

This is to certify that the thesis prepared

By: **Miss Christine Houry Kehyayan**

Entitled: **Using Synteny in Phylogenomics Algorithms to Cluster Protein Sequences**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philisophy (Computer Science)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair

Dr. M. Zahangir Kabir

_____ External Examiner

Dr. Anne Bergeron

_____ Examiner

Dr. Sabine Bergler

_____ Examiner

Dr. Clement Lam

_____ Examiner

Dr. Justin Powlowski

_____ Supervisor

Dr. Gregory Butler

Approved _____

Chair of Department or Graduate Program Director

_____ 20 _____ _____

Dr. Robin A. L. Drew, Dean

Faculty of Engineering and Computer Science

Abstract

Using Synteny in Phylogenomics Algorithms to Cluster Protein Sequences
Christine Houry Kehyayan, Ph.D.
Concordia University, 2014

With the rapid development of genome sequencing technologies, complete genomes are becoming more available and the need for computational methods for protein functional annotation is becoming more pressing. A long-standing problem in protein functional annotation is to distinguish orthologs from paralogs. Several academic efforts have recently emerged to automatically cluster proteins based on the premise that proteins in the same cluster are likely to have similar functions – or are orthologs. The effectiveness of these protein clustering algorithms is fundamental for building accurate functional annotation pipelines.

This dissertation first presents a study of the effectiveness of the similarity graph-based Markov CLuster algorithm (MCL) in detecting protein families and subfamilies when using it to cluster experimentally characterized enzymes from fungal genomes in the mycoCLAP database. Our study shows that the MCL algorithm successfully clusters proteins such that proteins in the same cluster always happen to be from the same family. However, in most cases, the MCL algorithm does not separate subfamilies. We evaluate the clusters with several cluster quality metrics, and show that these metrics can be used to spot outliers.

This dissertation then introduces SynAPhy, a novel graph-based approach for clustering proteins by leveraging the global context of complete genomes for predicting functional similarity. SynAPhy integrates genomic neighborhood information into sequence similarity for better prediction of functionally similar protein clusters. It computes the "syntenic reciprocal best hits" of proteins across genomes and uses this information to produce modified edge weight protein sequence similarity graphs. The similarity graphs are used as an input to the MCL algorithm to determine orthologous clusters across genomes. The results of applying SynAPhy on eight fungal genomes show that SynAPhy successfully generates clusters with more similar members than the MCL algorithm. However, there is no gold standard genome scale dataset to evaluate the capability of SynAPhy in generating orthologous clusters.

We introduce SynAVal, an evaluation framework that can be applied on an orthology prediction technique. SynAVal first detects paralogs within each input genome, and then detects conserved connections between genomes that are highly likely orthologs using the synteny knowledge of SynAPhy. It uses these data to identify and report confusions raised by paralogs. The results of applying SynAVal on eight fungal genomes show that SynAVal with synteny resolution can successfully resolve potential confusions raised by 9.1% of all the proteins of the eight fungal genomes, and 23.33% of the subset of the proteins of the eight fungal genomes that are likely to raise confusions.

To the memory of my beloved father
Aram Kehyayan (1949–1990)

Acknowledgments

I feel most fortunate to have Dr. Gregory Butler as my adviser. His first and foremost goal was to make my Ph.D. experience an academic and professional learning experience. I am eternally grateful for his guidance, patience, time, and funding throughout my Ph.D. years. Many thanks goes to Dr. Adrian Tsang for his valuable suggestions on this work. My committee members – Dr. Anne Bergeron, Dr. Sabine Bergler, Dr. Clement Lam, and Dr. Justin Powlowski have been very patient and constructive while I work through the challenges of this work. I am thankful to them.

My lab mates at the Concordia University – Faizah Aplop, Patricia Hanney, Jianlong Qi, Hajar Sadrarhami, Stuart Thiel, Lin Cheng, Jun Luo, and Stephan Barrett have been a great source of friendship and have provided lots of academic, professional, and personal support. I am grateful for all their support. The good people at the Centre for Structural and Functional Genomics at the Concordia University have provided lots of technical support. I am particularly thankful to Vahe Chahinian and Alexandre Beaudoin.

I would also like to thank the Department of Computer Science and Software Engineering at the Concordia University for giving me the opportunity to complete this Ph.D. Special thanks goes to Halina Monkiewicz for willingly and joyfully responding to my last minute inquiries and requests.

I owe my heartfelt gratitude to Dr. Bassem Elkarablieh, who has been my mentor and support throughout the course of this work.

Lastly, and most importantly, my profound gratitude goes to my family – my grandmother, Beatrice, my brother, Hakob, and my mother, Marie. My grandmother who we all call “Mezmama” inspired me with her passion and determination for the things that she loves. My brother has been my source of emotional support and continuous encouragement; he always shared his joy and enthusiasm on the progress of this work. Words cannot express my gratitude to my mother for all the sacrifices that she has made on my behalf. She has been there for me at every step of the way, and I could not be more grateful.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Increasing Importance of Protein Function Annotation

The genome of an organism is the complete collection of deoxyribonucleic acid (DNA) in that organism. The first complete genome was sequenced in 1995 and it was that of *Haemophilus influenzae* bacteria [FAW+95]. Thereafter, the advancements in DNA sequencing technologies led to the sequencing of genomes of thousands of organisms resulting in plethora of biological data. These data have been placed in biological databases for analysis. Three widely used publicly accessible biological sequence databases are: GenBank [BCC+13], the DNA DataBase of Japan (DDBJ) [OMK+13], and the European Molecular Biology Laboratory Nucleotide Sequence Database (EMBL-Bank), part of the European Nucleotide Archive (ENA) [LAB+11].

The genome assembly process arranges the DNA sequence of an organism to get a representation of the original chromosomes from which the DNA originated. Genome assembly is followed by a process called genome annotation that identifies genomic elements, such as genes, and annotates them with biological information including functional information. Genes encoding proteins are transcribed into messenger ribonucleic acid (RNA) molecules, which are then used to create proteins through a process called translation. Proteins are the main functional components of all cellular machinery. There are different aspects of a protein function, the most specific one is the molecular or biochemical function that is supported by a protein. For example, the molecular function of enzymes is to catalyze biochemical reactions needed for metabolism and for transporters is to move molecules from one location to another. Knowing the molecular function of a protein helps characterize its role in the cell. For example, a class of enzymes called glycoside hydrolases highly abundant in fungal species, individually or in complexes with other enzymes, when exposed to biomass resources, such as forestry residues, can help in ethanol production. The availability of complete genomes has provided access to a large number of proteins for which functional annotation is important to help determine their applications.

1.2 Accurate Protein Function Annotation

Accurate protein function annotation can be achieved by experimental characterization. However, its inherent difficulty and expense makes it challenging to scale up with the increase in protein database sizes. Computational methods have emerged in the last two decades to speed up the protein function annotation process. A commonly used computational method is based on protein sequence similarity and is referred to as Annotation Transfer by Homology (ATH). Homology is a term used to refer to proteins that share common evolutionary origins. ATH searches sequence databases for target sequences to a query sequence. If a protein sequence with a significant sequence similarity is detected, its annotation is transferred to the query. However, by virtue of evolution there exist types of homologs that do not necessarily have similar functions.

In 1970, Walter Fitch introduced the terms orthology and paralogy to distinguish types of homologs [Fit70, Fit00]. Orthologs are homologous sequences in two different species that are derived from a speciation event. Paralogs are homologous sequences in a species that are derived from a gene duplication event. Orthologs are believed to perform similar functions, while a copy of a paralogous pair is free to evolve new function. This distinction led researchers to consider orthology and paralogy relationships when comparing proteins [Fit70, Fit00].

In 1998, Jonathan Eisen introduced the term phylogenomics [Eis98] to refer to the study of genes with respect to their evolutionary relatedness. There are two phylogenomics approaches: tree-based and graph-based. Tree-based approaches gather homologs, construct a phylogenetic tree for the homologs, and reconcile the species tree of the homologs to identify the speciation and duplication events [ZE01, ZE02, KBKS06]. Graph-based approaches construct similarity graphs of the proteins and cluster the graphs such that proteins in the same cluster are similar proteins [TKL97, LSR03, RGD08]. Graph-based methods leverage the availability of complete genomes to identify the closest proteins across genomes and treat them as candidate orthologs. However, as with ATH, graph-based methods has the potential of selecting non-orthologs in the presence of highly similar paralogs and gene loss events.

In recent years, methods for orthology prediction that use gene neighborhood information have emerged. These methods leverage the availability of complete genomes but in the context of considering proteins with respect to the genomic neighborhoods of their corresponding genes. The premise behind using gene neighborhood is that in the course of evolution, conserved genomic regions across genomes contain conserved sequences with conserved adjacencies. These regions are referred to as syntenic blocks. It is expected that diverged paralogs have diverged sequences, thus they can be distinguished when compared with highly conserved orthologs with high sequence similarity. However, in the presence of highly conserved paralogs with high sequence similarity, orthologs and paralogs can not be distinguished. In these cases, integrating gene neighborhood information should help distinguish orthologs and paralogs towards accurate protein function annotation.

1.3 Thesis and Challenges

1.3.1 Thesis

Our thesis is that gene neighborhood information from complete genomes when integrated into protein sequence similarity data helps in resolving confusions raised by paralogs. This leads to more accurate protein function annotation.

We developed Synteny Algorithm in Phylogenomics (SynAPhy), where given a set of proteins in the context of their complete genomes, SynAPhy first integrates gene neighborhood information into sequence similarity data, and then applies the MCL algorithm to generate protein clusters. In order to evaluate the impact of gene neighborhood information, we developed SynAVal, an evaluate framework for an orthology prediction system that first detects the paralogs within each input genome, and then it detects conserved connections between genomes that are highly likely orthologous connections. SynAVal then evaluates orthology prediction with respect to these data.

1.3.2 Challenges

We faced several challenges while developing this work.

First, determining appropriate thresholds in protein sequence alignment algorithms. A protein sequence similarity score and its statistical significance are not objective measurements by which a pair of sequences with a specific score are assumed similar. Similarity based orthology prediction systems typically expose these thresholds as parameters with fixed default values. To determine appropriate thresholds for our analysis, we performed query and subject sequence coverage percentage and pairwise sequence percent identity analyses.

Second, graph-based orthology prediction systems are typically based on protein sequence similarity and do not allow providing gene neighborhood information as input. To leverage these systems, we represented gene neighborhood information as a weight function on top of protein sequence similarity values. This function was used to generate new similarity graphs that reflect gene neighborhood information.

Third, the absence of genome scale gold standard dataset against which to evaluate our experiments. This was one of our main challenges, as there is no publicly accessible genome scale gold standard dataset for subfamilies, orthologs, and orthologous groups. To address this challenge, we used the experimentally characterized enzymes of fungal genomes in the mycoCLAP database [MPW$^+$11]. This database, however, contains accurate annotations for a representative subset of the input genomes which we use to evaluate our clustering and classification results. To present a general evaluation of the results, we computed generic cluster quality metrics and inherent structures in input protein sequence similarity graphs that help us quantify the impact of our algorithm.

1.4 Contributions

This dissertation makes the following contributions:

- A study for the effectiveness of the graph-based Markov Clustering algorithm (MCL) for predicting protein families and subfamilies. The evaluation is performed on an experimentally characterized enzymes from fungal genomes in the mycoCLAP database. Our study shows that MCL was effective in accurately predicting protein families but the results were not conclusive for subfamilies.

- We present SynAPhy, a novel graph-based algorithm for resolving orthologs and paralogs. SynAPhy investigates gene neighborhoods of similar proteins and uses the amount of conservation in the gene neighborhoods to detect syntenic blocks in pairs of genomes. SynAPhy then uses this information to help resolve confusions raised by paralogs.

- We developed a model for representing similar proteins with corresponding genes in syntenic blocks. We used the synteny data in SynAPhy to adjust the similarity graph that is used as a seed for the MCL algorithm. Genes in syntenic blocks have higher similarity weights in the seed similarity graph directing the MCL algorithm to cluster such proteins together.

- We present SynAVal, an evaluation framework for an orthology prediction technique on clustering the proteins of genomes. The experimental results show that SynAVal can successfully resolve potential confusions raised by 9.1% of all the proteins of eight input fungal genomes, and 23.33% of the subset of the proteins of the eight fungal genomes that are likely to raise confusions.

1.5 Organization

The remainder of this dissertation is organized as follows. Chapter 2 describes the basic biological concepts that are relevant to our work. Chapter 3 presents a thorough study of using MCL to predict protein families and subfamilies for a given set of proteins. Chapter 4 presents the design and implementation of SynAPhy. It describes how synteny is computed and modeled in a similarity graph. This chapter also applies SynAPhy on eight fungal genomes. Chapter 5 presents the design and implementation of SynAVal, an evaluation framework for SynAPhy and other orthology prediction systems. Chapter 6 presents future directions concludes the dissertation.

Chapter 2

Background

2.1 Basic Concepts of Biology

2.1.1 Nucleic Acids

Nucleic acids are long biological molecules formed from smaller molecules called nucleotides. They carry the genetic information of an organism. There are two types of nucleic acids: deoxyribonucleic acid (DNA) and ribonucleic acid (RNA). The genetic information in DNA is coded with four types of characters called bases: adenine (A), guanine (G), cytosine (C), and thymine (T). The sequence of bases are arranged in two strands that form a spiral called a double helix. Each type of base on one strand is paired up with a specific type of base on the other strand to form a unit called base pair. A is paired with T and C with G. The sequence of bases on two strands have opposite directions. Strands start at the 5 prime end of the nucleotide sequence and end at the 3 prime end. DNA is found in the nucleus of eukaryotic cells and in the cytoplasm of prokaryotic cells. RNAs are usually single stranded and are assembled as a sequence of A, G, C, and uracil (U) bases. RNA molecules are synthesized on DNA templates and are used in protein synthesis in the cytoplasm.

2.1.2 Central Dogma of Molecular Biology

The genetic information on DNA sequence – or genes – of a biological system is used to synthesize messenger RNA (mRNA) molecules through a process called transcription. The information present in mRNA molecules is subsequently used to synthesize proteins through a process called translation. This flow of genetic information through transcription and translation is referred to as the central dogma of molecular biology and was first stated in 1958 by Francis Crick [Cri58].

A gene can be on either one of the two DNA strands. The strand that contains the gene is called the *template* or *sense* strand for that gene. The complementary strand is called the *nonsense* strand as it has no contribution in the synthesis of the specific mRNA. The direction of mRNA synthesis is 5 prime to 3 prime.

There is a difference in the transcription process of eukaryotic and prokaryotic cells. Because DNA is found in the nucleus of eukaryotic cells, transcription occurs in nucleus. mRNA molecules

are then transported to the cytoplasm to be translated. Transcription in prokaryotic cells occurs in the cytoplasm. Another major difference is that a eukaryotic gene has interleaved coding and non-coding segments, called *exons* and *introns*, respectively. Transcription in eukaryotic cells produces pre-mRNA strands that are subsequently converted into mRNA by removing introns and splicing exons.

The translation process synthesizes proteins from the mRNA molecules produced during transcription. Translation happens in the cytoplasm where an rRNA molecule called a ribosome attaches itself to mRNA and moves along it to produce a specific amino-acid sequence based on codon to amino-acid mapping. A *codon* is a triplet of bases coding for a specific amino-acid. There are 20 standard amino-acids. The mapping of codons to amino-acids was determined experimentally and is called the *genetic code* [CBBWT61]. There are 64 possible codons, therefore an amino-acid can be coded by more than one codon.

In our work, we use fungal organisms, which are eukaryotic organisms. SynAPhy uses start and end codons of genes to report their locations on the DNA.

2.1.3 Proteins

The primary structure of a protein is the sequence of its amino-acid molecules. Each amino-acid is represented by a letter from the English alphabet. A protein sequence is represented as a string of letters from a set of English alphabet of size 20. An important aspect of proteins is their function. The function of a protein is the role that the protein plays in a cell; it can be inferred from the three-dimensional structure of the protein, which in turn can be obtained from its primary structure [ARC$^+$54, Anf73, WP99]. A corollary to the central dogma is that proteins that share sequence similarity are expected to have similar functions. Therefore, it is important to quantify sequence similarity to determine whether proteins perform similar function or not.

Two protein sequences are said to be homologous if they share a common evolutionary origin. Homology is a qualitative inference, i.e., there is no degree of homology, proteins are either homologous or not. Sequence similarity, however, is a quantitative inference measured by sequence alignment algorithms. Homologous proteins are derived from two evolutionary events, gene duplication and gene speciation. Gene duplication occurs when regions of DNA containing genes are duplicated giving rise to duplicates in an organism [Ohn70]. Duplicates are free to evolve new functions. Gene speciation occurs when a species evolves into two diverged species giving rise to genes in the diverged species [Coo06, Coo08]. Speciation genes are likely to perform similar functions. Walter Fitch in 1970 [Fit70, Fit00] introduced the terms paralogy and orthology to distinguish the two types of homologs. Duplicated genes are paralogs and speciation genes are orthologs. Orthologous groups are groups of proteins in which every pair of proteins is ortholog.

Several other terms for orthologous and paralogous relationships have emerged throughout the literature to further classify evolutionary relationships. Of these terms are in-paralogs, co-orthologs, and out-paralogs [SK02]. Figure 1 shows a real-life example of the evolutionary relationships between human and fly gamma-butyrobetaine dioxygenase. Speciation and duplication events are shown on branches labeled with characters S and D, respectively. In-paralogs are recently diverged paralogs

that branched after a speciation event. Q9V6P0, Q9VY24, and Q9W5B5 are in-paralogs making them co-orthologs of BODG HUMAN. Out-paralogs are ancient paralogs that branched before a speciation event. Q9V6P0, Q9VY24, and Q9W5B5 are out-paralogous to Q9VDM7 and Q9NVH6. Co-orthologous relationship is often referred to as one-to-many when a single gene is a co-ortholog of in-paralogs, and many-to-many when all members of an in-paralogous subtree are co-orthologs to all members of another in-paralogous subtree.

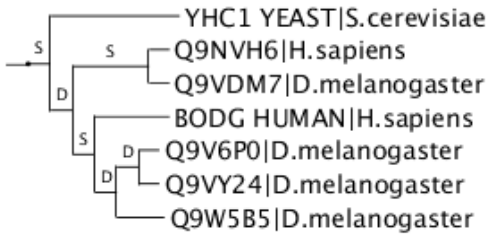

Figure 1: An illustration of the evolutionary relationships between human and fly gamma-butyrobetaine dioxygenase [SK02]

SynAPhy aims at resolving orthologs and paralogs in the presence of in-paralogs where sequence similarity is highly conserved. Out-paralogs are distantly related and are likely to have low sequence similarity when compared to both themselves and to sequences of genomes sharing a last common ancestor [SK02].

Functional Hierarchical Organization of Protein Sequences

Protein sequences are organized into a functional hierarchical classification. Margaret Dayhoff introduced the protein superfamily classification [Day74, DMBH75, Day76]. Several other classifications have been emerged since then including protein family and subfamily classifications. Protein sequences in a protein superfamily share common evolutionary origin that is observed from their structural and functional features. They do not necessarily have high sequence similarity. Protein sequences in a protein family share common evolutionary origin that is observed from structural and functional features, and have high sequence similarity. A protein subfamily is a more granular classification than a protein family. Sequences in a protein subfamily are more uniform in function than sequences in a protein family.

Domains

A protein domain is a substring of a protein sequence that can fold into a three-dimensional structure independent from the rest of the protein sequence. As such, it can have a function of its own. A protein sequence can have more than one domain, and if each performs different function, the result is a multi-functional protein sequence. For this reason, considering protein domains on their own is important in protein functional annotation. Protein domain databases exist that organize protein sequences into protein families based on their domains. Examples of commonly used domain databases are Pfam [PCE$^+$12] and Conserved Domain Database (CDD) [MBZC$^+$13].

2.2 Basic Graph and Clustering Terminology

Ortholog prediction algorithms make heavy use of graph theory when both modeling protein similarity and determining protein clusters. Below we list the graph theory concepts that we refer to throughout the document. More details on graph theory be found in [HHM08].

`Graph` A graph $G = (V, E)$ is an ordered pair of sets V and E, where V represents the set of vertices, and E represents the set of edges connecting the vertices. Two vertices v_1 and v_2 are *adjacent* if they are connected with an edge $e = <v_1, v_2>$.

`Weighed graph` A weighted graph is a graph $G = (V, E, W)$, where W represents a weight function that associates a weight label for each edge in the graph.

`Path` A path in a graph $G = (V, E)$ is an ordered subset of V.

`Connected component` A connected component C is a subgraph of G such that every pair of vertex in C is linked by a path.

`Clique` A clique is a subgraph of G such that all vertices in E are pairwise adjacent.

`bi-partite graph` A bi-partite graph is a graph whose vertices can be partitioned into two disjoint sets $S1$ and $S2$ such that every edge has one endpoint in $S1$ and the other endpoint in $S2$.

n`-partite graph` An n-partite graph is a graph with n disjoint partitions where there exist edges between the partitions and not within the partitions.

`Complete` n`-partite graph` A complete n-partite graph is a graph where every vertex of a partition is connected to every vertex in the other partitions.

We next define several metrics that we use in our evaluations.

Definition 2.1. (Graph edge density) Given a graph $G = (V, E)$, let the cardinality of E be $|E|$, and the cardinality of V be $|V|$. The edge density δ of G is

$$\delta = \frac{2\,|E|}{|V|\,(|V| - 1)}. \tag{1}$$

The range of δ is $[0, 1]$, where a δ of 0 means there are no edges in the graph, and a δ of 1 means the graph is complete, i.e., there exist an edge connecting every pair of vertices in the graph. When a cluster is represented as a graph, Equation 1 is used to calculate its edge density.

Definition 2.2. (Cluster purity with respect to classes) Given n objects to cluster, a set of clusters $\Omega = \{\omega_1, \ldots, \omega_k\}$, the set C of classes assigned to cluster members, and $|c_i|$ where $c_i \in C$ is the most frequent class in ω_i. The cluster purity is

$$purity(\Omega, C) = \frac{1}{n} \sum_{i=1}^{k} |c_i|\,. \tag{2}$$

Definition 2.3. (F-measure) Given the True Positives (TPs), False Positives (FPs), True Negatives (TNs), and False Negatives (FNs) of a test, F-measure is an aggregate metric that considers both precision and recall, where

$$precision = \frac{TP}{TP + FP}, \tag{3}$$

and

$$recall = \frac{TP}{TP + FN}. \tag{4}$$

F-measure is

$$F\text{–}measure = \frac{2 \times precision \times recall}{precision + recall}. \tag{5}$$

The F-measure value is in range $[0, 1]$. An F-measure value of 1 represents perfect classification, i.e., precision and recall values are 1. An F-measure value of 0 represents the case where precision or recall values are 0, i.e., $TP = 0$.

2.3 Bioinformatics Algorithms

2.3.1 Pairwise Sequence Alignment

A pairwise sequence alignment algorithm aligns the amino-acids of two sequences. An alignment position can be interpreted as one of the following: (1) mismatch or point mutation with either one or both of the amino-acids mutated from their last common ancestor, (2) match with either identical amino-acids or positive amino-acid substitution and (3) a gap also referred to as *indel*, with either an **in**sertion or **del**etion in one of the sequences. A match position is a conserved position with a degree of conservation. If the position is highly conserved then the position has identical amino-acids. The degree of conservation varies with different combination of amino-acid pairs. This variation is computed and populated in substitution scoring matrices with entries quantifying substituting one amino-acid with another. The two widely used substitution matrices are Point Accepted Mutation (PAM) [DSO78] and BLOcks of Amino Acid SUbstitution Matrix (BLOSUM) [HH92].

Given two sequences S_1 and S_2, and a substitution matrix, a pairwise sequence alignment algorithm aligns the S_1 and S_2, and computes a score for the alignment. A pairwise alignment can either be global or local. A global alignment aligns the entire sequences, and local alignment aligns subsequences of both sequences. One of the first pairwise sequence alignment algorithms is the Needleman-Wunsch algorithm described by Saul Needleman and Christian Wunsch in 1970 [NW70]. Needleman-Wunsch is a global dynamic programming pairwise sequence algorithm that arranges two sequences in a matrix and computes the optimal path along the diagonal of the matrix. The algorithm starts by generating a two-dimensional mxn matrix where the amino acids of sequence S_1 of length m cover the rows of the matrix, and the amino acids of sequence S_2 of length n cover the columns. The first row and the first column of the matrix are represented by gaps in the alignment position, where a gap penalty d is pre-defined. The algorithm populates the matrix with scores based on the following procedure. Let X_{ij} be the best possible score for $S_1[1...i...m]$ and $S_2[1...j...n]$. The goal is to find X_{mn}, the best score for the whole sequences not just prefixes. There are three

possible ways to end an alignment at positions i and j for which Needleman-Wunsch proceeds with the maximum $X_{ij_{\max}}$:

$$X_{i,j_{\max}} = \begin{cases} M_{S_1[i],S_2[j]} + X_{i-1,j-1} \\ d + X_{i-1,j} \\ d + X_{i,j-1} \end{cases}$$

where $M_{S_1[i],S_2[j]}$ is the substitution value of amino-acids at positions i and j in the S_1 and S_2, respectively. The base cases are $X_{i,0} = i.d$ and $X_{0,j} = j.d$.

Once the matrix is populated with scores, the optimal alignment is identified by a trace-back procedure from the bottom right corner of the matrix.

Another popular pairwise sequence alignment algorithm is the Smith-Waterman algorithm. It is a local alignment algorithm described by Temple Smith and Michael Waterman in 1981 [SW81]. Smith-Waterman resembles the Needleman-Wunsch algorithm in that it populates a matrix for two sequences and computes the optimal path along the diagonal of the matrix. Smith-Waterman has an additional score of 0 for the maximum value of $X_{i,j}$, and the base cases are equal to 0. This property allows the Smith-Waterman algorithm to align substrings of the sequences and not the whole sequences. Local alignments are more common because they capture conserved regions or domains of sequences.

2.3.2 Basic Local Alignment Search Tool

Sequence alignment algorithms are typically used to align a query sequence against all sequences in a sequence database to find similar sequences or matches. Sequence databases can contain millions of sequences making optimal alignments computationally expensive. As such, fast alignment algorithms were developed. A popular one is Basic Local Alignment Search Tool (Blast) [AGM$^+$90, AMS$^+$97]. Blast uses a heuristic algorithm to compute local alignments. The idea is that similar proteins must have short matches. Blast generates all possible short words or substrings of the query sequence. The default length of a word for protein sequences is 3 and for nucleic acid sequences is 11. The algorithm scans a sequence database for sequences that match the words with some threshold. Such matches are called *seeds*. The original Blast then extends the seeds to the right and left using ungapped alignments [AGM$^+$90]. In the following releases, Blast uses gapped alignments [AMS$^+$97]. The algorithm terminates when the score of the extended alignment falls below some threshold S. Blast reports the extended alignments or hits that have a score at least S with their statistical significance. Such hits are called High Scoring Pairs (HSPs).

Blast uses a substitution matrix to compute the scores of each HSP. Statistical analysis of Blast alignment scores have been performed in the literature [ABGW94, AG96, PJ01]. The statistical significance of a Blast score S is given by the expected number, E-value, of an HSP with a score equivalent to or better than S. The lower the E-value, the more significant the score and the alignment are.

For a pair of query and subject sequences, Blast reports all HSPs and their associated measurements. The measurements of interest for the purpose of this document are query coverage, subject coverage, percent identity, E-value, and score. Query coverage is the ratio of the length of the HSP

in the query sequence to the full length of the query sequence. Subject coverage is the ratio of the length of the hit in the subject sequence to the full length of the subject sequence. Percent identity is the percentage of identical amino-acids at the same positions in the alignment with respect to the alignment length. Score is the bit score, which is the raw score calculated from the substitution matrix normalized to parameters including the database size [AMS+97].

Definition 2.4. (all-versus-all Blast search for two sets of protein sequences P and Q) Let $P = \{p_1, \ldots, p_m\}$ and $Q = \{q_1, \ldots, q_n\}$ be two sets of protein sequences. An all-versus-all Blast search for P and Q aligns each query sequence p_i, $\forall i \in 1, 2, ..., m$ against each subject sequence q_j, $\forall j \in 1, 2, ..., n$, and each query sequence q_i, $\forall i \in 1, 2, ..., n$ against each subject sequence p_j, $\forall j \in 1, 2, ..., m$.

Definition 2.5. (all-versus-all self Blast search for a set of proteins P) Let $P = \{p_1, \ldots, p_m\}$ be a set of protein sequences. An all-versus-all self Blast search for P aligns each query sequence p_i, $\forall i \in 1, 2, ..., m$ against each subject sequence p_j, $\forall j \in 1, 2, ..., m$. Unlike Definition 2.4 in which a query sequence in a set of proteins is aligned against sequences of another set, self Blast search aligns a query protein against sequences of its own set.

2.3.3 Hidden Markov Models for Protein Sequences

Hidden Markov models (HMMs) were first described in the late 1960's [Str60] and subsequently employed in speech processing [Bak75]. In the area of speech processing, an HMM models sounds forming a word or phoneme and generates an output distribution with a high probability for the sounds of the word or phoneme it models. A satisfactory model is that which assigns high probability to the sounds of the word it models and low probability to the sounds of any other word. It was not until the late 1980's that HMMs were employed in several applications in computational biology including modeling homologous nucleotide or protein sequences [KBM+94].

Given the multiple sequence alignment of protein sequences of a protein family, the functional sites of the proteins are projected on the multiple sequence alignment as sites with conserved amino-acids. Other sites with no particular features are less conserved. Therefore, each site has a distinct probability distribution over the 20 amino-acids that measures the likelihood of each amino-acid occurring at that site of the protein family, as well as the probability of no amino-acid occurring. A multiple sequence alignment can then be modeled by a probabilistic model that captures the consensus nature of a multiple sequence alignment [KBM+94].

One widely used HMM tool is the HMMER package [Edd98]. It has a number of HMM related programs including *hmmbuild* to train HMMs and *hmmscan* to scan protein sequences against trained HMMs. We use *hmmbuild* to train subfamily HMMs and subsequently use *hmmscan* to scan protein sequences against trained HMMs in Chapter 3.

2.4 Markov Clustering Algorithm

Markov CLustering algorithm (MCL) [VD00a] is a graph-based clustering algorithm. It is based on the idea that a graph with cluster structure has many edges between members of clusters and few

edges between members of different clusters. Random walks in such graphs have high probability for paths that start and end in clusters.

The input to the MCL algorithm is a similarity graph $G = (V, E)$ of the objects to be clustered, where V is the set of objects and E is the set of weighted edges connecting the objects. Edge weights correspond to pairwise object similarity strength. The MCL algorithm starts by creating a stochastic matrix from the similarity graph. A stochastic matrix is a square matrix of nonnegative real numbers in which entries represent transition probabilities. It then simulates random walks by alternating two matrix operators, expansion and inflation. Expansion corresponds to matrix multiplication, and inflation corresponds to raising matrix entries to a power r, where $r > 1$. The resulting matrix is then re-normalized to make the matrix stochastic again. The inflation value is a parameter of the algorithm which controls cluster granularity. Higher values contribute to fine-grained clusters. Recommended values are from 1.1 to 10.0 [VD00a].

The *split/joint* distance [VD00b] between two clusterings generated by different inflation values is used to measure whether clusterings are identical or one is a sub-clustering of another. Therefore, it measures the consistency of a set of clusterings of different granularity.

Definition 2.6. (*split/joint* distance of two clusterings A and B) Let A and B be two clusterings of a set of n objects, the projection of clustering A onto clustering B and clustering B onto clustering A is

$$P_A(B) = \sum_{a \in A} \max_{b \in B} |a \cap b| \text{ and } P_B(A) = \sum_{b \in B} \max_{a \in A} |b \cap a|, \tag{6}$$

respectively. The *split/joint* distance is

$$d(A, B) = 2n - P_A(B) - P_B(A). \tag{7}$$

Definition 2.7. (Pair-value distance (d_A, d_B) of clusterings A and B) Let A and B be two clusterings of a set of objects of cardinality n, the pair-value distance (d_A, d_B) of clusterings A and B is

$$(d_A, d_B) = (n - P_A(B), n - P_B(A)), \tag{8}$$

where $P_A(B)$ and $P_B(A)$ are defined by Equation 6. The pair-value distance is useful because if any of d_A or d_B values is 0, then the corresponding clustering is a sub-clustering of the other.

Example 2.8. Let $n = 10$, and A and B be two clusterings $\{\{1\}, \{2\}, \{3\}, \{4, 5, 6\}, \{7, 8, 9\}, \{10\}\}$ and $\{\{1\}, \{2, 3\}, \{4, 5, 6, 7, 8\}, \{9\}, \{10\}\}$, respectively. In this example, $P_A(B) = 1+1+1+3+2+1$ and $P_B(A) = 1+1+3+1+1$, therefore $d(A, B) = 4$. The pair-value distance (d_A, d_B) is $(1, 3)$, this means that there is 1 rearrangement that has to be applied on A to make A a sub-clustering of B, and there are 3 rearrangements that have to be applied on B to make B a sub-clustering of A.

We use the MCL algorithm in both Chapter 3 and Chapter 4 with different inflation values. We evaluate the consistency of the clusterings with different inflation values on the same graph with the *split/joint* distance and show the pair-value distances.

2.5 Phylogenomics Approaches

Phylogenomics approaches fall into two main camps: tree-based and graph-based. In this section, we present several techniques from the two approaches.

A tree-based approach predicts the homologs for an input protein set and constructs a gene or phylogenetic tree for the homologs. It then reconciles the corresponding species tree of the input protein set with the gene tree to identify speciation and duplication events at the nodes of the gene tree. The main idea behind reconciliation is that species trees and gene trees are likely to have different topologies because of evolutionary events at the gene level such as gene duplication and gene loss. Therefore, reconciling species trees with gene trees leads to the identification of speciation and duplication events on the gene trees. Once the speciation and duplication events are identified, orthology and paralogy prediction for a pair of genes depends on the branches leading up to the genes.

Species tree and gene tree reconciliation is often performed using a maximum parsimony approach in which the most likely reconciliation is the one that minimizes the duplication events. Speciation and Duplication Inference (SDI) [ZE01] is an example technique that uses the parsimony approach. Let G be the set of nodes in the gene tree, and S be the set of nodes in the species tree. For each node g in G, let $\gamma(g)$ be the set of the corresponding species of the leaf nodes descendant from g. For each node s in S, let $\sigma(s)$ be the set of the species that are at the leaf nodes descendant from s. A mapping $M(g)$ for any node g in G is the node in S with the smallest number of species satisfying $\gamma(g) = \sigma(M(g))$. The node g in the gene tree is then labeled as a duplication event if and only if $M(g) = M(g_1)$ or $M(g) = M(g_2)$, where g_1 and g_2 are the children of g.

One key challenge of the tree-based orthology prediction approach is the availability of a reliable species tree. TreeBeST [LCR$^+$06] and Ensembl Compara [VSUV$^+$09] have worked around this challenge by not mapping those nodes of a gene tree that correspond to uncertain species tree regions. Because of the species tree limitation, other tree-based techniques have emerged in the literature that do not use species tree. Levels of Orthology From Trees (LOFT) [vdHRSvNH07] is one example technique that does not use a species tree. LOFT annotates gene tree nodes as duplication events if there exist leaf nodes descendant from the query tree nodes that are from the same species, otherwise the tree nodes are annotated as speciation events. PhylomeDB [HCDDG07] adopts a similar approach. Another challenge in tree-based orthology prediction approaches is the rooting of both gene and species trees. Resampled Inference of Orthologs [ZE02] uses the same parsimony approach as that of SDI [ZE01] but generates multiple gene trees with different rooting topologies and selects the one with the shortest tree height.

A graph-based algorithm models the orthology prediction problem as a graph and applies a graph algorithm to generate orthologous groups. An orthologous group is a group in which every pair of proteins is orthologs. Typically a graph-based algorithm requires the availability of whole genomes. For pairwise genomes, a graph-based algorithm applies a sequence alignment algorithm to find the Reciprocal Best Hits (RBHs) of the genes in the genomes. The main idea is that orthologs in two genomes diverged the least, therefore the best hit of a query gene in the other genome is a candidate ortholog of the query gene. RBHs are hits where the genes at the both ends of the hits are RBHs

when their corresponding genomes are compared. The reciprocal variant raises the confidence of orthology in cases of evolutionary events such as gene loss. Some orthology prediction algorithms put together in-paralogs with the orthologs in orthologous groups. The main idea is that in-paralogs are recently duplicated genes that maintained their ancestral function. Typically, in-paralogs are computed as the RBHs within genomes where the hits have higher similarity than they have to any other gene in within and between the given genomes. InParanoid [ÖSF+10] is one of the earliest graph-based orthology prediction algorithms for pairwise genomes that groups RBHs within and between genomes in orthologous groups.

When the input set is more than two genomes, the orthology prediction problem then becomes to identify orthologous groups with proteins from possibly all the input genomes. A graph-based algorithm for more than two genomes represents the genes as the vertices of the graph and the RBHs within and between genomes as the edges of the graph, and then applies a graph algorithm to generate orthologous groups. Cluster of Orthologous Groups (COGs) [TKL97] is a multi-organism orthology database. It is one of the earliest graph-based orthology prediction algorithms to generate orthologous groups for more than two genomes. It was initially constructed for prokaryotic genomes, but later got augmented with euKaryotic Orthologous Groups (KOGs) [TFJ+03]. Given n proteomes – the complete set of proteins expressed in a genome – the COGs algorithm starts by running an all-versus-all Blast search as defined in Definition 2.4 and finds RBHs within and between genomes. The best hits within genomes are joined to represent single entries. The proteins of the within and between RBHs are then represented as the vertices of an n-partite graph (see Section 2.2) whose edges are the average similarity scores of the RBHs. The algorithm then proceeds with finding triangles in the n-partite graph. A triangle is a subgraph with three edges connecting three vertices from three different proteomes. The idea behind a triangle is to validate the absence of non-orthologs. i.e., if the protein p from proteome P and protein q from proteome Q are connected by an RBH edge and they are both connected to the protein z from proteome Z by RBH edges, then the protein z validates that proteins p and q do not have different RBHs in proteome Z, therefore they are orthologs. The same concept holds for the other sides of the triangle. The algorithm then iteratively augments triangles that share a side and pronounces them orthologous groups. The algorithm terminates when no additional triangles exist to combine. The combined triangles are the predicted orthologous groups.

COGs and KOGs are further extended with additional genomes and are placed in the Non-supervised Orthology Groups (eggNOG) database [PFS+13]. eggNOG adopts a similar algorithm to identify the orthologous groups. However, it applies the algorithm to subsets of genomes separately. For example, fungal genomes are in the fuNOG database.

OrthoMCL [LSR03] is well established graph-based orthology prediction system that applies the Markov CLustering algorithm (MCL) on a protein sequence similarity graph. The input to the algorithm is a set of proteomes. Given a set of proteomes, the algorithm starts by running all-versus-all Blast defined in Definition 2.5 to generate alignment data, and then it processes the alignment data to create a protein sequence similarity graph. The algorithm connects two proteins in a proteome if their sequences are RBHs compared to all the other sequences in the input dataset. The algorithm connects two proteins from different proteomes if their sequences are RBHs when their

proteomes are compared. Within proteome connections are meant to represent recently diverged paralogs or in-paralogs with high sequence similarity. Between proteome connections are meant to represent orthologs. OrthoMCL also connects two proteins from different proteomes if they are connected through orthology and paralogy. Given proteins p, $p^{'}$, and q, where p and $p^{'}$ are in proteome P, and q is in proteome Q. Proteins $p^{'}$ and q are connected through orthology and paralogy if and only if p and q are RBHs when proteomes P and Q are compared, and p and $p^{'}$ are RBHs in proteome P. The algorithm then applies the MCL algorithm on the computed protein sequence similarity graph to generate orthologous clusters in which every pair of proteins is orthologous to each other. The orthoMCL database contains orthologous groups with in-paralogs.

OMA is a leading graph-based orthology prediction system. It starts with an all-versus-all global alignment using a Point Accepted Mutation (PAM) amino-acid substitution matrix [DSO78]. An entry in the PAM matrix indicates the likelihood of substituting the amino-acid of that row with the amino-acid of that column. The alignments in OMA are then refined to the best scoring alignments using a dynamic programming algorithm. The resulting score is a PAM evolutionary distance because the score is the log odds ratio of the substitutions calculated from the PAM substitution matrix. The algorithm then detects potential orthologs by identifying RBHs between proteomes. RBHs in OMA are referred to as stable pairs and are identified using the PAM evolutionary distances. Stable pairs are all pairs below some distance threshold. OMA advances stable pairs to verified pairs – or likely orthologs – if they satisfy an out-group condition. The out-group condition checks whether a stable pair has a different RBH when both elements of the stable pair are checked against a third proteome. Verified pairs from more than two proteomes form orthologous groups. At this stage orthologous groups contain paralogs. OMA proceeds with an additional step to generate orthologous groups without paralogs (i.e., each organism has at most one representative protein in an orthologous group). It represents the verified pair relationships as graphs, and implements a maximum-weight clique algorithm on this graph to compute the clique with the highest similarity score. This clique is then an orthologous cluster in which every pair of proteins is orthologous to each other. The OMA database contains orthologous groups without in-paralogs.

Table 1 summarizes the phylogenomics approaches. Altenhoff and Dessimoz in [AD12] provide a more comprehensive list of phylogenomics techniques.

Technique	Approach based	Requires	Preprocessing	Orthology inference	Reference
InParanoid	Graph	Whole genomes	BLAST scores & RBH	RBHs between pairs of genomes	[ÖSF+10]
COGs/KOGs	Graph	Whole genomes	BLAST scores & RBH & RBH	Combined triangles from RBHs	([TKL97, TFJ+03]
orthoMCL	Graph	Whole genomes	BLAST scores & RBH	MCL clusters	[LSR03]
eggNOG	Graph	Whole genomes	BLAST scores & RBH	Combined triangles from RBHs	[PFS+13]
OMA	Graph	Whole genomes	Smith-Waterman PAM distance & RBH	Cliques	[RGD08]
SDI	Tree	Gene & species tree	Parsimony	Speciated nodes	[ZE01]
RIO	Tree	Gene & species tree	Parsimony	Speciated nodes	[ZE02]
TreeBeST	Tree	Gene tree & collapsed species tree	Parsimony	Speciated nodes	[LCR+06]
Ensembl Compara	Tree	Gene tree & collapsed species tree	Parsimony	Speciated nodes	[VSUV+09]
LOFT	Tree	Gene tree	Parsimony	Speciated nodes	[vdHRSvNH07]
PhylomeDB	Tree	Gene tree	Parsimony	Speciated nodes	[HCDDG07]

Table 1: Phylogenomics approaches. This table lists the phylogenomics techniques in both tree- and graph-based approaches. The *Requires* column states the required artifacts. For graph-based techniques whole genomes are required, and for tree-based techniques gene tree and species tree are required. The *Preprocessing* column states any preprocessing steps required before the orthology prediction step. For graph-based techniques it is usually the Blast search and reciprocal best hit (RBH) computation. OMA runs a Smith-Waterman dynamic programming algorithm and computes the Point Accepted Mutation (PAM) evolutionary distances. For tree-based techniques, the preprocessing is the parsimony approach in identifying speciation and duplication events at gene tree nodes. The *Orthology inference* column indicates how the orthologs are inferred after preprocessing. A more comprehensive list can be found in [AD12].

The key challenge of phylogenomics techniques is the evaluation. There is no gold standard genome scale orthology dataset against which the techniques can be evaluated. Orthology benchmarking is difficult because the evolutionary history of complete genomes is not known. There is an ongoing effort by the "Quest for Orthologs" consortium [DGR+12] to construct a benchmark dataset of orthologs for complete genomes. The early benchmarking techniques [HHdVG06] use the conservation of protein function since orthologs are likely to have maintained their ancestral function. The conservation of function is measured by the conservation of gene co-expression, agreement of domain architecture, and agreement of protein-protein interaction connections. In addition, conservation of gene neighborhood as we describe in Section 2.6 is used in orthology benchmarking. The evaluation of conservation of function, however, is limited by the availability of functional data for complete genomes. Because orthology is an evolutionary inference, the phylogeny of the corresponding species is also used as a benchmarking technique [AD09]. However, this technique is also limited by the uncertainty of the species phylogeny. Therefore, benchmarking is bounded by the availability of the functional data or the species phylogeny for the underlying dataset.

2.6 Syntenic Block

Genomes evolve through changes in DNA sequences. Changes can be of various forms including inter- and intra-chromosomal rearrangements such as reciprocal translocation, inversion, and transposition. Figure 2 depicts toy examples of the three chromosomal rearrangements. We observe in the three rearrangement types that the genes that have not participated in the rearrangements have kept the order in which they appear on the chromosome. These regions where the rearrangements occur are evolutionary conserved regions. Conserved regions of two species have been used to measure their genomic distance. One of the earliest works towards development of genomic distance measures is that of Nadeau and Taylor [NT84]. It measures linkage conservation of human and mouse genomes based on the genetic lengths of chromosomal regions with homologous genes and the number of homologous genes mapped to these regions. At the time, genetic maps of all species were not available. For this reason, the term syntenic was introduced [SN96] to refer to genes that could be mapped to the same chromosome but for which there was no location data or linkage data. The term conserved synteny was then used to refer to chromosomal regions in two or more species with homologous genes that have conserved adjacencies. With the increase in the availability of complete genomes and their genetic maps, the terms synteny, syntenic, or syntenic block are used loosely by many researchers to mean conserved synteny.

2.6.1 Example

Figure 3 depicts a toy example of a syntenic block. A gene neighborhood of diameter d is a genomic region with d genes. Mapped neighborhoods in different genomes are referred to as a syntenic block if there are t_N homologous genes in the mapped neighborhoods. Gene neighborhood diameter d is set to 9 and the number of homologous genes t_N is set to 6. Because the mapped neighborhood or the neighborhood block has six homologous genes, the block is a syntenic block.

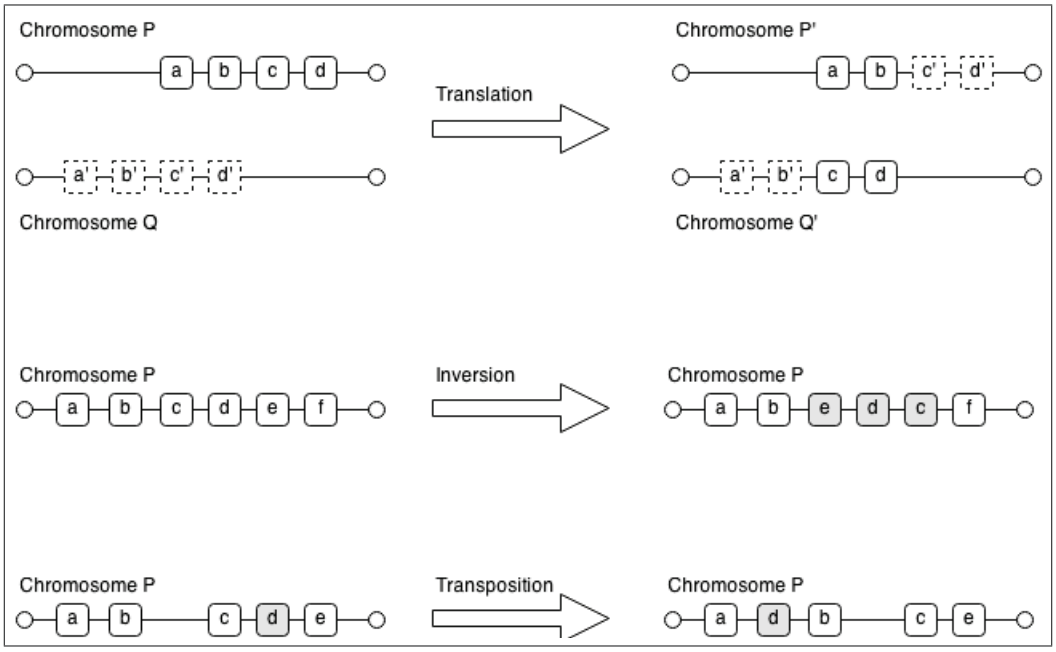

Figure 2: Inter- and intra-chromosomal rearrangements. Genes on chromosomes are depicted as boxes. There are three rearrangements present in this figure. The first one is translation. Translation is an inter-chromosomal rearrangement by which genes are exchanged between chromosomes while keeping their order. The second one is inversion. Inversion is an intra-chromosomal rearrangement by which genes reverse their order. The third one is transposition. Transposition is an intra-chromosomal rearrangement by which gene groups are shifted.

2.6.2 Synteny in Orthology Prediction Systems

Use of synteny to measure species divergence was among its early applications [ESN97]. This shed the light on leveraging gene neighborhood conservation in orthology prediction systems. Synteny is the basis of our work in SynAPhy; we boost the similarity values of protein sequences with corresponding genes in syntenic blocks, and use the MCL algorithm to cluster the sequences.

Several ortholog prediction systems have been emerged that have integrated synteny into the systems to distinguish orthologs from paralogs [CY03, GP06, WPFR07, FBB$^+$12]. We next briefly describe some of these systems.

2.6.2.1 OrthoParaMap

OrthoParaMap [CY03] distinguishes orthologs from paralogs by integrating synteny resolution and gene phylogenies. OrthoParaMap works on pairwise genomes. It first identifies syntenic blocks at the DNA level. A syntenic block in OrthoParaMap is a chromosomal diagonal that is constructed by joining chromosomes of two genomes that satisfy several conditions. It starts with a seed homologous genes from different genomes and extends the diagonal if neighboring genes are also homologous until the similarity of the diagonal drops to below a predefined threshold. OrthoParMap then identifies candidate gene families by a standard Blast search. For a gene family of interest, it constructs a

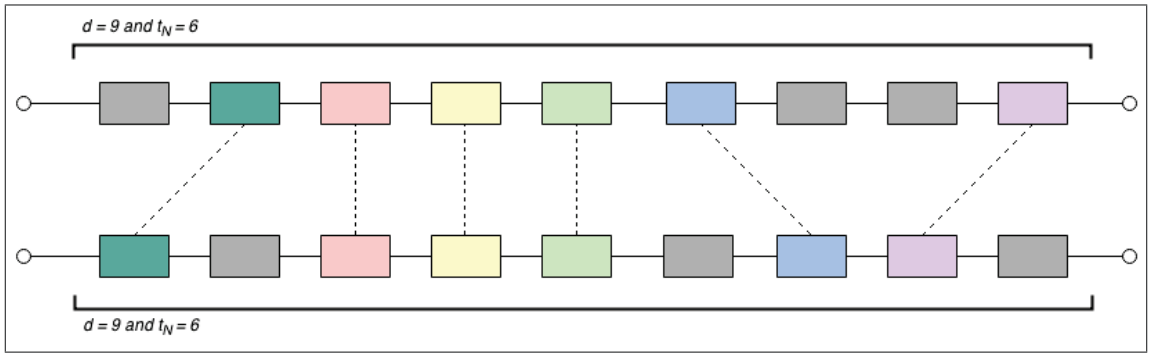

Figure 3: A syntenic block. Two chromosomal regions with genes depicted by boxes and homologous relationships between the genes of the chromosomes are depicted by dashed lines. There are six homologous relationships. The gene neighborhood diameter d is set to 9, and the minimum required number of homologous genes t_N in the mapped gene neighborhood for the neighborhood to be a syntenic block is set to 6. The block is syntenic because it has at least six homologous relationships.

phylogenetic tree and maps the chromosomal diagonal identifiers at the root of a subtree which holds the sequences of the diagonal. It then pronounces the genes in subtrees mapped to syntenic blocks as orthologs.

2.6.2.2 PhyOP

PhyOP [GP06] is a tree-based method that distinguishes orthologs from paralogs by reconciling a gene phylogenetic tree to a species phylogenetic tree. It does not rely on synteny. It uses synteny as a measure of evaluation of the predicted orthologs when compared to predicted paralogs. It constructs a gene phylogenetic tree based on a metric that is computed from the number of synonymous nucleotide substitutions per synonymous site, and reconciles the resulting gene tree to the species tree. One of the evaluation metrics adopted in PhyOP is the conservation of gene order in which adjacent orthologs in one species are neighbors in the other species. PhyOP uses the gene order conservation measures of orthologs as an evaluation metric. Although PhyOP does not use synteny in the core of ortholog detection system, it is one of the systems that relates synteny to orthology.

2.6.2.3 SYNERGY

SYNERGY [WPFR07] is a tree-based method that distinguishes orthologs from paralogs by constructing a gene phylogenetic tree for each internal node of a species tree. It integrates the amount of gene neighborhood conservation into protein sequence similarity scores that are computed based on the JTT amino-acid substitution rates [JTT92]. At each internal node of the species phylogenetic tree, SYNERGY computes candidate orthologous groups by applying a standard transitive closure algorithm on the weighted directed similarity graph generated for the proteins of the children species. The similarity graph is a weighted directed graph in which weights correspond to the combined similarity scores. For each candidate orthologous group, SYNERGY computes a gene phylogenetic tree

to label gene duplication and gene lose evolutionary events. Similar to other tree-based methods, SYNERGY is based on the assumption that a species tree of the genomes under study is provided.

2.6.2.4 PanOCT

PanOCT [FBB+12] is a graph-based method that is the closest to SynAPhy. It computes homologs and generates a weighted directed similarity graph for the homologs. It then applies a greedy agglomerative clustering to generate orthologous groups or orthologous clusters. It starts with a Blast search for the proteins of the genomes under study. It uses the Blast bit scores to compute hits between genomes that are reciprocally best hits. For each such hit, PanOCT uses a fixed gene neighborhood of diameter 11 and computes a score for the mapped neighborhoods based on several criteria of the homologous genes in the mapped neighborhoods. The criteria are (1) the strand of DNA on which the homologous genes reside, (2) chromosomal inversions as illustrated in Figure 2, and (3) distances from the RBHs. PanOCT is applied on bacterial strains because its neighborhood scores are tightened to specific criteria making it applicable to species of very close evolutionary distances or species strains.

Chapter 3

Enzyme Family and Subfamily Clustering

3.1 Introduction

The standard mechanism for protein function annotation in computational biology is Annotation Transfer by Homology (ATH). In this framework, a query protein sequence is searched against a target protein sequence database, and if a target sequence with a significant sequence similarity is detected, its annotation is transferred to the query. This gives an initial approximation of the function of the protein to prioritize further experimental or computational analysis. Determining similarity for ATH is commonly performed by Blast [AMS+97]. Another approach for protein functional annotation is when a protein is aligned against a classifier trained from a set of homologous proteins, and if the alignment is significant, the consensus annotation of the classifier is transferred to the query protein. As described in Section 2.1.3, proteins are organized into functional hierarchical classification in which the degree of similarity in each class differs to reflect the functions performed by the proteins in their classes. The widely used classifications in sequence databases are superfamily, family, and subfamily. Sequences in a superfamily share common evolutionary origin apparent in their structural and functional features. Sequences in a family share common evolutionary origin and have high sequence similarity. A protein subfamily is more granular classification than protein family. Sequences in a subfamily are more uniform in function than sequences in a family. If proteins need to be classified into protein family and subfamily, their sequences are searched against protein family and subfamily Hidden Markov Models (HMMs) described in Section 2.3.3. HMMs are trained for protein functional classes and protein sequences are aligned against the HMMs to determine membership. Because proteins in superfamilies are more diverged in sequence than in family and subfamily, typically, HMMs are trained for protein families and subfamilies.

There are specialized functional annotation pipelines that are specific to protein functions and biological classifications. There is an ongoing effort in putting together a functional annotation pipeline for fungal species [But13]. The manuscript describes in detail the steps of a functional

annotation pipeline. The initial step aligns protein sequences of interest against enzyme family and subfamily HMMs to determine membership. The database for Carbohydrate-active enzyme ANnotation (dbCAN) [YMY+12] has family HMMs for sequences in the Carbohydrate-Active enZymes DataBase (CAZyDB) [CCR+09]. An important factor influencing the quality of HMMs is the degree of similarity of the protein sequences in the training set. In this chapter, we present an evaluation of the dbCAN family HMMs. To the best of our knowledge, there has not been a comprehensive evaluation of the annotation accuracy of the dbCAN protein family HMMs. The evaluation presented in [YMY+12] is performed on the bacterial genome *Clostridium thermocellum* ATCC 27405 and the plant genome *Arabidopsis thaliana*. This analysis is described in detail in Section 3.3.

We extend our experiments to evaluate subfamily HMMs in addition to family HMMs. One key limitation of subfamily analysis is the absence of a representative publicly accessible subfamily dataset to use as a gold standard for evaluation. Unlike family HMMs, dbCAN does not have subfamily HMMs. This adds the requirement of training subfamily HMMs for our experiments.

Training subfamily HMMs itself requires a representative set of protein sequences with subfamily annotations. To determine such dataset, we experiment with two directions. First, using a clustering approach where we use the Markov Clustering algorithm (MCL) described in Section 2.4 to determine subfamilies of sequences in the mycoCLAP database [MPW+11]. We analyze the effectiveness of the MCL algorithm to provide a representative subfamily dataset. This analysis is described in detail in Section 3.4. Second, we map subfamily annotations from CAZyDB to the sequences in the mycoCLAP database and use this dataset to train our subfamily HMMs. This analysis is described in detail in Section 3.5.

3.2 Protein Sequence Resources

This section describes the protein sequence databases for family and subfamily clustering and HMM evaluation analysis.

mycoCLAP

The mycoCLAP database [MPW+11] is a gold standard database that contains Carbohydrate-Active enZymes (CAZymes) from Glycoside Hydrolase (GH), Polysaccharide Lyase (PL), Carbohydrate Esterase (CE), and Auxiliary Activities (AA) (such as oxidases) CAZyme classes for enzymes with experimental characterization of functions in the scientific literature. As of September 2013, mycoCLAP has 731 proteins in 60 different protein families from 216 different fungal organisms. Table 25 in Appendix A presents a summary of the CAZymes in the mycoCLAP database.

CAZyDB

CAZyDB [CCR+09] is a database of CAZymes that covers a wide range of taxonomic groups. As of September 2013, it has 271 CAZyme families across GH, PL, CE, and AA CAZyme classes. CAZyDB has a couple of limitations including access control over protein sequences and absence of automatic

annotation protocols. For our experiments, we are able to retrieve 9,691 fungal CAZymes, out of which 417 have exact copies. The total number of exact copies is 1,604. The remaining 8,081 from 101 different CAZyme families are used in our experiments. The exact copies are excluded because they will skew the results.

As of September 2013, CAZyDB has subfamily classifications for 13 families: GH family 5, GH family 13, GH family 30, PL family 1, PL family 3, PL family 4, PL family 7, PL family 9, PL family 11, PL family 14, AA family 1, AA family 3, and AA family 5 [ACW+12]. We extract available sequence subfamily classifications and use them to annotate mycoCLAP sequences.

dbCAN

dbCAN [YMY+12] is a database and a web server for CAZyme annotation. It is put together to expose new features, such as annotation of CAZymes, to address the limitations present in CAZyDB. It has HMMs for CAZyme families in CAZyDB. The HMMs are trained from the detected domains of CAZymes in CAZyDB. dbCAN, however, does not train subfamily HMMs from CAZyDB.

3.3 dbCAN Protein Family HMM Evaluation

We align the 8,081 fungal Carbohydrate-Active enZymes (CAZymes) we retrieved from CAZyDB against the dbCAN family HMMs using the *hmmscan* program from HMMER 3.0 [Edd98] package with the default search parameters. The *hmmscan* program has a number of search parameters including E-value and score thresholds of the alignments, and any filtering bias that has to be applied to filter out sequences. Note that dbCAN annotation pipeline also runs *hmmscan* with the default parameters. After the *hmmscan* run, we use dbCAN set thresholds to assign CAZymes to family HMMs. dbCAN uses two inclusion thresholds: E-value and HMM fraction covered. E-value is set to $1e-5$ for sequence alignment length greater or equal to 80 amino-acids, and $1e-3$ for sequence alignment length less than 80 amino acids. HMM covered fraction is the ratio of sequence alignment length to HMM alignment length. dbCAN sets this parameter to 0.3.

We evaluate the annotation accuracy of dbCAN family HMMs with confusion matrices. We compute the confusion matrix for each protein family that has at least one sequence passing the dbCAN inclusion thresholds. We take CAZyDB assigned families as a baseline reference. Therefore, for a CAZyme, we refer to the CAZyDB assigned family as its expected family, and the predicted family as its actual family. A CAZyme against a subject family is a:

- True Positive (TP) if the predicted dbCAN HMM matches both its actual and expected family,
- False Negative (FN) if the predicted dbCAN HMM does not match its actual family but its expected family is the same as the subject family,
- False Positive (FP) if the predicted dbCAN HMM matches its actual family but not its expected family, and

- True Negative (TN) if the predicted dbCAN HMM does not match its actual family and its expected family is different from the subject family.

3.3.1 Results

In this section we present the results of the annotation accuracy of the dbCAN family HMMs. Table 2 tabulates the confusion matrices of the CAZyDB protein families with respect to dbCAN family HMMs. The expected families are shown in the column labeled *Family*, the count of CAZymes classified as TP, FN, FP, and TN are shown in their respective columns. The actual families and the corresponding count of misclassified CAZymes are shown in the column labeled *Confusion*. In this table, we only show the families with F-measures defined in Definition 2.3 not equal to 1. For all the families, see Table 26 in Appendix A. Note that if a CAZyme is not classified to any dbCAN family HMM, it is counted as FN.

Out of $8,081$ input CAZymes, dbCAN family HMMs are able to classify $7,410$ out of which only two are misclassified. The first sequence is a GH family 27 misclassified to GH family 2. The second is a CE family 10 misclassified to CE family 1. The GH family 27 CAZyme is a multi-functional CAZyme with both GH family 27 and GH family 2 domains. This means that the misclassified CAZyme can be classified into both GH family 27 and GH family 2 families, and a classification into one or the other is not a confusion. The misclassified CE family 10 CAZyme has different domains than the rest of the CE family 10 sequences. Figure 4 shows the domain architectures of the CE family 10 CAZymes. The misclassified CAZYme is the one with GenBank accession number *CAA20138.1*. Unlike the other sequences in CE family 10, which have an abhydrolase_1 domain, *CAA20138.1* has a dipeptidyl peptidase IV domain. The different domains of *CAA20138.1* justify the misclassification, as not a single fungal CAZyme in CE family 10 has similar domains as that of *CAA20138.1*.

The results show that about 91% of the CAZymes are correctly classified. There are 671 CAZymes that are not classified to any dbCAN family HMM. This represents about 8.3% of the input dataset. The intriguingly high percentage of unclassified sequences directed the experiment towards analyzing the results of the thresholds set by dbCAN. We first analyze the E-value thresholds. We ignore the E-value thresholds set by dbCAN and analyze the E-values of members and non-members. Members are CAZymes with matching actual and expected CAZyDB family when aligned against dbCAN family HMMs. They are the aforementioned true positives. Non-members are CAZymes with non-matching actual and expected CAZyDB family when aligned against dbCAN family HMMs. They are the aforementioned false positives. Figure 5 displays the E-value box plots of members and non-members. The binary logarithm of $-\log_{10}$(E-value) is shown on the y-axis. The horizontal dashed line and solid line represent the $1e-3$ and $1e-5$ E-value thresholds, respectively.

We observe that all members have E-values below the set thresholds. Therefore, the E-value threshold does not have any effect on the unclassified CAZymes. Note how some non-members have low E-values. These are CAZymes with better hits to expected HMMs. This is because only two sequences are misclassified as reported in Table 2.

Family	TP	FN	FP	TN	F-measure	Confusion
AA9	178	1	0	7902	0.9972	-
GH76	169	1	0	7911	0.9971	-
GH11	170	2	0	7909	0.9942	-
GH47	161	2	0	7918	0.9938	-
GH37	77	1	0	8003	0.9936	-
GH72	206	3	0	7872	0.9928	-
GH12	68	1	0	8012	0.9927	-
GH31	133	2	0	7946	0.9925	-
GH16	361	6	0	7714	0.9918	-
GH81	59	1	0	8021	0.9916	-
GH10	110	2	0	7969	0.991	-
GH20	53	1	0	8027	0.9907	-
GH1	49	1	0	8031	0.9899	-
PL3	48	1	0	8032	0.9897	-
GH92	47	1	0	8033	0.9895	-
GH6	84	2	0	7995	0.9882	-
GH18	745	18	0	7318	0.9881	-
GH38	39	1	0	8041	0.9873	-
GH43	177	5	0	7899	0.9861	-
GH65	35	1	0	8045	0.9859	-
GH51	34	1	0	8046	0.9855	-
AA5	34	1	0	8046	0.9855	-
AA3	125	4	0	7952	0.9843	-
GH55	62	2	0	8017	0.9841	-
GH71	60	2	0	8019	0.9836	-
GH78	57	2	0	8022	0.9828	-
GH2	83	2	1	7995	0.9823	-
GH5	450	17	0	7614	0.9815	-
GH54	25	1	0	8055	0.9804	-
GH115	22	1	0	8058	0.9778	-
GH79	41	2	0	8038	0.9762	-
GH35	40	2	0	8039	0.9756	-
GH63	40	2	0	8039	0.9756	-
GH17	159	10	0	7912	0.9695	-
GH3	298	19	0	7764	0.9691	-
GH28	384	27	0	7670	0.966	-
GH32	149	11	0	7921	0.9644	GH2 : 1
CE1	54	3	1	8023	0.9643	-
GH27	77	6	0	7998	0.9625	-
PL1	115	10	0	7956	0.9583	-
GH7	333	30	0	7718	0.9569	-
GH45	31	3	0	8047	0.9539	-
GH13	266	28	0	7787	0.95	-
AA2	227	30	0	7824	0.938	-
GH26	11	2	0	8068	0.9167	CE1 : 1
GH15	97	20	0	7964	0.9065	-
CE10	4	1	0	8076	0.8889	-
GH94	3	1	0	8077	0.8571	-
GH36	17	13	0	8051	0.7234	-
GH39	3	8	0	8070	0.4286	-
AA1	94	359	0	7628	0.3437	-

Table 2: Confusion matrices of CAZyDB protein families with respect to the dbCAN family HMMs. The expected families are shown in the column labeled *Family*, the count of CAZymes classified as TP, FN, FP, and TN are shown in their respective columns. The actual families and the corresponding count of misclassified CAZymes are shown in the column labeled *Confusion*. Note that only families with F-measures not equal to 1 are shown in this table. For the full list see Table 26 in Appendix A. The rows are sorted in descending order based on F-measure.

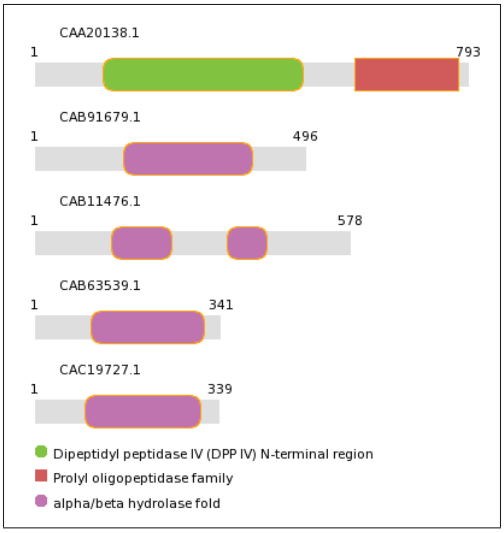

Figure 4: The domain architectures of CE family 10 fungal CAZymes. The CAZyme with GenBank accession number *CAA20138.1* is misclassified to CE family 1 HMM. We observe that *CAA20138.1* has a dipeptidyl peptidase IV domain unlike the other fungal CAZymes of CE family 10, which have an abhydrolase_1 domain. CE family 10 is no longer in CAZyDB.

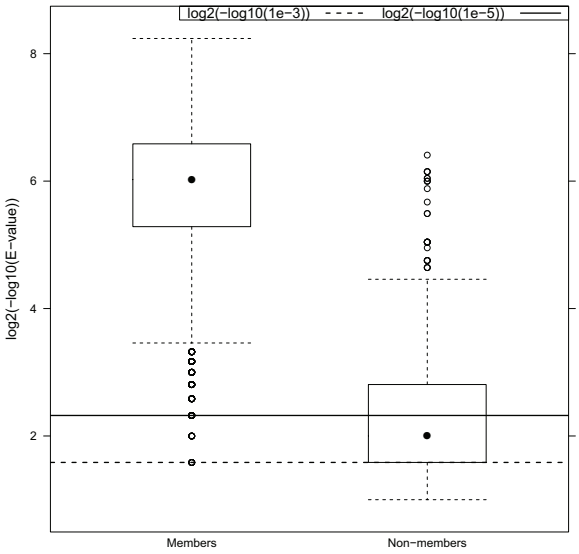

Figure 5: E-value distribution of dbCAN family HMM members and non-members. Members are CAZymes with matching actual and expected CAZyDB family when aligned against dbCAN family HMMs. Non-members are CAZymes with non-matching actual and expected CAZyDB family when aligned against dbCAN family HMMs. The binary logarithm of $-\log_{10}$(E-value) is shown on the y-axis. The horizontal dashed line and solid line represent the $1e-3$ and $1e-5$ E-value thresholds, respectively.

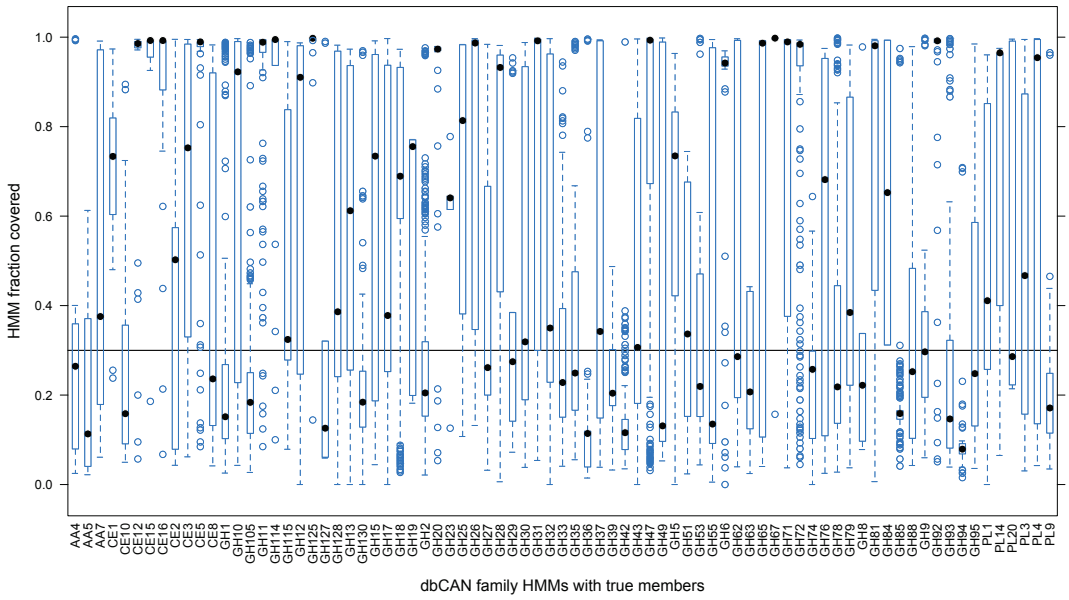

Figure 6: HMM covered distribution of member CAZymes. The x-axis shows the CAZyDB families. The y-axis shows the HMM covered fraction. The horizontal solid line represents the dbCAN set threshold for HMM covered fraction (0.3). 609 out of the 671 unclassified CAZymes fall under the cutoff line.

We next analyze the effect of the threshold set for HMM fraction covered. We report the distribution of the HMM fraction covered for members of each CAZyDB family for above and below the dbCAN set threshold. Figure 6 displays a box plot of the HMM fraction covered by the members of each CAZyDB family. The x-axis shows CAZyDB family. The y-axis shows the HMM covered fraction. The horizontal solid line represents the dbCAN set threshold for HMM covered fraction.

We observe that a large number of members have lower than 0.3 HMM covered fraction. The number of members with lower than 0.3 HMM covered fraction is 609, which is about 7.5% of the input dataset (7.5% of the 8,081 input CAZymes). The HMM covered fraction is introduced to exclude sequence fragments from being classified. To verify whether the 609 unclassified CAZymes are fragments, we analyze their sequence lengths against the average sequence lengths of their expected families. The average sequence length of a family is computed based on the correctly classified members of that family. Figure 7 displays a bar plot of the frequencies of the unclassified sequences in ranges corresponding to ratios of their sequence length to the average sequence length of their families. The bars show three segments in different bar stacks. The segments correspond to length ratio range. Black segments correspond to ratio range [0, 0.7], grey segments correspond to ratio range (0.7, 1.2], and light grey segments correspond to ratio range (1.2, 2.0]. We observe from the plot that AA family 1 has the most number of unclassified sequences, the majority of which have a length ratio greater than 0.7. About 50% of the unclassified sequences from all the families fall in the range $(0.7, 1.2]$. This in an indicator that about 50% of the unclassified sequences are

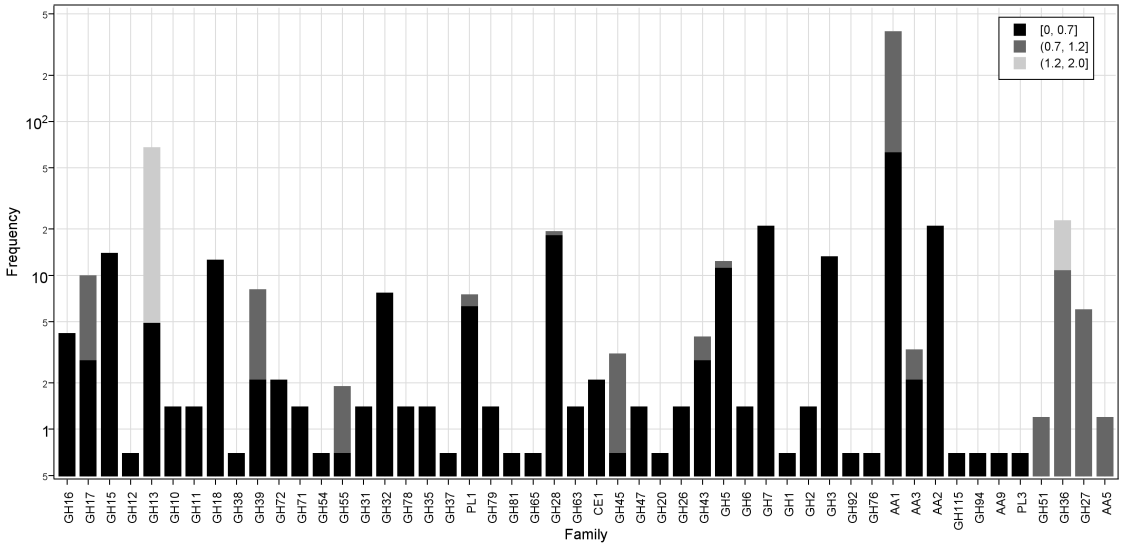

Figure 7: Protein frequencies with respect to the ratio of their sequence length to the average sequence length of their families. The x-axis shows the CAZyDB families. The y-axis shows the frequency in logarithmic scale of the unclassified protein sequences in three ranges as shown in the legend.

not fragments. We examine their dbCAN family HMM classifications without the HMM covered fraction, and notice that only two out of 609 are misclassified.

3.3.2 Discussion

The experimental results show that dbCAN family HMMs are able to correctly classify about 91% of the input dataset. This indicates that the dbCAN family HMMs are mostly satisfactory to be used in a protein functional annotation pipeline. About 8.3% of the input dataset is unclassified. We observe that the majority of the unclassified CAZymes did not pass the HMM fraction covered inclusion threshold. The HMM fraction covered inclusion threshold is introduced to exclude sequence fragments from being classified. However, we observe that about 50% of the unclassified CAZymes have comparable sequence lengths to the average sequence lengths of their expected families. This indicates that not all of the unclassified CAZymes are fragments. We examine their dbCAN HMM classifications without the HMM fraction covered threshold, and we observe that only two out of 609 unclassified CAZymes are misclassified. This indicates that the HMM covered fraction inclusion threshold has a negative effect on the sensitivity of the HMMs. The results of this experiment show that the variability of thresholds has a direct impact on the results of HMM classifications. This means that if HMM fraction coverage threshold is to be applied, a manual inspection should be followed as to confirm that there are no HMM classifiers for the unclassified sequences, because the

threshold may exclude a large proportion of the input sequences with corresponding HMM classifiers present in the pool of HMMs.

3.4 Clustering of the mycoCLAP Dataset

We next move to subfamily analysis of the mycoCLAP database. We evaluate the effectiveness of the Markov CLustering algorithm (MCL) (see Section 2.4) to cluster protein sequences in the mycoCLAP database into subfamilies. Accurate subfamily clustering would allow us to train HMMs for the subfamilies and use those HMMs for subfamily prediction in functional annotation pipelines. We use the mycoCLAP database because it is a gold standard dataset for which proteins are experimentally characterized with their biochemical activities and properties. Therefore, if satisfactory representative subfamily clusters are provided, their HMMs can be used to annotate novel genes with respect to a gold standard dataset.

The clusters generated for a set of sequences with the MCL algorithm are based on the similarity values among the sequences. The similarity values are based on the Blast hits. In addition to E-value, the Blast hits have query and subject coverage fractions, and percent identity. Some hits with good similarity scores might be binding domains and not functional domains. These hits will be clustered together because of their high similarity scores. However, their functional domains might be different. For this reason, it is important to study the effect of query and sequence coverage fractions along with similarity score to determine whether sequences clustered together are based on shorter but more similar domains or longer and more significant domains such as functional domains. We run a study to explore the possibility of generating representative subfamily clusters. In this study, we generate different input weighed similarity graphs based on query and subject coverage fractions and percent identity to pass them to the MCL algorithm. Another dimension of the MCL algorithm is the inflation parameter, which dictates the granularity of the clusters. We run another study, in which we generate different clusterings of the same graph with different inflation values and analyze the results.

3.4.1 Similarity Graphs for Markov Clustering Algorithm

We generate three weighted undirected similarity graphs for the protein sequences in the mycoCLAP dataset. The first one is based on the E-values, the second one adds the effect of query and subject coverage percentages, and the third one adds the effect of pairwise sequence percent identity. To generate these graphs, we first run all-versus-all Blast defined in Definition 2.5 on the set P of the protein sequences in the mycoCLAP database. Blast 2.2.26+ software version is used with the following calls and parameters:

>makeblastdb -in <*input_file*> **-dbtype 'prot' -out** <*database_name*>
>blastp -db <*database_name*> **-query** <*input_file*> **-out** <*output_file*> **-evalue 1e-3 soft_masking true -outfmt 6 -use_sw_tback**

makeblastdb creates a Blast database from the input file <*input_file*>, specifies its type (**-dbtype 'prot'**), and outputs it to the output file <*database_name*>. **blastp** searches protein sequences in the input file <*input_file*> against the Blast database <*database_name*>, and outputs the alignment data to the output file <*output_file*>. The E-value threshold is set to **1e-3**. The parameter **soft_masking true** soft masks or ignores low information sequence segments only in the search phase and not in the alignment phase. The default option is hard masking, which masks the low information sequence segments in both search and alignment phase. Soft masking uses the time-saving purpose of the masking while generating more accurate alignments. The parameter **-outfmt 6** sets the output format to a tabular format, and **-use_sw_tback** uses the Smith-Waterman algorithm to compute local optimal alignments as implemented in Blast 2.2.26+. The default alignment algorithm is the Blast heuristic alignment algorithm.

We define pairwise protein sequence similarity scores and edge weights as follows.

Definition 3.1. (Similarity value $bi\_sim(p, q)$ of proteins p and q) Given the all-versus-all Blast results of the set P of proteins, and proteins $p, q \in P$. Proteins p and q have a set $H = \{h\_1, h\_2, ..., h\_n\}$ of high scoring pairs (see Section 2.3.2). Then the $E\text{–}value(p, q)$ is defined as

$$E\text{–}value(p, q) = \{E\text{–}value(p, q)_{h_i} \mid h_i \in H \text{ and } h_i \text{ has the minimum } E\text{–}value \text{ in } H\}. \tag{9}$$

The similarity value of p and q is then defined as

$$bi\_sim(p, q) = \begin{cases} max(-log(E\text{–}value(p, q)), -log(E\text{–}value(p, q))), & \text{if } E\text{–}value(p, q) \neq 0 \\ & \text{and } E\text{–}value(p, q) \neq 0; \\ Max, & \text{if } E\text{–}value(p, q) = 0 \\ & \text{or } E\text{–}value(p, q) = 0. \end{cases} \tag{10}$$

where Max is a positive integer strictly larger than all values $-log(E\text{–}value(p, q))\ \forall\ p_i, q_j \in P$.

Definition 3.2. (Similarity edge weight $w_{mlec}(p, q)$ of proteins p and q) Given the all-versus-all Blast results of the set P of proteins, and proteins $p, q \in P$ where $p \neq q$, the *mlec* similarity edge weight is defined as

$$w_{mlec}(p, q) = bi\_sim(p, q) \times query\text{–}coverage \times subject\text{–}coverage, \tag{11}$$

where $bi\_sim(p, q)$ is the similarity value of proteins p and q defined in Definition 3.1, *query–coverage* is the ratio of the length of the hit to the length of the query sequence, and *subject–coverage* is the ratio of the length of the hit to the length of the subject sequence. $w_{mlec}(p, q)$ is rounded to the nearest integer.

Definition 3.3. (Similarity edge weight $w_{mleci}(p, q)$ of proteins p and q) Given the all-versus-all Blast results of the set P of proteins, and proteins $p, q \in P$ where $p \neq q$, the *mleci* similarity edge weight is defined as

$$w_{mleci}(p, q) = bi\_sim(p, q) \times query\text{–}coverage \times subject\text{–}coverage \times percent\text{–}identity, \tag{12}$$

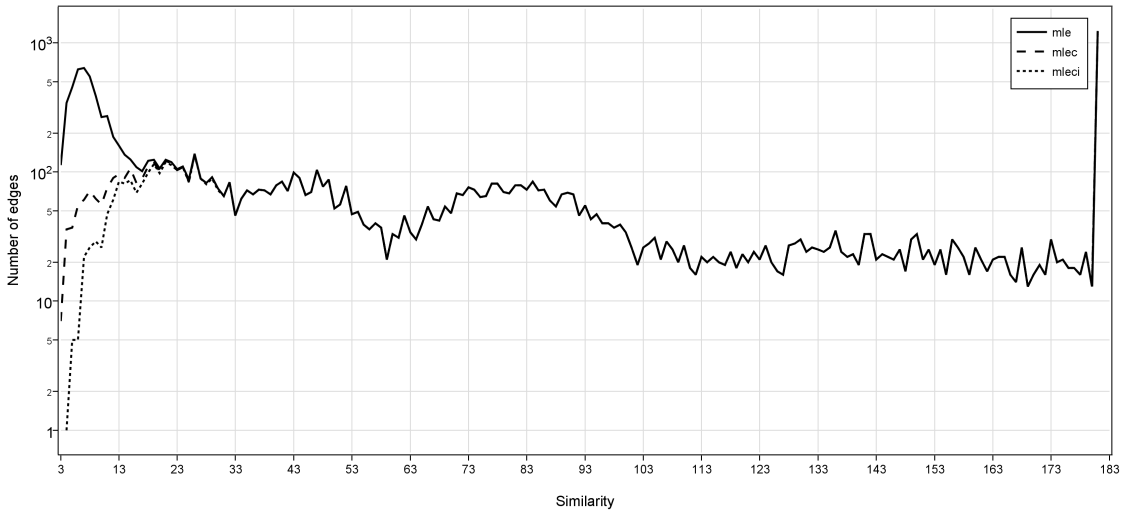

Figure 8: Edge frequencies with respect to the similarities of the edges of three protein similarity graphs. The three similarity graphs are *mle*, *mlec*, and *mleci*. The x-axis shows similarity values. The y-axis depicts the number of edges on a logarithmic scale for the *mle* graph by the continuous line, for the *mlec* graph by the dash dotted line, and for the *mleci* graph by the dotted line. Note that the similarity values of the *mlec* and *mleci* graphs are normalized with respect to their respective scalars applied on the original similarity values.

where $bi\_sim(p, q)$ is the similarity value of proteins p and q defined in Definition 3.1, *query–coverage* is the ratio of the length of the hit to the length of the query sequence, *subject–coverage* is the ratio of the length of the hit to the length of the subject sequence, and *percent–identity* is the percentage of identical amino-acids at the same positions in the hit with respect to the alignment length. $w_{mleci}(p, q)$ is rounded to the nearest integer.

We refer to the three similarity graphs as *mle*, *mlec*, and *mleci*. The edge weights of the *mle* graph are the corresponding similarity values of the vertices at the ends of edges defined in Definition 3.1. The edge weights of the *mlec* and *mleci* graphs correspond to the edge weights defined in Definition 3.2 and Definition 3.3, respectively.

We compare the three graphs in terms of the quality of the remaining and filtered edges. There are 731 proteins in the mycoCLAP database. The total number of edges in the *mle* graph is $13,077$. This number drops to $9,599$ with the *mlec* graph, and to $9,176$ with the *mleci* graph. The drop happens when the similarity values become 0 with Definition 3.2 and Definition 3.3. We first compare the edges of the three graphs in terms of similarity values. Figure 8 depicts the number of edges at different similarity values. The x-axis shows the similarity values of the edges. The y-axis depicts the number of edges on a logarithmic scale for the *mle* graph by the continuous line, for the *mlec* graph by the dash dotted line, and for the *mleci* graph by the dotted line.

We observe from Figure 8 a drop in the number of edges for similarity value range 3 to 25 in the

mlec graph when compared to the *mle* graph, as well as in the *mleci* graph when compared to the *mlec* graph. Whereas the number of edges are comparable for similarity values in the range 25 to 181 in the three graphs. This indicates that the *mlec* and *mleci* graphs are losing loosely connected edges with low similarity values. These edges could be short segments, for example, binding domains.

We next compare the edges in terms of query and subject coverage percentages to analyze whether the lost edges correspond to hits formed from short segments. Figure 9 depicts the number of edges for each query and subject coverage intervals in the *mle*, *mlec*, and *mleci* graphs in three panels with respective headings. The horizontal axes for the three panels show query coverage percentage, and the vertical axes show subject coverage percentage. The frequency in binary logarithm for a query and subject coverage percentage is color coded.

We observe from Figure 9 that *mlec* and *mleci* graphs drop a large number of edges for query and subject coverage less than 20%. This is shown in the lower left corner of the charts for the *mlec* and *mleci* graphs compared to the *mle* graph. These hits correspond to short segments because they have low query and subject coverage fractions.

We next compare the edges in terms of edge connectivity between protein families. Figure 10, Figure 11, and Figure 12 depict the inter- and intra-protein family edges for *mle*, *mlec*, and *mleci* graphs, respectively. The dots in the figures represent proteins in the mycoCLAP database. Proteins are grouped according to their families represented as isolated structures in the figures. Edges between proteins are depicted by solid lines.

We observe that the *mlec* graph in Figure 11 compared to the *mle* graph in Figure 10 drops a significant number of between protein family edges. The *mleci* graph in Figure 12 drops fewer between protein family edges.

We observe that aggregating query and subject coverage percentages and percent identity is dropping a large number of loosely connected edges in a weighted protein sequence similarity graph compared to a graph based only on E-value. The standard mechanism to consider query and subject coverages and percent identity is to introduce inclusion thresholds. The reason we performed this experiment is not to introduce additional thresholds for query and subject coverage percentage and percent identity.

3.4.2 Running MCL

We run the *mcl* program from the MCL 12-068 release [VD00a] on the *mle*, *mlec*, and *mleci* graphs with eight different inflation values in order to analyze different clusterings generated with different granularity. The eight inflation values are: 1.4, 2.0, 2.6, 3.2, 3.8, 4.4, 5.0, and 5.6. For the protein family level, we consider the clusterings with inflation value 5.6, because protein families are less granular than subfamilies and higher inflation values generate less granular clusters. In the clusterings generated from the *mle* and *mlec* graphs, we notice two clusters with sequences from multiple families. The families are GH families 5, 9, 45, and CE family 1. The sequences are connected because they share binding domains. In the clustering generated from the *mleci* graph, the clusters are pure with respect to protein families, i.e., there is no cluster with multiple families in it.

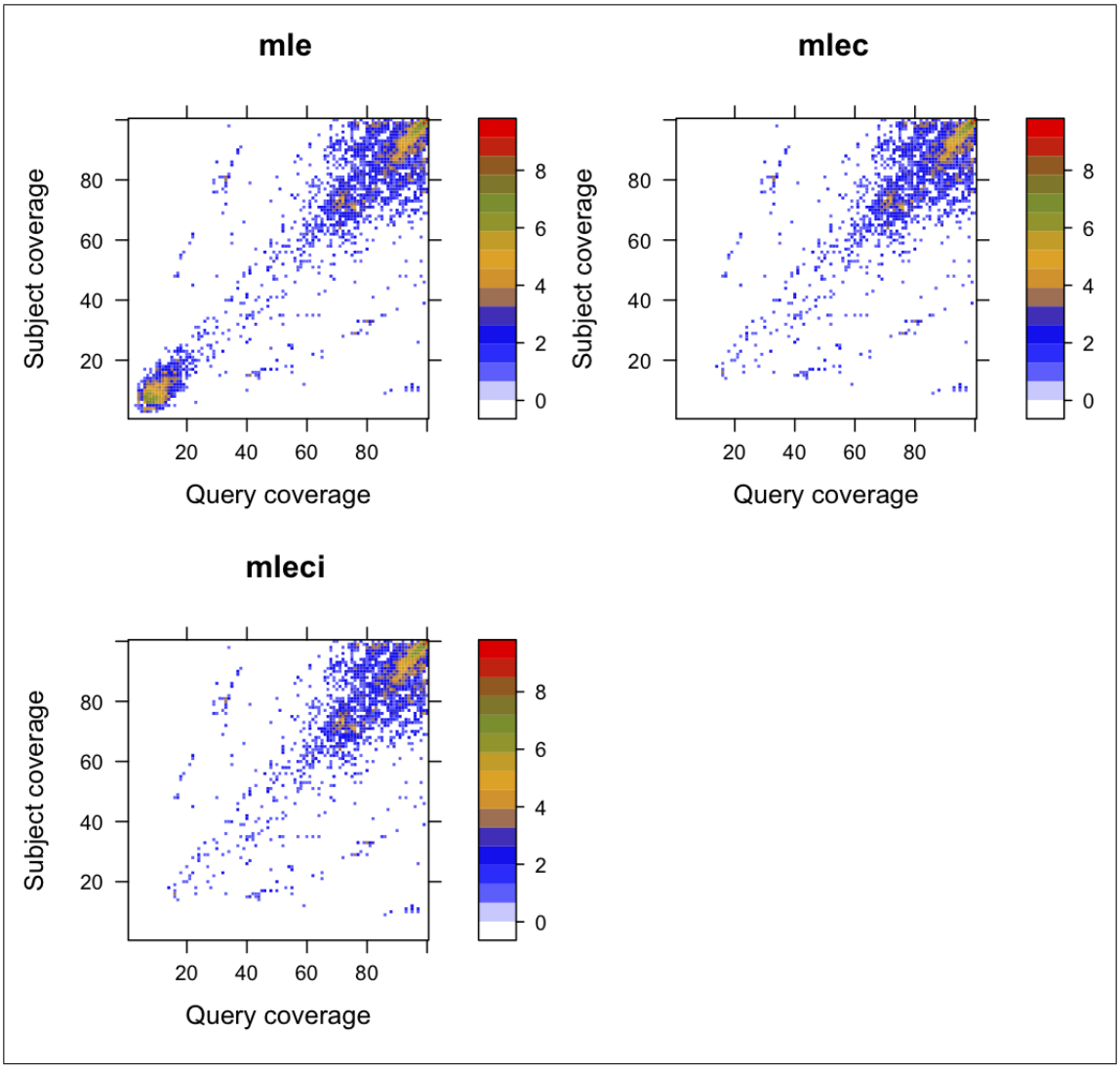

Figure 9: Edge frequencies at query and subject coverage percentages of three protein similarity graphs. The frequency in binary logarithm for each query and subject coverage percentage is color coded. The range of values and their corresponding colors are shown in the color legend to the right of each panel.

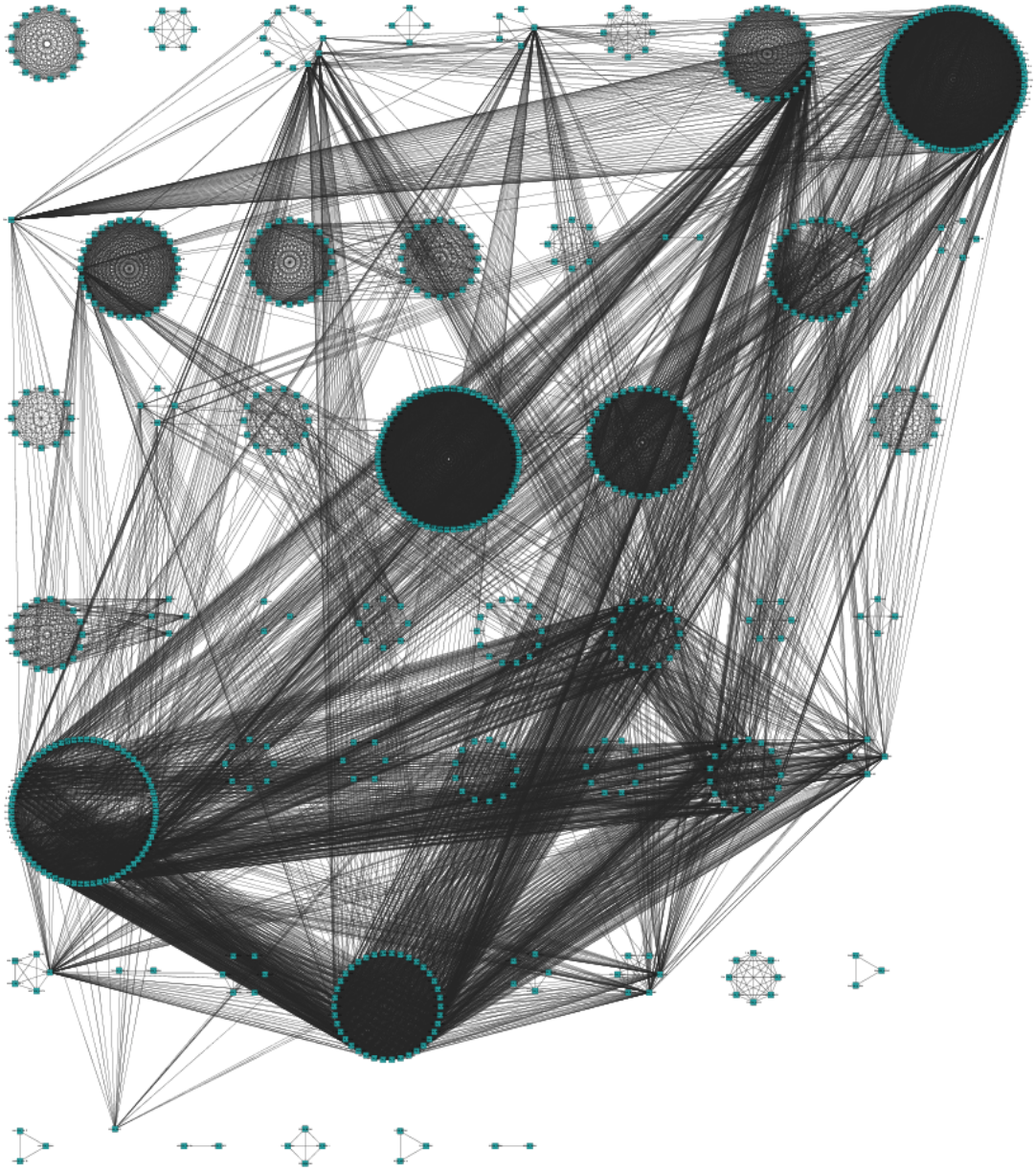

Figure 10: The *mle* protein similarity graph of the proteins in the mycoCLAP dataset. The dots represent proteins in mycoCLAP. Proteins are grouped according to their families represented as isolated structures. Edges between proteins are depicted by solid lines. The families of the groups listed row by row starting from the first row in the left to right direction: AA2, AA3, CE1, CE4, CE5, GH2, GH10, GH11, GH11_CE1, GH12, GH13, GH15, GH16, GH17, GH18, GH2, GH20, GH26, GH27, GH28, GH3, GH30, GH31, GH32, GH32_GH43, GH35, GH36, GH43, GH45, GH47, GH49, GH5, GH51, GH53, GH54, GH55, GH6, GH61, GH62, GH65, GH67, GH7, GH71, GH74, GH75, GH78, GH81, GH9, GH93, PL1, PL3, and PL4.

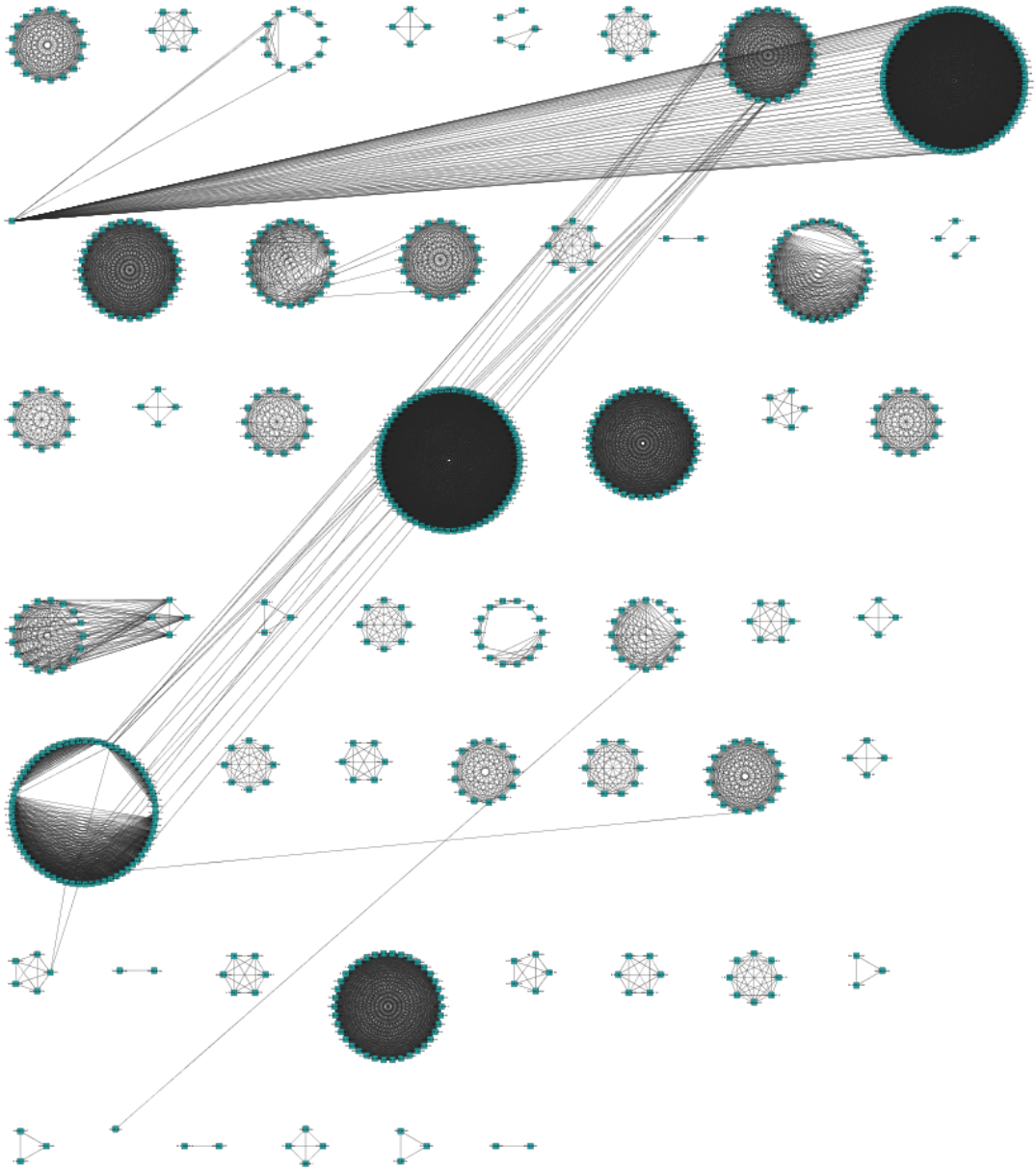

Figure 11: The *mlec* protein similarity graph of the proteins in the mycoCLAP dataset. Compared to the edge connectivity of the *mle* presented in Figure 10, *mlec* has fewer edges between protein families. The families of the groups listed row by row starting from the first row in the left to right direction: AA2, AA3, CE1, CE4, CE5, GH2, GH10, GH11, GH11_CE1, GH12, GH13, GH15, GH16, GH17, GH18, GH2, GH20, GH26, GH27, GH28, GH3, GH30, GH31, GH32, GH32_GH43, GH35, GH36, GH43, GH45, GH47, GH49, GH5, GH51, GH53, GH54, GH55, GH6, GH61, GH62, GH65, GH67, GH7, GH71, GH74, GH75, GH78, GH81, GH9, GH93, PL1, PL3, and PL4.

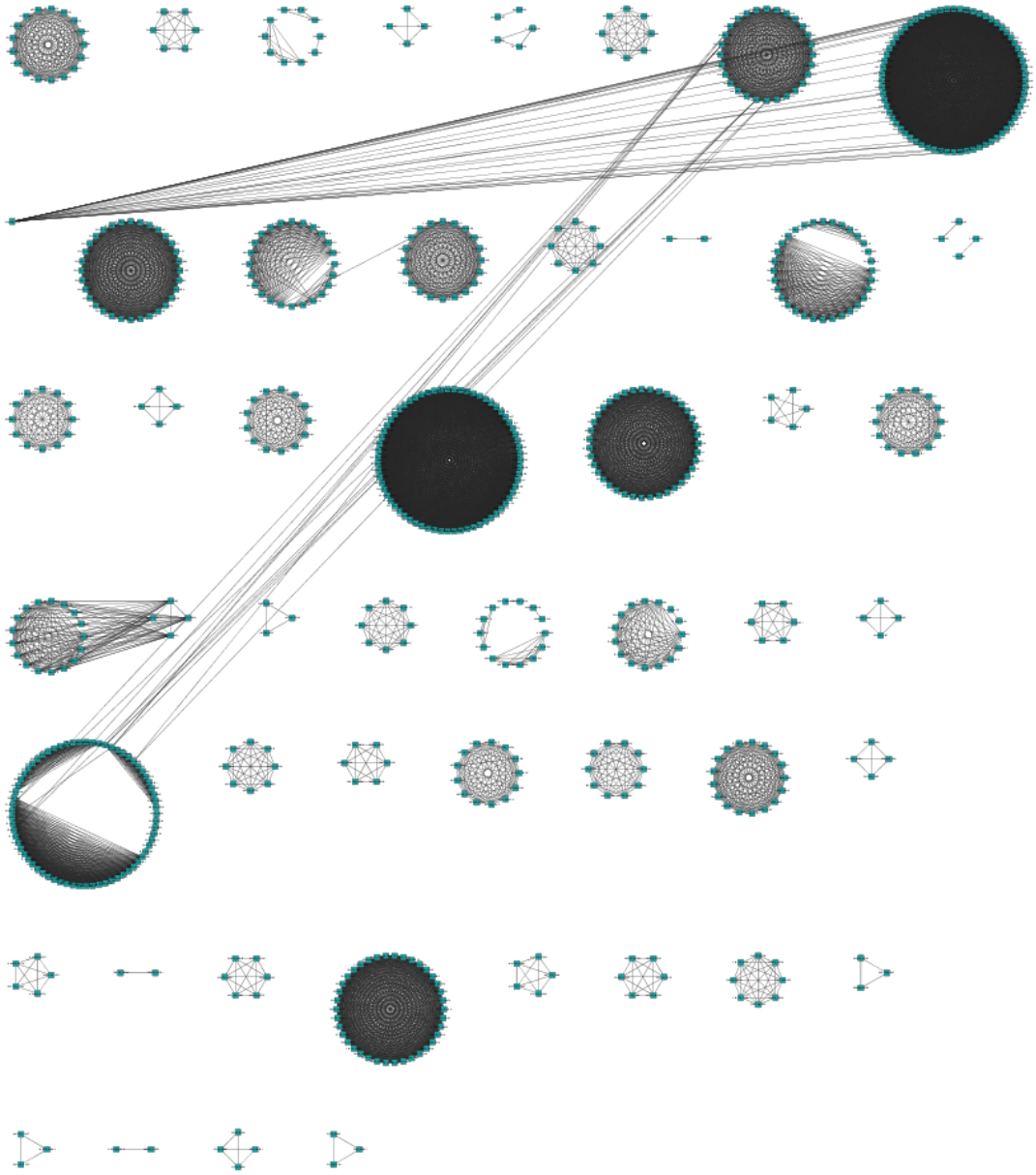

Figure 12: The *mleci* protein similarity graph of the proteins in the mycoCLAP dataset. Compared to the edge connectivity of the *mlec* presented in Figure 11, *mleci* has fewer between protein families edges, however, it does not drop them all. The families of the groups listed row by row starting from the first row in the left to right direction: AA2, AA3, CE1, CE4, CE5, GH2, GH10, GH11, GH11_CE1, GH12, GH13, GH15, GH16, GH17, GH18, GH2, GH20, GH26, GH27, GH28, GH3, GH30, GH31, GH32, GH32_GH43, GH35, GH36, GH43, GH45, GH47, GH49, GH5, GH51, GH53, GH54, GH55, GH6, GH61, GH62, GH65, GH67, GH7, GH71, GH74, GH75, GH78, GH81, GH93, PL1, and PL3.

The *mleci* graph compared to *mle* and *mlec* has the least number of loosely connected edges in terms of low similarity values as observed in Figure 8, has higher query and subject coverage percentages as observed in Figure 9, and fewer inter-family edges as observed in Figure 12. In addition, the clusters are pure with respect to protein families. For this reason, we choose the *mleci* graph to proceed with protein subfamily clustering.

We use four evaluation metrics for clusters generated with the MCL algorithm. The first one is the cluster density defined in Definition 2.1. The second one is the average of the normalized edge weights defined in Definition 3.4, the third one is the coefficient of variation of the average of the normalized edge weights defined in Definition 3.5, and the fourth one is the cluster purity defined in Definition 2.2.

Definition 3.4. (The average of the normalized edge weights $\overline{w}$ of a cluster C) Given the all-versus-all Blast results of the set P of proteins, a cluster C, and the set of edges $E = \{e_1, \ldots, e_n\}$ in cluster C, the average of the normalized edge weights $\overline{w}$ is

$$\overline{w} = \frac{\sum_{i=1}^{|E|} w(e_i)}{|E| \times Max}, \tag{13}$$

where $|E|$ is the cardinality of the edges in C, $w(e)$ is the similarity weight of the edge e, and Max is a positive integer strictly larger than all values $bi\_sim(p_i, p_j)$ (see Definition 3.1) for all $p_i \in P$ and $p_j \in P$. The range of $\overline{w}$ is $[0, 1]$, where a $\overline{w}$ of 0 represents a non-similar cluster, and a $\overline{w}$ of 1 represents a perfectly similar cluster.

Definition 3.5. (The coefficient of variation c_v of the average of the normalized edge weights of a clustering) Given a set of clusters and their average of the normalized edge weights, the coefficient of variation c_v is

$$c_v = \frac{\sigma}{\mu}, \tag{14}$$

where σ is the standard deviation of $\overline{w}$, and μ is the mean of $\overline{w}$. The lower the c_v value, the less variability there is in the weights.

We present the results of the MCL runs at different inflation values on the *mleci* graph in Table 3. The runs are given by column labeled i. The inflation values are given by column labeled I. The distance of consecutive clusterings defined in Definition 2.7 is given by column labeled $d(C_i, C_{i+1})$. This distance measure indicates the number of rearrangements that have to be applied on the corresponding clustering to obtain a sub-clustering of the other. The minimum, maximum, mean, and standard deviation over all intra-cluster densities defined in Definition 2.1 are given by columns labeled δ_{min}, δ_{max}, δ_{μ}, and δ_{σ}, respectively. The minimum, maximum, mean, and standard deviation over all intra-cluster average of the normalized edge weights defined in Definition 3.4 are given by columns labeled $\overline{w}_{min}$, $\overline{w}_{max}$, $\overline{w}_{\mu}$, and $\overline{w}_{\sigma}$, respectively. The minimum, maximum, mean, and standard deviation over all intra-cluster edge weight variation defined in Definition 3.5 are given by columns labeled $c_{v_{min}}$, $c_{v_{max}}$, $c_{v_{\mu}}$, and $c_{v_{\sigma}}$, respectively.

We observe from the column (C_i, C_{i+1}) of Table 3 that there is a consistency in the *split/joint* pair-value distances of consecutive runs of the MCL at different inflation values, i.e., the clusterings

i	I	$d(C_i, C_{i+1})$	δ_{min}	δ_{max}	δ_{μ}	δ_{σ}	$\overline{w}_{min}$	$\overline{w}_{max}$	$\overline{w}_{\mu}$	$\overline{w}_{\sigma}$	$c_{v_{min}}$	$c_{v_{max}}$	$c_{v_{\mu}}$	$c_{v_{\sigma}}$
1	1.4	(67, 0)	0.8	1.0	0.99	0.04	0.01	0.94	0.39	0.22	0.0	2.17	0.55	0.47
2	2.0	(18, 1)	0.8	1.0	0.99	0.04	0.01	0.98	0.43	0.22	0.0	2.17	0.45	0.43
3	2.6	(11, 1)	0.8	1.0	0.99	0.04	0.01	0.98	0.45	0.23	0.0	2.17	0.42	0.41
4	3.2	(11, 1)	0.8	1.0	0.99	0.04	0.01	0.98	0.46	0.23	0.0	2.17	0.4	0.41
5	3.8	(7, 1)	0.8	1.0	0.99	0.03	0.01	0.98	0.47	0.22	0.0	2.17	0.39	0.41
6	4.4	(7, 3)	0.8	1.0	0.99	0.03	0.01	0.98	0.47	0.22	0.0	2.17	0.38	0.4
7	5.0	(8, 1)	0.8	1.0	0.99	0.03	0.01	0.98	0.47	0.22	0.0	2.17	0.38	0.4
8	5.6	-	0.8	1.0	0.99	0.03	0.01	0.98	0.48	0.22	0.0	2.17	0.37	0.4

Table 3: The runs are given by column labeled i. The inflation values are given by column labeled I. The distance of consecutive clusterings defined in Definition 2.7 is given by column labeled $d(C_i, C_{i+1})$. The columns prefixed with δ show intra-cluster edge density data. The columns prefixed with $\overline{w}$ show intra-cluster normalized edge weight data. The columns prefixed with c_v show intra-cluster edge weight variation data.

with higher inflation values are sub-clusterings of the ones with lower inflation values. Consider entry $d(C_i, C_{i+1})$ of run 1 $(67, 0)$. The entry has the number of rearrangements that have to applied on corresponding clusterings to obtain the other. The first value of the tuple is the distance of the clustering at inflation value 1.4 to the intersection of the clusterings at inflation values 1.4 and 2.0, and the second value of the tuple is the distance of the clustering at inflation value 2.0 to the intersection of the clusterings at inflation values 1.4 and 2.0. This means there are 67 rearrangements in the clustering at inflation value 1.4 to obtain the intersection of the clusterings at inflation values 1.4 and 2.0, and 0 rearrangements in clustering at inflation value 2.0 to obtain the intersection of the clusterings at inflation values 1.4 and 2.0. This trend converges at higher inflation values.

Intra-cluster edge density data of all the clusterings are satisfactory with a mean of 0.99, and a standard deviation of 0.03 and 0.04. This indicates that the majority of the clusters are dense clusters in which the number of edges is close to the number of vertices.

Intra-cluster average of the normalized edge weights, however, is not satisfactory. This indicates that there are clusters with low edge weights. The minimum for all the clusterings is 0.01 as shown in the column labeled $\overline{w}_{min}$. On the other side of the spectrum, there are clusters with high edge weights. This is shown in the column labeled $\overline{w}_{max}$. However, we observe that the means for all the clusterings as shown in the column labeled $\overline{w}_{\mu}$ are lower than 0.5. The variability of the intra-cluster edge weights are shown in the columns prefixed with c_v. The minimums for all the clusterings as shown in the column labeled $c_{v_{min}}$ are 0. This means there is no variability in the edge weights in these clusters. The maximums for all the clusterings as shown in the column labeled $c_{v_{max}}$ are 2.17. This is relatively a high value compared to 0. Note that the higher the c_v, the greater the variability in the given values. This is suggestive of clusters with very high and very low edge weights. The edge weight variation means are decreasing with higher inflation values. This indicates intra-cluster edge weights are less dispersed with higher inflation values.

The combined analysis of the three metrics allows the identification of good and bad clusters. We present all the clusters of the *mleci* graph at MCL inflation value 5.6 in Table 27 in Appendix A. More than 50% of the clusters in this table have $\overline{w}$ of less than 0.5. This result did not motivate us

CAZyDB subfamily	CAZyDB count	mycoCLAP count	Enzymatic activity
GH5_5	108	21	endoglucanase
GH13_1	112	17	alpha-amylase
GH5_9	175	13	exo-1,3-beta-glucanase
GH5_7	169	12	beta-mannanase
GH5_4	32	7	endoglucanase
PL1_4	98	2	pectin lyase
GH30_3	12	2	endo-1,6-beta-glucanase
GH30_5	4	1	galactanase
GH30_7	13	1	xylanase
GH13_40	60	1	alpha-glucosidase
GH5_2	127	1	bifunctional endoglucanase/xylanase
GH5_15	15	1	endo-1,6-beta-glucanase
GH5_16	17	1	galactanase
PL1_7	32	1	pectate lyase
PL3_2	163	1	pectate lyase
PL4_1	17	1	rhamnogalacturonan lyase
PL4_3	14	1	rhamnogalacturonan lyase
AA3_2	124	1	aryl-alcohol oxidase

Table 4: mycoCLAP sequences in CAZyDB subfamilies. CAZyDB subfamilies are given in the column labeled *CAZyDB subfamily*, the number of mycoCLAP sequences assigned to CAZyDB subfamilies is given in the column labeled *Count*. The enzymatic activity of a sequence as in mycoCLAP annotation is given in the column labeled *Enzymatic activity*. The rows are sorted in descending order based on the *mycoCLAP count*. Note that CAZyDB subfamilies with more than one sequence have the same enzymatic activity as annotated in the mycoCLAP database.

to use the MCL clusters as representative of subfamily clusters for the mycoCLAP database to be used in a protein functional annotation pipeline.

3.5 Protein Subfamily Hidden Markov Models

The cluster quality metrics indicate that MCL clusters are not good representatives of subfamilies. In order to evaluate whether subfamily HMMs trained from mycoCLAP sequences are satisfactory to be used in a functional annotation pipeline, we train subfamily HMMs with mycoCLAP sequences that have well studied subfamily classifications in CAZyDB. Out of 731 sequences in the mycoCLAP database, only 85 sequences from seven families are classified into 18 well studied subfamilies in CAZyDB. Table 4 shows the distribution of these sequences in CAZyDB subfamilies. The column showing the enzymatic activities of the sequences for each CAZyDB subfamily indicate that mycoCLAP sequences in CAZyDB subfamilies are pure with respect to function. We therefore train HMMs with the mycoCLAP sequences in these subfamilies. Note that we only consider those subfamilies with more than one sequence to train the HMMs.

We run CAZyDB sequences that are classified into subfamilies against the trained HMMs. We leave out the training set used to train the HMMs. Out of $8,081$ fungal CAZymes we retrieved from CAZyDB, $4,024$ are assigned into subfamilies. Table 5 shows the confusion matrices of the subfamilies. We observe that four out of seven subfamilies have F-measure value 1. This means their

CAZyDB subfamily	TP	FN	FP	TN	F-measure	Confusion
GH5_5	87	0	0	3,916	1	-
GH13_1	95	0	0	3,912	1	-
GH5_7	157	0	0	3,855	1	-
GH30_3	10	0	0	4,012	1	-
PL1_4	79	17	0	3,926	0.90	-
GH5_9	155	7	2	3,849	0.97	-
GH5_4	0	25	0	3,992	indeterminate	GH5_9: 2

Table 5: Confusion matrices of HMMs trained from mycoCLAP sequences. The identifier of the set of sequences from which HMMs are trained is given in the column labeled *CAZyDB subfamily*. The identifiers are of form *familyID_subfamilyID*, where the *subfamilyID* correspond to CAZyDB subfamily identifiers. The count of the true positives, false negatives, false positives, and true negatives are given in the columns labeled TP, FN, FP, and TN. The F-measure is given in the column labeled F-measure. The number of the misclassified sequences together with the misclassified HMMs is shown in the column labeled *Confusion*.

HMMs achieve satisfactory classification. Subfamilies PL1_4, GH5_9, and GH5_4 have a number of sequences that are not classified. We count unclassified sequences as FNs, which results in a drop in the F-measure.

We observe that subfamily HMMs trained from relatively small number of sequences are satisfactory. However, we cannot draw any conclusions from this set, as this set is a small set with low subfamily diversity within a family. In this set, only GH family 5 has more than one subfamily, and we notice a confusion between GH5_9 and GH5_4.

3.6 Conclusion

This chapter presents a family and subfamily HMM evaluation for classifying fungal CAZymes. Our experiments show that existing dbCAN CAZyme family HMMs are satisfactory, i.e., the HMMs achieve good F-measures with the majority of them scoring greater than 0.85. The F-measure results indicate that the existing dbCAN family HMMs can be used in a protein functional annotation pipeline and there is no need to train family HMMs from the experimentally characterized sequences in the mycoCLAP database.

There are no existing CAZyme subfamily HMMs in the literature to evaluate their potential use in a protein functional annotation pipeline. For this reason, we clustered the mycoCLAP database into potential subfamily clusters using the MCL algorithm and evaluated their results. The evaluation was conducted with cluster quality metrics. The metrics successfully pointed out outliers in the results, and indicated that while some clusters contain highly similar members, others do not. This indicates that these clusters may not be subfamily clusters in which every sequence in a cluster is in the same subfamily. This motivated us to use the very few sequences in the mycoCLAP database that are classified into well studied CAZyDB subfamilies to train subfamily HMMs and evaluate their results. We aligned all CAyDB sequences that are classified into well studied subfamilies against the subfamily HMMs. The subfamily HMMs were satisfactory with reported F-measures of greater than 0.9. However, there were not enough subfamily variety within specific families to evaluate whether

the subfamily HMMs trained from small number of sequences are satisfactory to be used in a protein functional annotation pipeline.

We conclude that as more subfamily data emerges, better experiments can be conducted with subfamilies as to how many sequences are required to train satisfactory HMMs, and whether only sequences from a specific biological kingdom are needed to train satisfactory subfamily HMMs.

Chapter 4

The SynAPhy algorithm

4.1 Introduction

This chapter presents SynAPhy (Synteny Algorithm in Phylogenomics), a synteny based algorithm to predict orthologs. The algorithm is based on protein sequence similarity and gene neighborhood conservation. It showcases proteins with significant sequence similarity and conserved genome contexts. The idea is that in the course of evolution homologous associations are maintained in conserved genomic regions. As such, speciation genes – or orthologs – are expected to reside in conserved genomic regions. We apply SynAPhy on eight well-studied fungal genomes, and compare the results to that of OrthoMCL [LSR03] and Orthologs MAtrix project (OMA) [RGD08]. We chose OrthoMCL and OMA because they are both graph-based phylogenomics approaches and their algorithms are available for download.

4.1.1 Introduction to the SynAPhy Algorithm

In this section, we present a brief overview of the SynAPhy algorithm. Figure 13 shows the SynAPhy components and illustrates the relationship between them. SynAPhy represents the core of our orthology prediction framework. It is comprised of three main components: Reciprocal Best Hit (RBH) detector, syntenic RBH detector, and synteny injector. The input to SynAPhy is the protein sequence similarity graph generated from all-versus-all Blast searches of a set of proteomes. The output is a synteny-similarity graph that is used as a seed for the MCL algorithm to generate protein clusters.

Reciprocal best hits are not a new concept, they are used in OrthoMCL, OMA, and other orthology prediction algorithms [TKL97, ÖSF+10, PST+12]. RBHs are considered candidate orthologs. The idea is that orthologs are the closest proteins when their proteomes are compared because their last common ancestor diverged by a speciation event after a duplication event. Whereas paralogs diverged by a duplication event prior to a speciation event. However, in the presence of recently diverged paralogs (i.e., in-paralogs that diverged after a speciation event) where sequence similarity is highly conserved, RBHs may introduce erroneous orthologous relationships. Other instances of

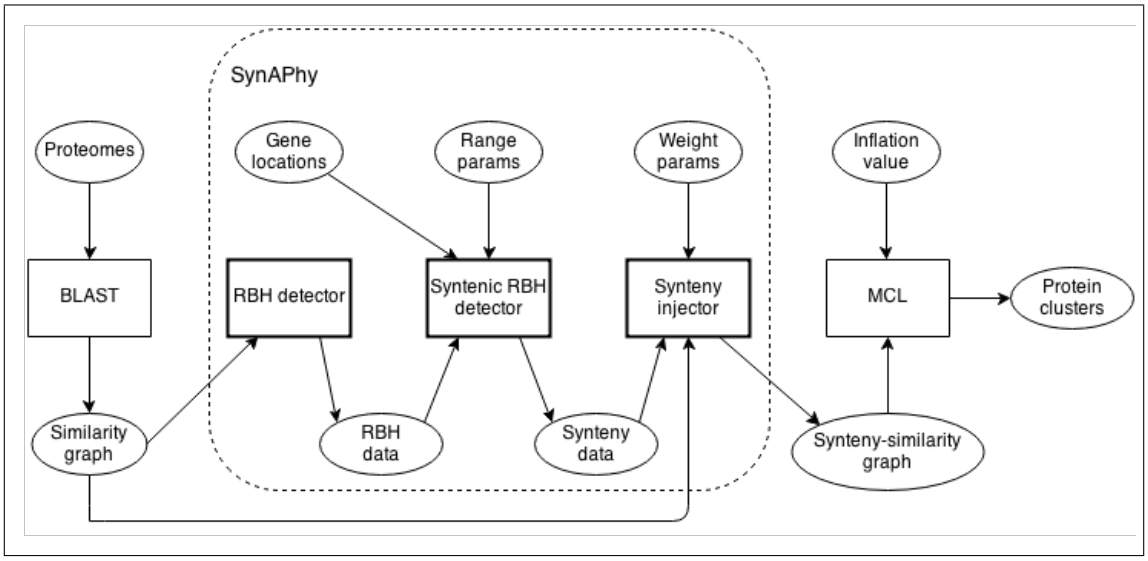

Figure 13: SynAPhy components. SynAPhy has three main components: RBH detector, syntenic RBH detector, and synteny injector. They are contained by the dashed rectangle and are depicted as bold border-lined rectangles. The input and output of the components are depicted as ellipses. The input to SynAPhy is a protein sequence similarity graph generated from all-versus-all Blast searches. The output of SynAPhy is a synteny-similarity graph that is used as a seed for the MCL to generate protein clusters.

erroneous orthologous relationships can be formed in the presence of gene loss events. If in a gene loss event the ortholog of a protein is lost, the protein sequence may form an erroneous reciprocal best hit connection with a non-ortholog.

The input to the RBH detector component is the protein sequence similarity graph. It detects the RBHs of a set of proteomes based on the similarity values. The output is the list of RBHs that are labeled in the similarity graph and passed to the syntenic RBH detector component. The syntenic RBH detector component checks whether the mapped genomic regions of the detected RBHs are conserved or not. Genomic regions of RBHs are determined based on gene locations and the range parameter d passed to the component. Genomic region conservations are determined with the range parameters t_N passed to the component. If an RBH of interest is in a syntenic region then the connection is labeled as syntenic RBH. The output of the syntenic RBH detector is the list of syntenic RBHs that are labeled in the similarity graph and passed to the synteny injector component. The similarity graph at this stage has RBH and syntenic RBH connections labeled. The synteny injector component scales the similarity values with the input weight parameters of the RBHs and syntenic RBHs and generates a syntenic-similarity graph. This graph is then used as a seed for the MCL algorithm to generate protein clusters.

4.1.2 Evaluation Methodology

We evaluate the performance of SynAPhy in generating high quality clusters. High quality clusters typically have highly similar members and good presence of RBH and syntenic RBH connections. We use four cluster quality metrics to perform the evaluation: (1) cluster edge density as defined in Definition 2.1, (2) cluster normalized edge weights as defined in Definition 3.4, (3) cluster RBH density as defined in Definition 4.5, and (4) cluster syntenic RBH density as defined in Definition 4.6.

4.1.3 Organization of this Chapter

The remainder of this chapter is organized as follows. Section 4.2 defines the terminologies that we use in this chapter. Section 4.3 presents a summary of the input dataset. Section 4.4 describes the SynAPhy algorithm in details and presents the algorithmic components. Section 4.5 presents the experiments and the results of clusterings on the input dataset. Section 4.6 compares SynAPhy to orthology prediction systems that use synteny to predict orthologs. Section 4.7 concludes the chapter.

4.2 Terminology

This section presents terminologies, definitions, and notations that we use throughout this chapter.

4.2.1 Neighborhood

The following notations are used to define a gene neighborhood. Let

`p` denote a protein;

`p.chromosome` denote the identifier of the chromosome on which the gene of the protein `p` lies; and

`p.head` denote the first position on p.chromosome of the gene of `p` reading from 3 prime to 5 prime end of the chromosome.

Definition 4.1. (**Neighborhood $N(p,d)$ of a protein p of diameter $d = 2 \times k + 1$**) Let p be a protein of the proteome P and let $(p_1, \ldots, p_m)$ be the proteins corresponding to the genes of `p.chromosome` in order of their head. If p is p_i in this ordered list, then

$$N(p,d) \;=\; \{\; p_j \mid (\; i-k \;\leq\; j \;\leq\; i+k \;) \text{ and } (\; 1 \;\leq\; j \;\leq\; m \;)\;\}, \tag{15}$$

where k is the number of upstream and downstream genes. Note that the second clause in the condition accommodates neighborhoods at or near the end of the chromosome. Let `p.neighborhood(d)` denote the neighborhood of a protein p of diameter d.

4.2.2 Protein Sequence Similarity Graph

This section presents definitions and notations used to describe protein sequence similarity graphs.

Definition 4.2. (**Similarity edge weight** $sim(p,q)$ **of proteins** p **and** q) Given the all-versus-all Blast searches of proteomes P and Q, and proteins $p \in P$ and $q \in Q$, the similarity edge weight is then defined as

$$sim(p,q) = \begin{cases} -log(E\text{–}value(p,q)), & \text{if } E\text{–}value(p,q) \neq 0; \\ Max, & \text{if } E\text{–}value(p,q) = 0. \end{cases} \tag{16}$$

where Max is a positive integer strictly larger than all values $-log(E\text{–}value(p_i, q_j))$ for all $p_i \in P$ and for all $q_j \in Q$. Note that $sim(p,q)$ is unidirectional, i.e., it only considers forward hits when a proteome is compared against another. This is different from the $bi\_sim(p,q)$ defined in Definition 3.1 in which forward and backward hits are considered to assign a similarity score for a pair of proteins.

Definition 4.3. (**Reciprocal Best Hit**) Proteins p and q from distinct proteomes P and Q are *reciprocal best hit*, or RBH, if and only if they satisfy all of the following criteria.

1. $sim(p,q) \geq 30$;
2. $sim(q,p) \geq 30$;
3. q is the best hit for p in proteome Q; that is,

$$\forall y \in Q,\ sim(p,y) \leq sim(p,q) \tag{17}$$

   and

4. p is the best hit for q in proteome P; that is,

$$\forall x \in P,\ sim(q,x) \leq sim(q,p) \tag{18}$$

then $rbh(p,q) = \texttt{true}$.

Definition 4.4. (**Syntenic Reciprocal Best Hit**) Proteins p and q from distinct proteomes P and Q are *syntenic reciprocal best hit*, or syntenic RBH, if and only if they satisfy all of the following criteria.

1. p and q are reciprocal best hits; and
2. p and q have syntenic neighborhoods, that is, for a neighborhood of diameter d, the number of reciprocal best hits meets or exceeds the threshold t_N:

$$|\{ <x,y> \mid x \in N(p,d) \ \wedge \ y \in N(q,d) \ \wedge \ rbh(x,y) = \texttt{true} \}| \ \geq \ t_N \tag{19}$$

Given proteins p in proteome P and q in proteome Q. We use the following notations to refer to the features of p and q in a protein sequence similarity graph. Let

`(p, q)` denote an edge connecting proteins p and q;

`(p, q).sim` denote $sim(p,q)$;

`(p, q).maxsim` denote $max($`(p, q).sim`, `(q, p).sim`$)$;

`(p, q).type` denote the type of edge in [`edge`, `hit`, `best_hit`, `rbh`, `syn_rbh`], where

`edge` indicates `(p, q).sim` ≥ 3,

`hit` indicates `(p, q).sim` ≥ 30,

`best_hit` indicates `(p, q).type` $=$ `hit` and $\forall y \in Q$, `(p, y).sim` $\leq$ `(p, q).sim` ,

`rbh` indicates `(p, q).type` $=$ `best_hit` and `(q, p).type` $=$ `best_hit`,

`syn_rbh` denotes that p and q satisfy the criteria for RBH and furthermore the number of RBHs in the neighborhood of diameter $d = 9$ meets or exceeds the threshold $t_N = 4$; and

`(p, q).weight` denote the edge weight. This is defined in terms of the parameters β_e, β_r, β_s as follows

$$\texttt{(p, q).weight} = \begin{cases} \beta_e \times \texttt{(p, q).maxsim}, & \text{if } \texttt{(p, q).type} = \texttt{edge}; \\ \beta_e \times \texttt{(p, q).maxsim}, & \text{if } \texttt{(p, q).type} = \texttt{hit}; \\ \beta_e \times \texttt{(p, q).maxsim}, & \text{if } \texttt{(p, q).type} = \texttt{best\_hit}; \\ \beta_r \times \texttt{(p, q).maxsim}, & \text{if } \texttt{(p, q).type} = \texttt{rbh}; \textit{and} \\ \beta_s \times \texttt{(p, q).maxsim}, & \text{if } \texttt{(p, q).type} = \texttt{syn\_rbh}. \end{cases} \tag{20}$$

The weighted protein sequence similarity graph for the set $(P_1, \ldots, P_k)$ of proteomes is given by $G(P_1, \ldots, P_k)$. We apply k^2 all-versus-all Blast searches with the following Blast call:

>blastp -db <*database_name*> **-query** <*input_file*> **-out** <*output_file*> **-evalue 1e-3 soft_masking true -outfmt 6 -use_sw_tback**

The Blast call is the same one as in Section 3.4.1. The similarity edge weights of the graph $G(P_1, \ldots, P_k)$ is given by the Definition 2.4.

4.3 Dataset

Our dataset is comprised of eight fungal genomes. Table 6 shows their sources as well as abbreviations we use to refer to them throughout this document. The access date for all is 29 September 2013. Table 7 shows summary data for the eight fungal genomes.

Strain name	Source	Version	Abbreviation
Aspergillus niger CBS 513.88	www.aspergillusgenome.org	s01-m06-r02	*A. niger*
Aspergillus oryzae RIB40	www.aspergillusgenome.org	s01-m08-r15	*A. oryzae*
Neurospora crassa OR74A	www.broadinstitute.org	12	*N. crassa*
Aspergillus nidulans FGSC A4	www.aspergillusgenome.org	s10-m01-r10	*A. nidulans*
Aspergillus fumigatus Af293	www.aspergillusgenome.org	s03-m04-r13	*A. fumigatus*
Candida albicans SC5314	www.candidagenome.org	A21-s02-m07-r10	*C. albicans*
Saccharomyces cerevisiae S288	www.yeastgenome.org	R64-1-1_20110203	*S. cerevisiae*
Schizosaccharomyces pombe ASM294	www.pombase.org	v2.20	*S. pombe*

Table 6: Sources of the eight input fungal genomes.

Organism	Assembly	# components	# proteins
A. niger	Chromosome	19	14,060
A. oryzae	Chromosome Contig	8 3	11,902
N. crassa	Supercontig	7	10,785
A. nidulans	Chromosome	8	10,701
A. fumigatus	Chromosome	8	9,783
C. albicans	Chromosome	8	6,217
S. cerevisiae	Chromosome	16	5,862
S. pombe	Chromosome	3	5,144

Table 7: Summary data for the eight input fungal genomes. The abbreviated organism name is given by the column labeled *Organism*. The genome assembly components as given by the source database are given in the column labeled *Assembly*. The number of assembly components is given by the column labeled # components. The number of protein sequences present in the corresponding database is given by the column labeled *# proteins*. The rows are sorted in descending order based on number of proteins.

4.4 The SynAPhy Algorithm

In this section, we describe the details of the SynAPhy algorithm (Algorithm 1). SynAPhy takes in three classes of input for the set $(P_1, \ldots, P_k)$ of proteomes: (1) the sequence similarity graph that represents the initial similarity graph that holds the Blast similarity values, (2) neighborhood range parameters labeled as d and t_N as defined in Definition 4.4, where $d = 9$ represents the diameter of the neighborhood to consider when calculating syntenic RBHs, and $t_N = 4$ represents the number of required RBHs in a neighborhood to determine if a pair of proteins is a syntenic RBH, and (3) a sequence of weight parameters labeled as β_e, β_r, β_s, which are used to compute the edge weights for the edge types described in Section 4.2.2.

A SynAPhy run over the input graph attaches several labels to the edges representing the edge types as well as the edge weights. The resulting graph is used as a seed for the MCL algorithm.

Algorithm 1 The SynAPhy algorithm

```
1: function SYNAPHY(G(P_1, ..., P_k), β_e, β_r, β_s, d, t_N)
2:     for all <P_i, P_j> ∈ pairs(P_1, ..., P_k) do
3:         find_hits(P_i, P_j)
4:         find_rbh(P_i, P_j)
5:         find_syn_rbh(P_i, P_j, d, t_N)
6:         assign_edge_weights(P_i, P_j, β_e, β_r, β_s)
7:     end for
8: end function
```

SynAPhy starts by running a sequence of functions on the input proteomes to determine the edge types. Each function operates on a pair of proteomes at a time and labels parts of the similarity graph with the edge types based on the available similarity data.

The first step, *find_hits* (Algorithm 2) determines if edges connecting proteins are in the list [`edge`, `hit`, `best_hit`]. This function takes two proteomes P and Q as input. Based on the similarity values, this function first determines if the edge type is either an `edge` or a `hit` (Algorithm 2, line 4).

Algorithm 2 Determines edge types in [`edge`, `hit`, `best_hit`] for each protein pair in a pair of proteomes P and Q

```
1: function FIND_HITS(P, Q)
2:     for all <p, q> ∈ <P, Q> do
3:         sim ⇐ (p, q).maxsim
4:         if sim ≤ 30 then
5:             (p, q).type ⇐ edge
6:             (q, p).type ⇐ edge
7:         else
8:             (p, q).type ⇐ hit
9:             (q, p).type ⇐ hit
10:        end if
11:    end for
12:    for all p ∈ P do
13:        max_sim ⇐ 0
14:        max_hit ⇐ nil
15:        for all q ∈ Q do
16:            if (p, q).sim ≥ max_sim then
17:                max_sim ⇐ (p, q).sim
18:                max_hit ⇐ q
19:            end if
20:        end for
21:        (p, max_hit).type ⇐ best_hit
22:    end for
23:    for all q ∈ Q do
24:        max_sim ⇐ 0
25:        max_hit ⇐ nil
26:        for all p ∈ P do
27:            if (q, p).sim ≥ max_sim then
28:                max_sim ⇐ (q, p).sim
29:                max_hit ⇐ p
30:            end if
31:        end for
32:        (q, max_hit).type ⇐ best_hit
33:    end for
34: end function
```

Algorithm 3 Determines if the best hits in a pair of proteomes are RBHs

```
1: function FIND_RBH(P, Q)
2:     for all <p, q> ∈ <P, Q> do
3:         forward_hit ⇐ (p, q).type ≡ best_hit
4:         backward_hit ⇐ (q, p).type ≡ best_hit
5:         if forward_hit ∧ backward_hit then
6:             (p, q).type ⇐ rbh
7:             (q, p).type ⇐ rbh
8:         end if
9:     end for
10: end function
```

Algorithm 4 Determines if the RBHs in a pair of proteomes are syntenic RBHs

```
1: function FIND_SYN_RBH(P, Q, d, t_N)
2:     for all <p, q> ∈ <P, Q> do
3:         N_p ⇐ p.neighborhood(d)
4:         N_q ⇐ q.neighborhood(d)
5:         rbh_size ⇐ 0
6:         for all <n_p, n_q> ∈ <N_p, N_q> do
7:             if (n_p, n_q).type ≡ rbh then
8:                 rbh_size ⇐ rbh_size + 1
9:             end if
10:        end for
11:        if rbh_size ≥ t_N then
12:            (p, q).type ⇐ syn_rbh
13:            (q, p).type ⇐ syn_rbh
14:        end if
15:    end for
16: end function
```

Once all edges are labeled, the function runs another pass on the proteins to determine the best hits for every protein in the given proteomes (Algorithm 2, lines 12 and 23). Note that when calculating the best hits we find both, the best hits for every protein from proteome P in proteome Q as well as the best hits for every protein from proteome Q in proteome P. Edges that correspond to best hits are then relabeled with the edge type `best_hit`.

The second step, *find_rbh* (Algorithm 3) determines the RBHs in the graph as defined in Definition 4.3. Similar to *find_hits*, this function takes two proteomes P and Q as input. Using the edge type data from the first step, this method visits each protein pair $<p, q>$ in the input proteomes and labels the edge type as `rbh` if both `(p, q).type` is `best_hit` and `(q, p).type` is `best_hit` (Algorithm 3 line 5). The resulting graph with all edge labels combining with the ones in *find_hits* is referred to as G_{sim} throughout the chapter.

Now that RBH edges are calculated, the last labeling step is to determine whether the RBH edges are syntenic or not. This is performed in *find_syn_rbh* (Algorithm 4). This function takes in two proteomes, as well as the range parameters d and t_N. *find_syn_rbh* performs the following on each protein pair $<p, q>$: (1) finds the neighborhoods $N(p, d)$ and $N(q, d)$ as defined in Definition 4.1, (2) determines the number of RBHs between the proteins in the two neighborhoods (Algorithm 4 line 5), and (3) labels `(p, q).type` as `syn_rbh` if the number of neighboring RBHs meets or exceeds t_N (Algorithm 4 line 11).

After analyzing the edge types, SynAPhy proceeds with assigning weights to the graph edges. This is performed in *assign_edge_weights* (Algorithm 5), where the weights are calculated as a function of the similarity values, the edge types as calculated in the previous steps, and the input weight parameters β_e, β_r, and β_s. The resulting graph is referred to as G_{syn_sim} throughout the chapter.

The edge type and weight assignment process is repeated for every pair of proteomes in the input graph G. SynAPhy terminates when all the edges of the input graph are assigned a weight value.

Algorithm 5 Assigns weights to the edges based on the edge types

```
1: function ASSIGN_EDGE_WEIGHTS(P, Q, β_e, β_r, β_s)
2:     for all <p, q> ∈ <P, Q> do
3:         if (p, q).type ≡ syn_rbh then
4:             (p, q).weight ⇐ (p, q).maxsim × β_s
5:         else if (p, q).type ≡ rbh then
6:             (p, q).weight ⇐ (p, q).maxsim × β_r
7:         else
8:             (p, q).weight ⇐ (p, q).maxsim × β_e
9:         end if
10:    end for
11: end function
```

4.5 Experiments

We apply MCL, SynAPhy, OrthoMCL, and OMA on the eight fungal genomes listed in Section 4.3. We start our experiments by performing a detailed analysis of the input proteomes. The analysis presents the differences of the best hits, RBHs, and syntenic RBHs in terms of their sequence alignment similarity, hit coverage, and percent identity. This analysis helps us determine whether thresholds for E-value, query and subject coverage percentages, and percent identity need to be set for RBHs and syntenic RBHs.

We evaluate the output clusters generated by the MCL, SynAPhy, OrthoMCL, and OMA algorithms using the generic cluster quality metrics: cluster edge density, cluster normalized edge weights, cluster RBH edge density, and cluster syntenic RBH edge density.

We present two example output clusters generated by the MCL algorithm that have member sequences with different functions as annotated by the mycoCLAP database. Both examples show how in the presence of paralogs and absence of syntenic RBHs orthologous relationships are not distinguished. In particular, the first example shows how the presence of both paralogs and many RBHs does not allow SynAPhy to separate cluster members with different functions into different clusters. In addition, it shows how the absence of syntenic RBHs does not allow SynAPhy to separate members into orthologous clusters in which every pair is orthologous to each other. The second example showcases the usefulness of RBHs with which SynAPhy was able to separate sequences with different functions into different clusters.

4.5.1 Similarity of Proteomes

In this section, we study the variation of edge type frequencies with respect to alignment data. The purpose of this experiment is to investigate if any alignment thresholds are needed to be set to filter out "poor" hits for SynAPhy. We consider three edge types and three alignment metrics. The edge types are: best hits, RBHs, and syntenic RBHs, and the alignment metrics are: E-value, query and subject coverage percentages, and percent identity. The three edge types as presented in Section 4.2.2 are connections between proteomes. Therefore, the experiments in this section are performed between proteome and not within proteome connections.

We first illustrate the frequencies for best hits, RBHs, and syntenic RBHs at different similarity values. Figure 14 depicts the kernel density estimates showing the distribution of similarity values by sequence relationship type. The x-axis shows similarity value, and the y-axis shows kernel density estimate. Best hits are depicted with a dotted line, RBHs are depicted with a dashed line, and syntenic RBHs are depicted with a solid line.

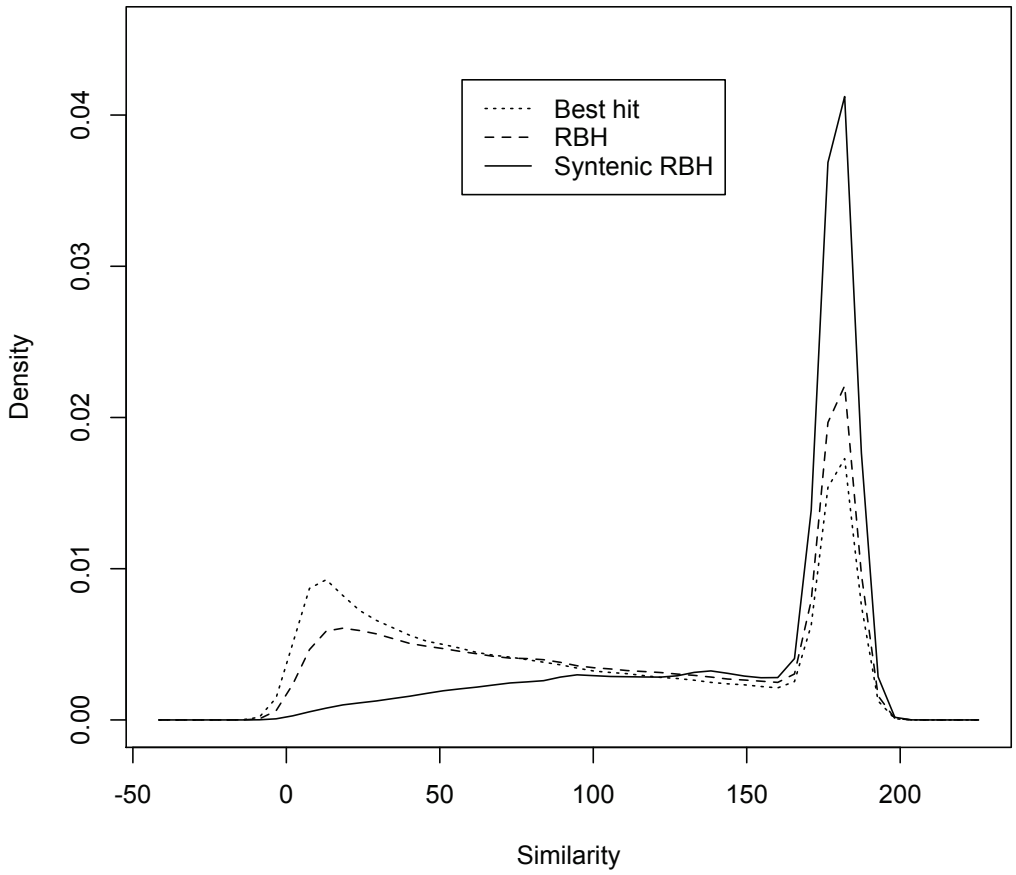

Figure 14: Kernel density estimates showing the distribution of similarity values by sequence relationship type

For best hits, we observe a bump for similarity values within the range 3 to 30. This bump decreases with RBHs and is not present with syntenic RBHs. Best hits within the range 3 to 30 may be "poor" hits that are filtered out with RBHs, and in succession "poor" RBHs that are filtered out with syntenic RBHs. For all three sequence relationship types, we observe peaks for similarity values across the range 175 to 181. These hits are good hits and are expected to cover a high fraction of query and subject sequence lengths.

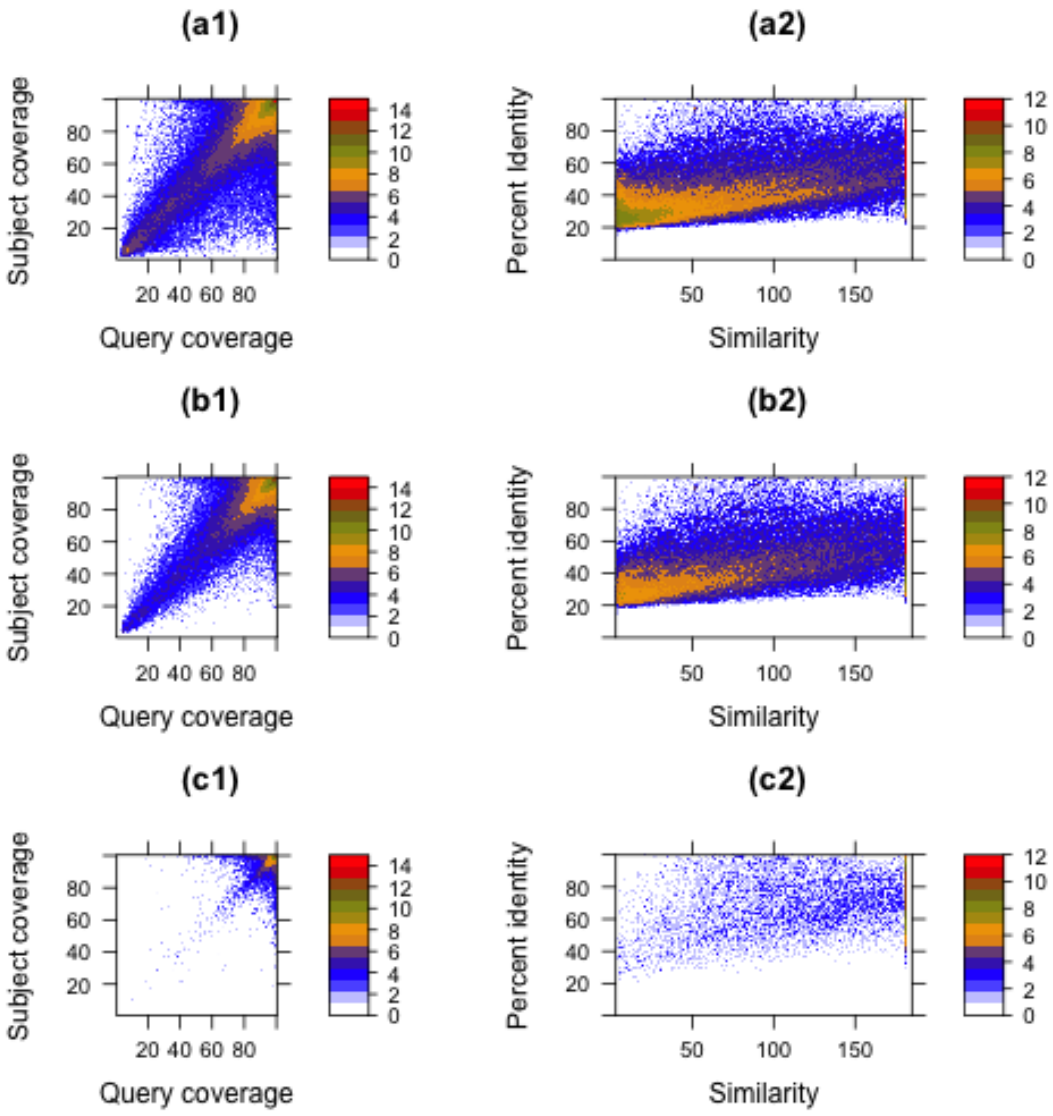

Figure 15: Blast search hit frequency by sequence relationship type. Panels **(a1)**, **(b1)**, and **(c1)** show hit frequencies at query and subject coverage percentages of best hits, RBHs, and syntenic RBHs, respectively. Panels **(a2)**, **(b2)**, and **(c2)** show hit frequencies for similarity and percent identity pair value of best hits, RBHs, and syntenic RBHs, respectively. For all panels, frequencies are in binary logarithm displayed as color coded ranges shown in the color legends.

We next demonstrate that most of the hits that are filtered out with RBHs and syntenic RBHs are short hits. Figure 15 shows hit frequencies by sequence relationship type. Panels (a1), (b1), and (c1) show frequencies of best hits, RBHs, and syntenic RBHs, respectively, on plots of the percentage coverage of query sequences against subject sequences. Light blue indicates a very low frequency and red indicates a very high frequency. We observe from the three panels that the majority of best hits, RBHs, and syntenic RBHs have almost 100% query and subject coverages and sit in the upper-right corners of the plots with high query and subject coverage percentages. Moreover, RBHs shown in panel (b1) filter out a large number of hits with 20% query and subject coverages. This is shown in the lower-left corner of panel (b1) when compared to panel (a1). Syntenic RBHs shown in panel (c1) in succession filter out most of the hits below 80% query and subject coverage compared to RBHs shown in panel (b1). This indicates that RBHs and the majority of syntenic RBHs have good fraction of their sequences aligned.

We next demonstrate that most of the hits that are filtered out with RBHs and syntenic RBHs have low percent identities. Panels (a2), (b2), and (c2) in Figure 15 show frequencies of best hits, RBHs, and syntenic RBHs, respectively, on plots of their similarity value against percent identity. Light blue indicates a very low frequency and red indicates a very high frequency. We observe from the three panels that the majority of best hits, RBHs, and syntenic RBHs fall in a percent identity range from 60 to 80. These are the hits that are not filtered out with RBHs and syntenic RBHs. Moreover, we observe that most of the hits that are filtered out with RBHs in panel (b2) when compared to best hits in panel (a2) are hits falling in a percent identity range from 20 to 40 with similarity value range 3 to 30. More hits are filtered out with higher similarity values. Syntenic RBHs shown in panel (c2) in succession filter out the majority of hits in a percent identity range from 20 to 40.

The results of the experiments show that setting a low E-value threshold for similarity values eliminates the need to incorporate query and subject coverage percentages and percent identity in similarity edge weights for best hits, RBHs, and syntenic RBHs.

4.5.2 Markov Clustering with and without Synteny

4.5.2.1 General Results

Recall from Section 4.4 that the similarity graph with edge type labels in [`edge`, `hit`, `best_hit`, `rbh`, `syn_rbh`] is denoted by G_{sim} and the weighted synteny-similarity graph is denoted by G_{syn_sim}. In order to evaluate the impact of scaling RBH and syntenic RBH similarity weights, we apply the MCL algorithm on both G_{sim} and G_{syn_sim}.

We report four cluster quality metrics for the MCL clusters that help us highlight the differences of G_{sim} and G_{syn_sim}: (1) cluster edge density as defined in Definition 2.1, (2) cluster normalized edge weights as defined in Definition 3.4, (3) cluster RBH density as defined in Definition 4.5, and (4) cluster syntenic RBH density as defined in Definition 4.6.

Definition 4.5. (**Cluster RBH density**) Given a cluster with the tuple $(|P_1|, \ldots, |P_k|)$ of protein cardinality, where $|P_i|$ is the number of sequences in the cluster from proteome P_i. The maximum

possible number of edges connecting the proteins of the k proteomes is given by

$$ParMax = \sum_{\forall i,j \in k \ | \ i<j} |P_i| \times |P_j| \tag{21}$$

Note that the cluster is considered a k-partite graph (see Section 2.2), where proteins from a specific proteome are contained in a partition. Given n RBH edges between proteins of different proteomes, the RBH density of the cluster is

$$\delta_{rbh} = \frac{n}{ParMax} \tag{22}$$

Definition 4.6. (**Cluster syntenic RBH density**) Given a protein cluster, m syntenic RBH edges between proteins of different proteomes, and the maximum possible number of edges between the proteins of the different proteomes in the cluster $ParMax$ (given by Equation 21), the syntenic RBH density is

$$\delta_{syn\_rbh} = \frac{m}{ParMax} \tag{23}$$

Table 8 shows the results of the MCL runs at different inflation values in relation to the four metrics. The runs are given by the column labeled i. The inflation values are given by the column labeled I. The *split/joint* distances as defined in Definition 2.6 given by pair-value distances as defined in Definition 2.7 between contiguous clusterings are given by the column labeled $d(C_i, C_{i+1})$. The RBH edge weight scalar is given by the column labeled β_r, and the syntenic RBH edge weight scalar is given by the column labeled β_s. The minimum, maximum, mean, and standard deviation of cluster edge densities are given by the columns labeled δ_{min}, δ_{max}, δ_{μ}, and δ_{σ}, respectively. The minimum, maximum, mean, and standard deviation of cluster normalized edge weight $\overline{w}$ are given by the columns labeled $\overline{w}_{min}$, $\overline{w}_{max}$, $\overline{w}_{\mu}$, and $\overline{w}_{\sigma}$, respectively. The mean and standard deviation of RBH densities are given by the columns labeled $\delta_{rbh_{\mu}}$ and $\delta_{rbh_{\sigma}}$, respectively. The mean and standard deviation of syntenic RBH densities are given by the columns labeled $\delta_{syn\_rbh_{\mu}}$ and $\delta_{syn\_rbh_{\sigma}}$, respectively.

We run MCL on three different graphs: (1) G_{sim} indicated in Table 8 with both β_r and β_s equal to 1, we refer to these runs as MCL; (2) G_{syn_sim} with $\beta_r = 1$ and $\beta_s = 5$, we refer to these runs as SynAPhy_1_5; and (3) G_{syn_sim} with $\beta_r = 3$ and $\beta_s = 5$, we refer to these runs as SynAPhy_3_5. We first discuss the *split/joint* pair-value distances. For each graph, we observe a consistency in the *split/joint* pair-value distances of contiguous clusterings up until inflation value 15 in that clusterings with higher inflation values are sub-clusterings of the ones with lower inflation values. This behavior is apparent in the pair-value distances. Consider the entry $d(C_i, C_{i+1})$ of run 1. The entry has the *split/joint* pair-value distance of the clusterings at inflation values 1.4 and 3.2. The first value of the tuple is 13,705, which is the number of rearrangements that have to be applied on the clustering at inflation value 1.4 to obtain a sub-clustering of 3.2, and the second value of the tuple is 374, which is the number of rearrangements that have to be applied on the clustering at inflation value 3.2 to obtain a sub-clustering of 1.4. This trend is interrupted at run 5 where the clustering at inflation value 20 is no longer a sub-clustering of the one at inflation value 15. This is shown with the pair-value (1026, 2309) in the $d(C_i, C_{i+1})$ entry of run 5. We observe the same trend in SynAPhy_1_5 and SynAPhy_3_5. Clusterings at inflation value 20 are not the sub-clusterings of the ones at inflation value 15 as shown in runs 13 and 21.

i	I	$d(C_i, C_{i+1})$	β_r	β_s	δ_{min}	δ_{max}	δ_μ	δ_σ	$\overline{w}_{min}$	$\overline{w}_{max}$	$\overline{w}_\mu$	$\overline{w}_\sigma$	δ_{rbh_μ}	δ_{rbh_σ}	$\delta_{syn\_rbh_\mu}$	$\delta_{syn\_rbh_\sigma}$
1	1.4	(13705, 374)	1	1	0.2222	1.0	0.935	0.1277	0.0134	1.0	0.3979	0.2716	0.6715	0.3566	0.1591	0.2114
2	3.2	(4364, 383)	1	1	0.25	1.0	0.9579	0.0994	0.0083	1.0	0.4518	0.2848	0.6924	0.355	0.1585	0.2171
3	6.0	(1616, 279)	1	1	0.25	1.0	0.9621	0.0944	0.0083	1.0	0.4698	0.2896	0.6923	0.3578	0.158	0.2215
4	8.0	(2336, 387)	1	1	0.25	1.0	0.9635	0.0926	0.0083	1.0	0.4746	0.2912	0.6928	0.3591	0.1576	0.2222
5	15.0	(1026, 2309)	1	1	0.2857	1.0	0.9656	0.0896	0.01	1.0	0.4833	0.2938	0.6915	0.3617	0.1577	0.2252
6	20.0	(1019, 5768)	1	1	0.1737	1.0	0.9663	0.089	0.01	1.0	0.4856	0.2951	0.6933	0.3623	0.1583	0.2272
7	25.0	(750, 4900)	1	1	0.0356	1.0	0.9666	0.0892	0.0055	1.0	0.4842	0.2973	0.7006	0.3613	0.1624	0.231
8	30.0	-	1	1	0.0185	1.0	0.9671	0.0892	0.0027	1.0	0.4792	0.2985	0.709	0.3595	0.1653	0.2328
9	1.4	(13525, 582)	1	5	0.2308	1.0	0.941	0.1215	0.0134	1.0	0.4173	0.2729	0.6895	0.3487	0.1699	0.2141
10	3.2	(2820, 801)	1	5	0.2857	1.0	0.9607	0.0951	0.0107	1.0	0.4774	0.295	0.7063	0.3456	0.1857	0.2458
11	6.0	(851, 376)	1	5	0.25	1.0	0.9634	0.0917	0.0083	1.0	0.4859	0.2964	0.7046	0.3482	0.1808	0.2453
12	8.0	(1231, 347)	1	5	0.25	1.0	0.9643	0.0902	0.0083	1.0	0.4873	0.2967	0.7031	0.3496	0.1785	0.2438
13	15.0	(925, 1578)	1	5	0.25	1.0	0.9657	0.0883	0.0083	1.0	0.4895	0.297	0.701	0.3522	0.1764	0.2441
14	20.0	(1587, 4324)	1	5	0.1808	1.0	0.9663	0.0878	0.0083	1.0	0.4932	0.2998	0.7043	0.3525	0.1828	0.2535
15	25.0	(1464, 3939)	1	5	0.0472	1.0	0.9667	0.088	0.0078	1.0	0.4921	0.3008	0.7123	0.35	0.1917	0.2616
16	30.0	-	1	5	0.0247	1.0	0.9673	0.0877	0.0039	1.0	0.4924	0.3035	0.7237	0.3474	0.2041	0.2749
17	1.4	(15247, 393)	3	5	0.2418	1.0	0.9469	0.1137	0.012	1.0	0.4243	0.2744	0.7028	0.3401	0.1631	0.2102
18	3.2	(3100, 584)	3	5	0.25	1.0	0.9649	0.0913	0.0107	1.0	0.4968	0.2977	0.743	0.3212	0.1574	0.2186
19	6.0	(858, 274)	3	5	0.25	1.0	0.9671	0.0886	0.011	1.0	0.5108	0.3028	0.7495	0.3202	0.1532	0.2209
20	8.0	(1330, 283)	3	5	0.25	1.0	0.9678	0.0876	0.011	1.0	0.5148	0.3044	0.7519	0.3202	0.1519	0.2217
21	15.0	(804, 844)	3	5	0.25	1.0	0.9689	0.0863	0.011	1.0	0.5204	0.3066	0.7541	0.3216	0.1508	0.2236
22	20.0	(1115, 3429)	3	5	0.2291	1.0	0.9692	0.0859	0.014	1.0	0.5217	0.3079	0.758	0.3213	0.1529	0.2269
23	25.0	(1236, 3412)	3	5	0.0699	1.0	0.9695	0.0859	0.013	1.0	0.5169	0.3077	0.7651	0.3197	0.1583	0.2317
24	30.0	-	3	5	0.0341	1.0	0.9696	0.0862	0.0058	1.0	0.5114	0.3081	0.773	0.3177	0.1667	0.2412

Table 8: Cluster quality of MCL runs at different inflation values on different similarity graphs of the eight input fungal genomes. The runs are given by the column labeled i. The inflation values are given by the column labeled I. The *split/joint* pair-value distances between contiguous clusterings are given by the column labeled $d(C_i, C_{i+1})$. RBH and syntenic RBH edge weight scalars are given by the columns labeled β_r and β_s, respectively. The columns prefixed with δ show cluster edge density data. The columns prefixed with $\overline{w}$ show the average of the normalized edge weights. The columns prefixed with δ_{rbh} and δ_{syn_rbh} show RBH and syntenic RBH edge density data, respectively.

We next discuss the results of the four metrics. The cluster edge density minimums of the three graphs drop starting at inflation value 20 indicated by the entries δ_{min} of runs 6, 14, and 22. This is because the clusterings at inflation value 20 generate larger clusters than the ones at inflation value 15. The cluster edge density maximums of all the clusterings indicated by the column labeled δ_{max} are 1, which indicates the presence of complete clusters. The cluster edge density means of all the clusterings indicated by the column labeled δ_{μ} are above 0.9, which indicates the cluster edge density data for all the clusterings are satisfactory.

We next discuss the average of the normalized edge weights $\overline{w}$. The minimums of all the clusterings are low. This indicates the presence of clusters with members that have low sequence similarity. The maximums of all the clusterings are 1, which indicates the presence of perfect clusters with maximum possible sequence similarity between the members. The means of each graph increase with higher inflation values. Moreover, the means of the SynAPhy_1_5 at every inflation value are higher than the ones in MCL, and in turn the means of the SynAPhy_3_5 at every inflation value are higher than the ones in SynAPhy_1_5. This indicates sequence similarities in SynAPhy_3_5 clusters are higher than in SynAPhy_1_5 and MCL.

We next discuss RBH edge densities and syntenic RBH edge densities. For all the clusterings, $\delta_{rbh_{\min}}$ and $\delta_{syn\_rbh_{\min}}$ are equal to 0, and $\delta_{rbh_{\max}}$ and $\delta_{syn\_rbh_{\max}}$ are equal to 1. This indicates that there are clusters with no RBH or syntenic RBH edge, and there are clusters with every edge typed as RBH or syntenic RBH edge. The RBH density means indicated by the column labeled $\delta_{rbh_{\mu}}$ of SynAPhy_1_5 at every inflation value are higher than the ones in MCL, and in turn the means of SynAPhy_3_5 at every inflation value are higher than the ones in SynAPhy_1_5. This indicates SynAPhy_3_5 has the most number of clusters with RBH edges. The syntenic RBH edge density means indicated by the column labeled $\delta_{syn\_rbh_{\mu}}$ of SynAPhy_1_5 at every inflation value are higher than the ones in MCL. However, this behavior is not the same for SynAPhy_3_5 compared to SynAPhy_1_5. This is because SynAPhy_3_5 clusters RBHs with syntenic RBHs, which is indicated in high $\delta_{rbh_{\mu}}$.

In summary, SynAPhy_3_5 generates clusters with higher sequence similarity among the members than SynAPhy_1_5 and MCL, as well as clusters with more RBH connections. It does not, however, generate clusters with more syntenic RBH connections than SynAPhy_1_5. For the rest of this chapter, we only consider the clusterings of MCL, SynAPhy_1_5, and SynAPhy_3_5 at inflation value 15. This is because up until inflation value 15, consecutive clusterings are consistent in relation to *split/joint* distances. We refer to the MCL clustering at inflation value 15 as MCL_I15, and we keep the references for SynAPhy_1_5 and SynAPhy_3_5.

4.5.2.2 Example Clusters

We illustrate example output clusters generated by the MCL_15, SynAPhy_1_5, and SynAPhy_3_5 clusterings. Our example clusters are clusters with at least one sequence present in the mycoCLAP database. The eight fungal genomes have a total of 165 sequences present in the mycoCLAP database. MCL_I15, SynAPhy_1_5, and SynAPhy_3_5 cluster 104 out of the 165. We present two example output clusters that have member sequences with different functions as annotated by the

mycoCLAP database. The first example shows a cluster with paralogs and many RBH connections with not enough syntenic RBH connections. In this example, MCL_15, SynAPhy_1_5, and SynAPhy_3_5 generate the same cluster.

Figure 16 and Figure 17 illustrate the first example output cluster. Nodes in the figures represent proteins and edges represent their relationship type. There are three types of edges; `hit`, `rbh`, and `syn_rbh`. Recall from Section 4.2.2 that a `hit` is an edge with similarity value greater or equal to 30. The two figures are different snapshots of the same cluster, where Figure 16 has all the three types of edges depicted, and Figure 17 has only `rbh` and `syn_rbh` edge types depicted. We show them in different figures for clarity.

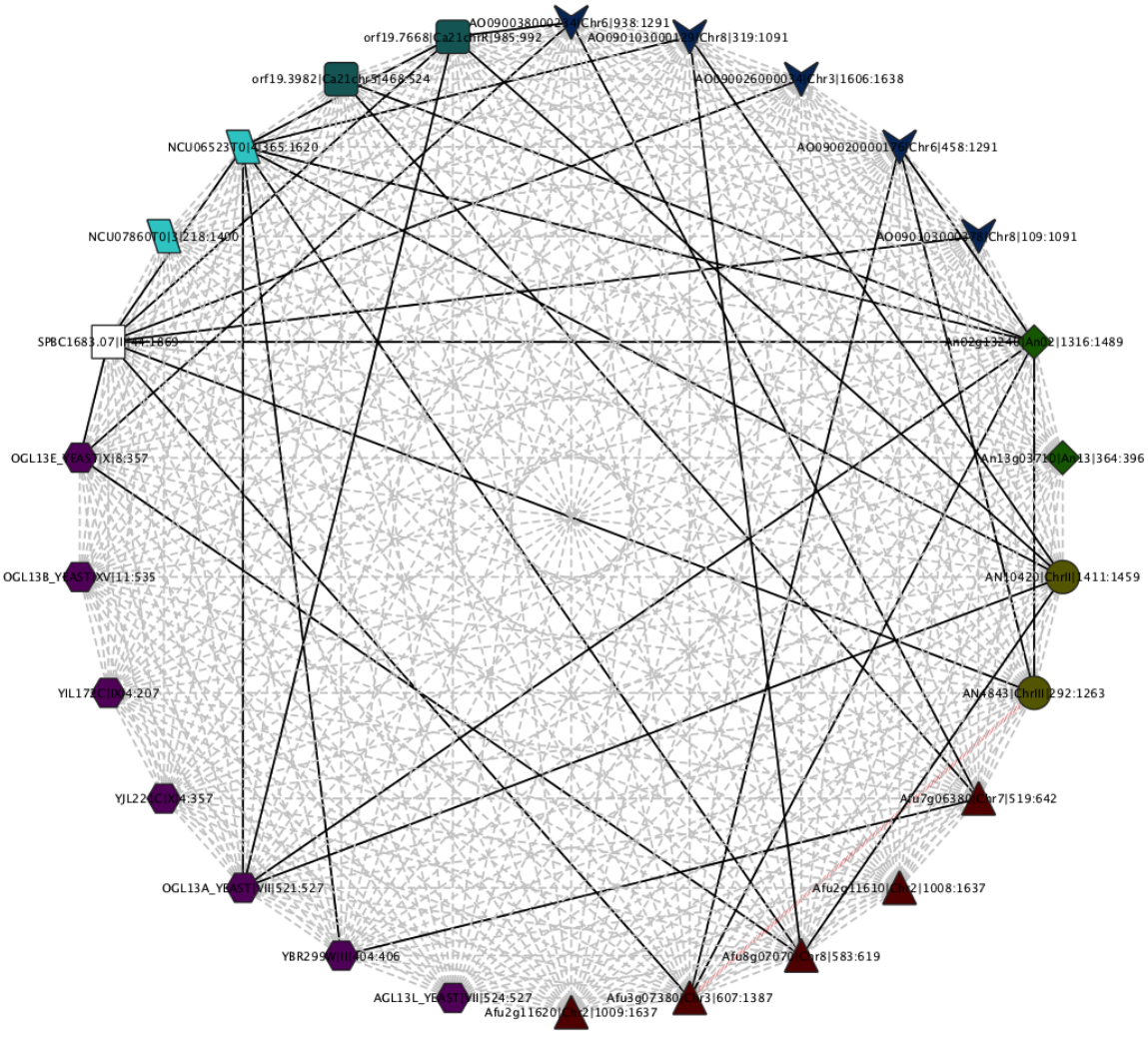

Figure 16: A cluster with GH family 13 sequences from the mycoCLAP database with `hit`, `rbh`, and `syn_rbh` sequence relationship types. Nodes in the figure represent proteins and edges represent their relationship type. There are three types of edges; `hit` edges, `rbh` edges, and `syn_rbh` edges. `hit` edges are depicted by grey dashed lines, `rbh` edges are depicted by solid black lines, and `syn_rbh` edges are depicted by red slashed lines. A node label of a protein `p` is a pipe-delimited string of the source database identifier, `p.chromosome`, and the location of the `p.head` on the gene ordered list of `p.chromosome` joined by the total number of genes on `p.chromosome`. Species are shape coded. Sequences from *A. fumigatus* are depicted as triangles, *A. niger* as diamonds, *A. nidulans* as ellipses, *A. oryzae* as arrow heads, *C. albicans* as round rectangles, *S. pombe* as rectangles, *S. cerevisiae* as hexagon, and *N. crassa* as parallelograms.

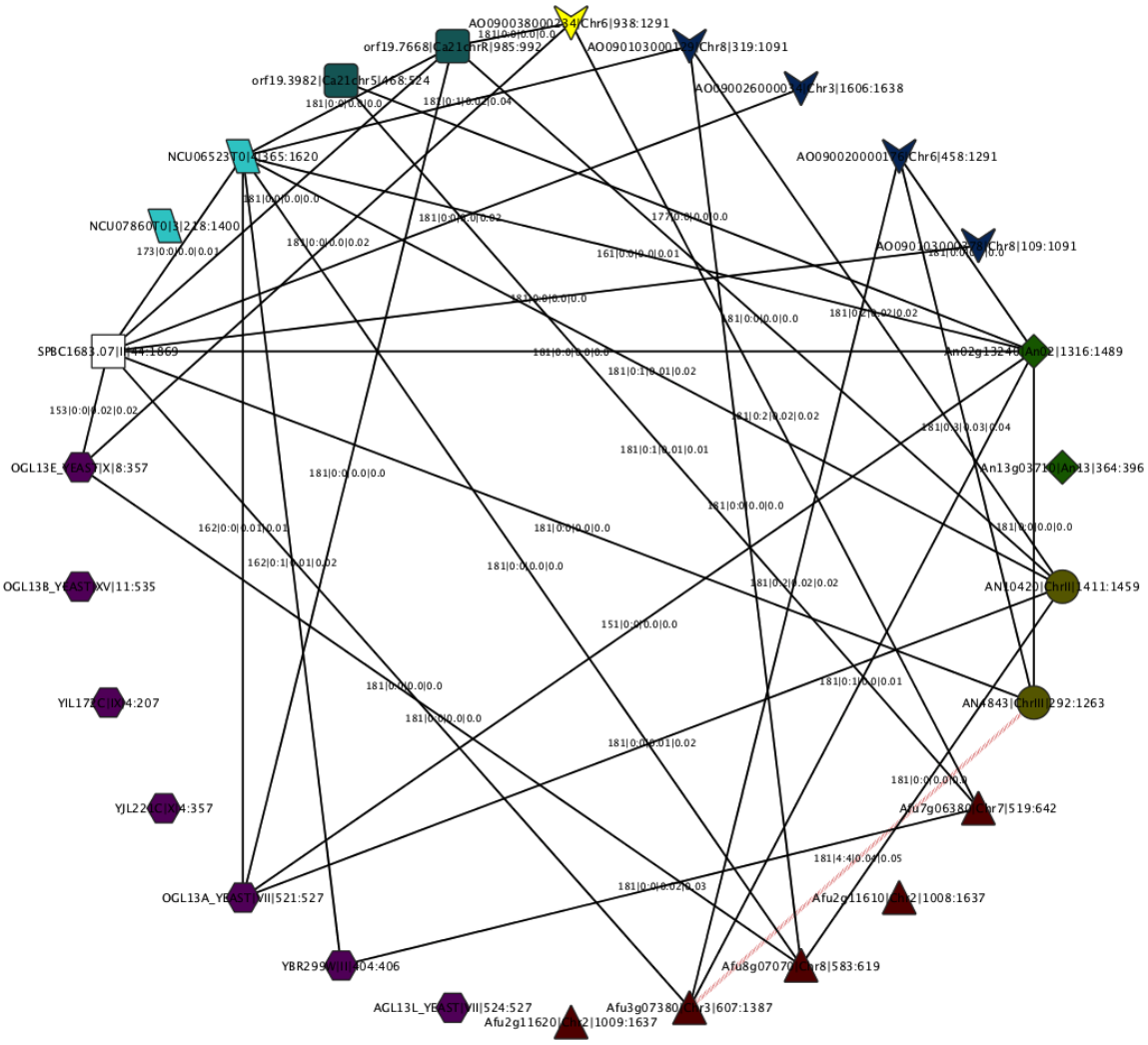

Figure 17: A cluster with GH family 13 sequences from the mycoCLAP database with RBH and syntenic RBH relationship types. The same cluster as in Figure 16 but with only `rbh` and `syn_rbh` edges and added edge labels. An edge is labeled with a pipe-delimited string $x|i:j|y|z$. x corresponds to their similarity value. i and j correspond to the number of RBHs and syntenic RBHs in their neighborhood, respectively. y corresponds to the neighborhood edge density, and z corresponds to the neighborhood normalized edge weights.

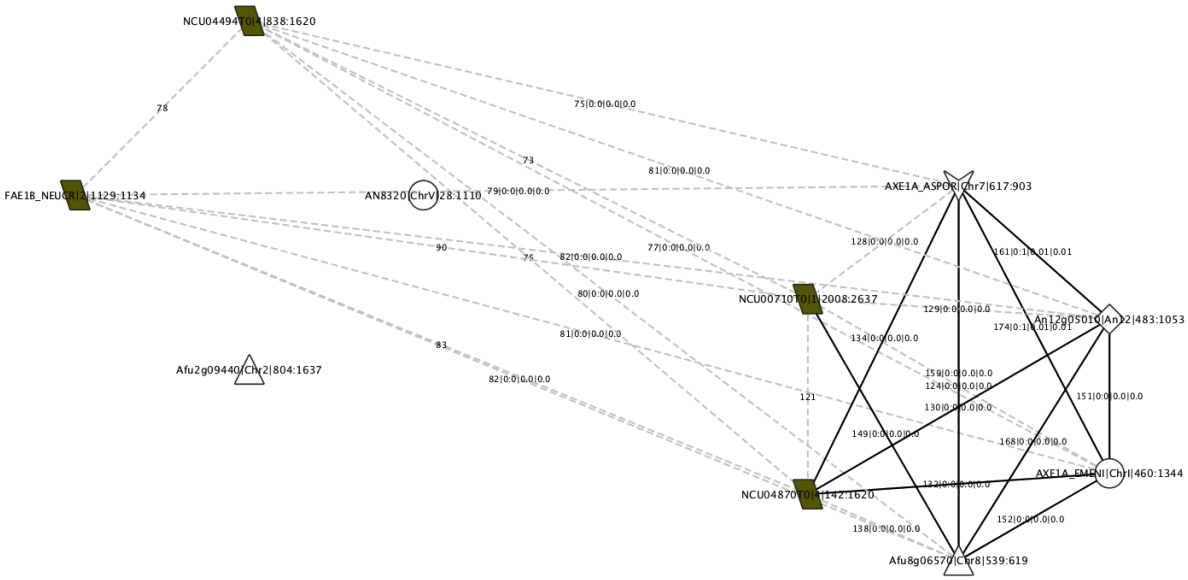

Figure 18: MCL_I15 cluster with CE family 1 sequences from the mycoCLAP database. Nodes in the figures represent proteins and edges represent their relationship type. There are two types of edges; `hit` edges and `rbh` edges. `hit` edges are depicted by grey dashed lines and `rbh` edges are depicted by solid black lines. A node label of a protein `p` is a pipe-delimited string of the source database identifier, `p.chromosome`, and the location of the `p.head` on the gene ordered list of `p.chromosome` joined by the total number of genes on `p.chromosome`. Species are shape coded. Sequences from *A. fumigatus* are depicted as triangles, *A. niger* as diamonds, *A. nidulans* as ellipses, *A. oryzae* as arrow heads, and *N. crassa* as parallelograms. An edge is labeled with a pipe-delimited string $x|i:j|y|z$. x corresponds to their similarity value. i and j correspond to the number of RBHs and syntenic RBHs in their neighborhood, respectively. y corresponds to the neighborhood edge density, and z corresponds to the neighborhood normalized edge weights.

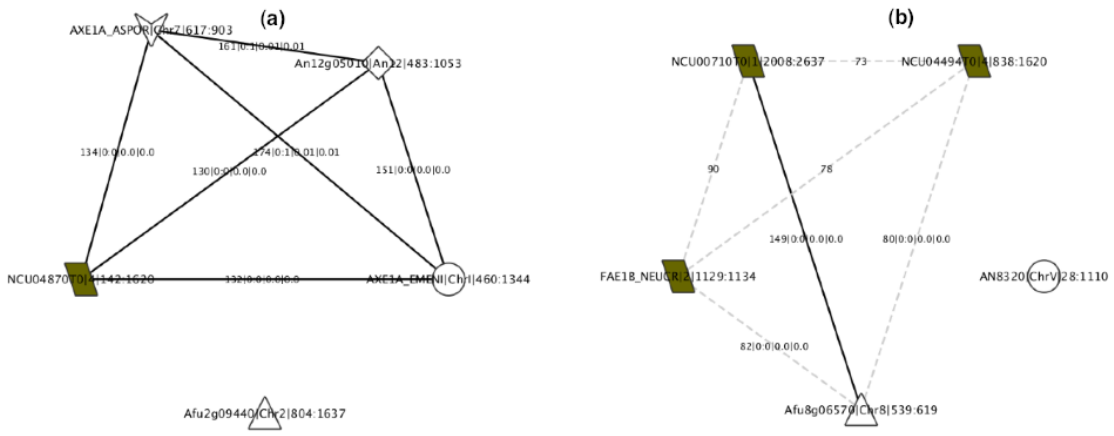

Figure 19: SynAPhy_3_5 clusters with CE family 1 sequences from the mycoCLAP database. SynAPhy_3_5 breaks down the cluster in Figure 18 into two separate clusters shown in panels (a) and (b).

Consider Figure 16. There is one `syn_rbh` edge, and 35 `rbh` edges. The graph is a clique when only `hit` edges are considered, and a connected component when only RBH edges are considered. The cluster has four Glycoside Hydrolase (GH) protein family 13 sequences present in the mycoCLAP database, and they are: AGL13L_YEAST, OGL13A_YEAST, OGL13B_YEAST, and OGL13E_YEAST. They are all from the *S. cerevisiae* genome. AGL13L_YEAST has alpha-glucosidase as a functional annotation, and the other three have oligo-1,6-glucosidase. The cluster has a total of seven sequences from *S. cerevisiae* genome. When these seven *S. cerevisiae* sequences are put together as a graph they have a normalized edge weight value 1, i.e., every pair in the component has the maximum possible similarity value 181. Therefore, they are clearly paralogs. Note that AGL13L_YEAST is an example of a functionally diverged paralog from OGL13A_YEAST, OGL13B_YEAST, and OGL13E_YEAST. This is an example cluster in which many RBH connections did not allow SynAPhy to separate sequences with different functions into different clusters. In addition, with the absence of syntenic RBHs, we can not draw conclusions as to which RBHs are the orthologs.

The presence of many RBHs and the absence of syntenic RBHs motivated us to examine RBH neighborhood conservations to ensure that RBHs are not syntenic RBHs when we increase the neighborhood diameter. Figure 17 only depicts `rbh` and `syn_rbh` edges along with edge labels. Except for the single `syn_rbh`, there is no `rbh` edge that has conserved neighborhood. This is indicated in the count of RBHs in their neighborhood. In fact, all the neighborhood RBH counts of the RBHs are 0. This indicates that even if we increase the neighborhood diameter, RBHs in this cluster will not become syntenic RBHs.

We next describe our second example output cluster illustrated in Figure 18. The cluster has three Carbohydrate Esterase (CE) family 1 sequences from the mycoCLAP database, and they are: AXE1A_EMENI, AXE1A_ASPOR, and FAE1B_NEUCR. AXE1A_EMENI and AXE1A_ASPOR

have acetylxylan esterase as a functional annotation, and are from the *A. nidulans* and the *A. oryzae* genomes, respectively. FAE1B_NEUCR has feruloyl esterase as a functional annotation, and is from the *N. crassa* genome. AXE1A_EMENI and AXE1A_ASPOR are RBHs. FAE1B_NEUCR has no RBH in the cluster. There are two sequences from *A. fumigatus*, two from *A. nidulans*, one from *A. niger*, one from *A. oryzae*, and four from *N. crassa*. This is an example cluster in which the RBH connections allowed SynAPhy to separate sequences with different functions into separate clusters.

In this example of a CE family 1 cluster, MCL_I15 and SynAPhy_1_5 have the same cluster with the same sequence composition. SynAPhy_3_5, however, breaks apart the cluster into two separate clusters shown in panels (a) and (b) in Figure 19. The cluster in panel (a) has the two acetylxylan esterases (AXE1A_EMENI and AXE1A_ASPOR), and the cluster in panel (b) has the feruloyl esterase (FAE1B_NEUCR). While the multi-functional cluster is separated, the *A. fumigatus* sequence, Afu8g06570, that forms the most number of RBH edges with sequences in panel (a) in the MCL_I15 clustering, is in a separate cluster in panel (b). Due to the absence of syntenic RBHs in the cluster, we can not resolve which of the *N. crassa* sequences is the ortholog, and if the *A. fumigatus* sequence, Afu8g06570, is the ortholog of the other *aspergilli* sequences.

4.5.3 Comparison with OrthoMCL and OMA

4.5.3.1 Cluster Size Distribution

In this section, we compare the results of SynAPhy with that of OrthoMCL and OMA. We first look at the cluster size distributions of MCL_I15, SynAPhy_1_5, SynAPhy_3_5, OrthoMCL, and OMA to study the variation of the cluster sizes in different systems. Figure 20 depicts their cluster size distributions. For a cluster of size n shown on the x-axis, the y-axis shows the fraction of all sequences in a cluster of size at most n, i.e., cumulative fraction of sequences. OMA has a cluster size range 2 to 8. This is expected because OMA does not report singletons and clusters with paralogs, i.e., each cluster contains one protein from each input proteome. Most of the clusters fall in cluster size range 1 to 8 as is observed with the cumulative fraction of sequences approaching 1 in that range. This makes sense as we have eight genomes. One expects to see clusters with members from all the input genomes and no more than one member from an input genome as is the case with OMA. These clusters would be orthologous clusters. However, as we will see in Chapter 5, the eight genomes have large number of paralogs. Therefore, it is expected to see large clusters when paralogs are also clustered together. MCL_I15 has larger clusters than the others. Its largest cluster size is 312. The largest of SynAPhy_1_5 is 106, of SynAPhy_3_5 is 98, and of OrthoMCL is 86. The majority of the sequences in the large clusters are sugar transporters based on their Pfam domains.

4.5.3.2 Cluster Quality Metrics

We compute the four cluster quality metrics for OrthoMCL and OMA, and compare the results with that of MCL_I15, SynAPhy_1_5, and SynAPhy_3_5. The results are tabulated in Table 9. MCL_15, SynAPhy_1_5, and SynAPhy_3_5 cluster 99.98% of the eight proteomes, while OrthoMCL

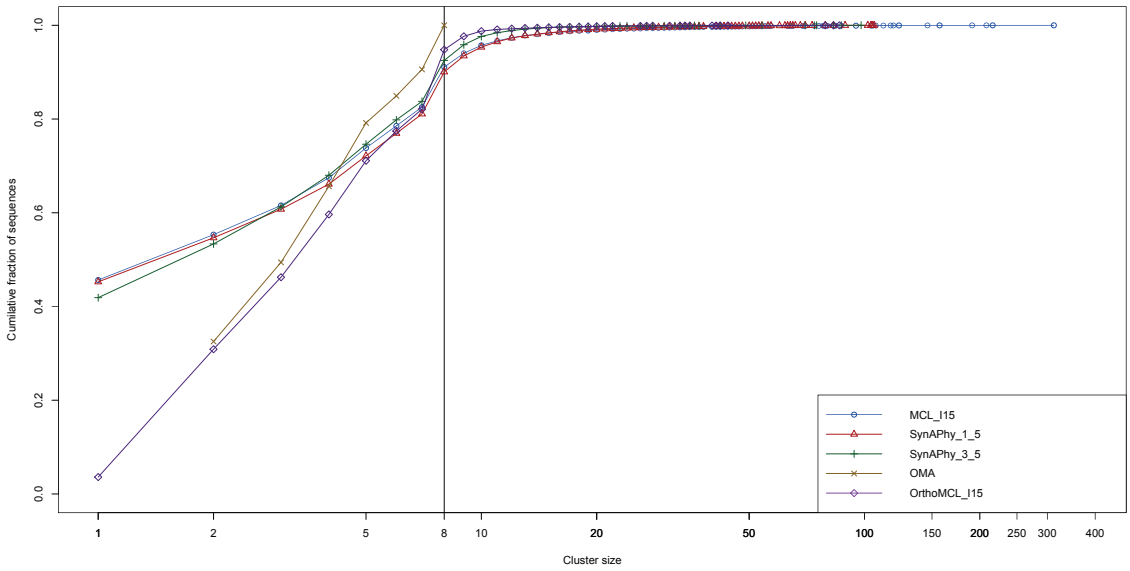

Figure 20: Cluster size distributions for different clusterings of the eight input fungal genomes. For a cluster of size n shown on the x-axis, the y-axis shows the cumulative fraction of sequences in a cluster of size at most n. The vertical line at cluster size 8 is drawn to highlight sequence distributions in cluster sizes for the eight input genomes.

and OMA cluster 78.57% and 66.17%, respectively. As mentioned earlier, OrthoMCL and OMA only cluster RBHs. OrthoMCL clusters between and within proteome RBHs, and OMA clusters between proteome RBHs. In addition, OMA does not include members from the same genome in the same cluster. These restrictions applied by OrthoMCL and OMA have direct effect on the four clustering metrics.

When we compare MCL, SynAPhy_1_5, and SynAPhy_3_5, SynAPhy_3_5 has the highest cluster edge density mean, δ_μ, and cluster normalized edge weight mean, $\overline{w}_\mu$. The δ_μ indicates that the clusters of SynAPhy_3_5 are denser than the ones of MCL_15 and SynAPhy_1_5. When we compare OrthoMCL and MCL, OrthoMCL has better δ_μ than OMA. In fact, OMA has $\delta_{\min}$ of 0, this means OMA has clusters with no edges in them. Recall that the edges are Blast `hits` with $E\text{–}value \leq 30$. OMA has exactly 75 clusters with cluster edge density value 0. However, when we compare $\overline{w}_\mu$ of OrthoMCL and OMA, OMA outperforms OrthoMCL. When we compare all the five clusterings, OrthoMCL outperforms in cluster edge density, and OMA outperforms in cluster normalized edge weight. This behavior is expected as both OrthoMCL and OMA filter out a large number of the input proteins.

We now move to discuss the cluster RBH and syntenic RBH edge densities. The minimums and maximums for cluster RBH and syntenic RBH edge densities of all the five systems are 0 and 1, respectively. When we compare MCL, SynAPhy_1_5, and SynAPhy_3_5. SynAPhy_3_5 has the the highest cluster RBH edge density mean δ_{rbh_μ}. This is expected as SynAPhy_3_5 scales

Clustering	δ_{min}	δ_{max}	δ_{μ}	δ_{σ}	$\overline{w}_{min}$	$\overline{w}_{max}$	$\overline{w}_{\mu}$	$\overline{w}_{\sigma}$	$\delta_{rbh_{\mu}}$	$\delta_{rbh_{\sigma}}$	$\delta_{syn\_rbh_{\mu}}$	$\delta_{syn\_rbh_{\sigma}}$
MCL	0.2857	1.0	0.9656	0.0896	0.01	1.0	0.4833	0.2938	0.6915	0.3617	0.1577	0.2252
SynAPhy_1_5	0.25	1.0	0.9657	0.0883	0.0083	1.0	0.4895	0.297	0.701	0.3522	**0.1764**	0.2441
SynAPhy_3_5	0.25	1.0	**0.9689**	0.0863	0.011	1.0	**0.5204**	0.3066	**0.7541**	0.3216	0.1508	0.2236
orthoMCL	0.5714	1.0	**0.9946**	0.0357	0.0221	1.0	0.5998	0.307	0.8395	0.3127	0.1778	0.2673
OMA	0.0	1.0	0.9936	0.0784	0.0	1.0	**0.6401**	0.3091	**0.8657**	0.3084	**0.2221**	0.3081

Table 9: Cluster quality metrics for MCL_I15, SynAPhy_1_5, SynAPhy_3_5, OrthoMCL, and OMA. The columns prefixed with δ show cluster edge density data. The columns prefixed with $\overline{w}$ show the normalized edge weights. The columns prefixed with δ_{rbh} and δ_{syn_rbh} show cluster RBH and syntenic RBH edge density data, respectively.

the similarity values of both RBHs and syntenic RBHs and SynaPhy_1_5 only scales the similarity values of syntenic RBHs. As a result, the cluster RBH edge densities of SynAPhy_3_5 are higher than those of SynAPhy_1_5. SynAPhy_1_5, however, has higher cluster syntenic RBH edge density mean, $\delta_{syn\_rbh_{\mu}}$, than SynAPhy_3_5. This is because when SynAPhy_3_5 scales the similarity values of RBHs, many RBHs are clustered with syntenic RBHs reducing the cluster syntenic RBH edge densities. When we compare the results of OrthoMCL and OMA, OMA outperforms in both metrics. This is expected because OMA does not cluster proteins from the same genome in the same cluster.

The four cluster quality metrics are good indicators of cluster qualities, however, they do not indicate whether orthologs and paralogs are resolved. Chapter 5 presents SynAVal, an orthology and paralogy evaluation framework with respect to RBHs and syntenic RBHs.

4.6 Related Work Revisited

In Section 2.6.2 we describe four orthology prediction systems that use synteny. In this section, we discuss how these systems are different from SynAPhy.

The first system OrthoParaMap [CY03] is different from SynAPhy in several aspects. First, OrthoParaMap computes syntenic blocks at the DNA level, whereas SynAPhy uses protein sequences. This aspect introduces a significant difference because in SynAPhy RBHs are treated as candidate orthologs and examined whether their corresponding genes are in syntenic blocks or not. Second, OrthoParaMap acts on gene families separately, whereas SynAPhy targets all genes in input genomes. Third, OrthoParaMap is a tree-based system, whereas SynAPhy is a graph-based system.

The second system PhyOP [GP06] does not use synteny in the core orthology prediction algorithm, it uses synteny to evaluate orthology.

The third system SYNERGY [WPFR07] is different from SynAPhy in several aspects. First, the integration of syntenic block resolution into sequence similarity scores in SYNERGY is a weighted measure, whereas in SynAPhy it is a boolean value. i.e., SYNERGY uses the fraction of the conserved neighboring genes corresponding to the proteins of interest, whereas SynAPhy for an RBH uses the decision of either in a syntenic block or not in a syntenic block. Second, SYNERGY has different protein sequence similarity scores and generates different similarity graphs. Third, SYNERGY recursively generates a similarity graph at each internal node of the species, whereas SynAPhy generates one graph for all the protein sequences of the genomes.

The fourth system PanOCT [FBB+12] is different from SynAPhy in several aspects. The

PanOCT weighted similarity graph is a directed graph for which gene neighborhood information is used to break near ties. The SynAPhy weighted similarity graph is undirected, and the inclusion of ties in SynAPhy is a boolean flag of the algorithm. Similar to SYNERGY, gene neighborhood conservation in PanOCT is a weighted measure, whereas in SynAPhy it is a boolean value. In PanOCT, gene neighborhood conservation is computed based on several criteria that is assumed to be conserved in species strains, whereas in SynAPhy it is relaxed to accommodate applicability on distant species.

In general, the main differences between orthology prediction systems that use synteny and SynAPhy are (1) the methodology of how synteny is computed, and (2) whether the algorithm is tree-based or graph-based.

4.7 Conclusion

This chapter presents SynAPhy, a novel graph-based algorithm for predicting orthologous clusters. The algorithm introduces synteny resolution as a mechanism to direct the MCL algorithm into clustering proteins corresponding to genes in conserved genomic regions. The algorithm was evaluated on eight fungal genomes. There are several key observations from the experimental results of SynAPhy.

First, using the mycoCLAP dataset, SynAPhy confirms the premise of RBHs and syntenic RBHs having similar molecular functions. There are 165 sequences from the mycoCLAP dataset that are present in the eight fungal genomes. Out of these 165, there are 42 RBHs out of which six are syntenic RBHs. The proteins on both sides of the 42 RBHs have similar functional annotations. Although mycoCLAP sequences cover a small percentage of the eight fungal genomes, they demonstrate the potential of identifying more functionally similar RBHs and syntenic RBHs with the emergence of more experimentally characterized datasets.

Second, the experiments on the similarity of proteomes show that RBHs are more similar than best hits, and that syntenic RBHs are more similar than RBHs. They also show that RBHs and syntenic RBHs have longer sequence alignments and better sequence alignment percent identities than best hits. Sequence similarity and alignment percent coverage and identity are strong indicators that RBHs and syntenic RBHs filter out "poor" best hits that might lead to erroneous orthologous relations.

Third, we observe from the results of the cluster quality metrics that SynAPhy generates clusters with more similar members than the MCL algorithm. This behavior is due to SynAPhy generating clusters with more RBH and syntenic RBH connections which is apparent in the cluster RBH and syntenic RBH edge densities. However, when compared to OrthoMCL and OMA, SynAPhy does not perform as well in the cluster quality metrics because OrthoMCL and OMA filter out a large number of the input proteins.

Finally, the cluster quality metrics show the performance of the algorithms in generating clusters with more similar members, however, they do not evaluate the orthology and paralogy of the cluster members. As there is no gold standard genome scale orthology dataset, we can not perform a

comprehensive analysis of the orthology prediction systems. Therefore, we propose in Chapter 5 SynAVal an evaluation framework for SynAPhy and other orthology prediction systems in generating orthologous clusters.

Chapter 5

The SynAVal Algorithm

The resolution of orthology and paralogy can be integrated into protein functional annotation pipelines to identify orthologous proteins in the presence of highly similar paralogous sequences. Recall that orthologs are genes in different organisms that speciated from a common ancestor while paralogs are genes in the same organism that duplicated from a common ancestor. Orthologs are functional counterparts and can be used in protein functional annotation pipelines to infer functions of newly sequenced genomes. A paralog, however, is free to evolve new functions. Both orthologs and paralogs can be highly similar sequences. Many orthology prediction systems use sequence similarity as the basis for predicting orthologs. Similar paralogous sequences pose a key challenge for orthology prediction systems because the wrong paralogous copies may be predicted as orthologs.

Protein functional annotation pipelines use the results of orthology prediction systems to identify functionally similar proteins. However, in the presence of confusions raised by paralogous relations, orthology prediction systems may mistakenly report functionally dissimilar proteins as orthologs. As a result, a protein functional annotation pipeline reports functionally dissimilar proteins as functionally similar.

Identifying confusions raised by paralogous relations is a hard problem. There is an ongoing effort by the scientific community [DGR+12] to evaluate the results of orthology prediction systems. A common evaluation challenge is the absence of genome scale gold standard orthology dataset, i.e., there is no dataset that has the orthologs of each and every gene in a genome when compared to other genomes.

In this chapter, we present SynAVal, an evaluation framework for an orthology prediction system. SynAVal identifies and reports confusions raised by paralogs. Given a set of genomes, SynAVal first identifies the paralogous relations in each genome. This step identifies and quantifies potential confusion cases. SynAVal then identifies conserved connections between genomes that are likely orthologs and uses these connections together with the paralogous relations to evaluate orthologous clusters and report confusion cases. We evaluate SynAVal on the eight fungal genomes listed in Section 4.3. The results show that SynAVal with synteny resolution is able to resolve confusions raised by the 9.1% of the proteins of the eight genomes, and 23.33% of the proteins from the eight genomes that are highly likely to raise confusions because they are paralogs with hits in other

genomes.

5.1 Example

In this section, we present an example cluster to motivate SynAVal. Figure 21 illustrates the example cluster. The cluster has two sequences from each of *A. fumigatus*, *A. niger*, *A. nidulans*, and *A. oryzae* genomes. It has two GH protein family 28 sequences present in the gold standard mycoCLAP database, and they are: PGX28A_ASPNG and PGX28X_ASPNG. They are both from *A. niger* genome and have exo-polygalacturonase functional annotation as annotated in the mycoCLAP database.

This cluster is generated by the MCL_I15 clustering. Recall that the MCL_I15 clustering is the clustering of the proteins of the eight fungal genomes with the MCL algorithm with inflation parameter value 15. There are two key observations in this cluster. The first one is the presence of paralogous proteins from each of the four genomes. The paralogs in *A. fumigatus* are Afu6g02980 and Afu8g07265, in *A. niger* are PGX28X_ASPNG and PGX28A_ASPNG, in *A. nidulans* are AN8761 and AN9045, and in *A. oryzae* are AO090026000784 and AO090113000199. An orthology prediction system should be able to predict which paralogous copies are the orthologs. The second observation is the RBH clique structure present in the cluster. An RBH clique is a subgraph in a graph in which any two vertices are connected to each other by an `rbh` edge. The RBH clique is formed from four proteins, one from each of the four genomes. The protein sequence identifiers of this RBH clique are Afu6g02980, PGX28X_ASPNG, AN8761, and AO090113000199. Note that the paralogs of these four sequences do not form an RBH clique or do not have any RBH connection in the cluster. RBHs are potential orthologs, and RBH cliques of size greater than two raise the confidence of orthology because there is no proof of non-orthology in the other genomes for a pair. For this reason, it is expected for an orthology prediction system to put the RBH clique members together in a cluster.

Given this example cluster, SynAVal reports the paralogs and the RBH clique in the cluster. The report indicates the paralogous proteins that raise confusions, the cluster with confusion, and the true orthologs based on RBH resolution.

5.2 SynAVal

We next describe SynAVal. We first start by describing how paralogous relations in each input genome are detected. Second, we describe how RBH and syntenic RBH cliques are detected. Finally, we put all the components of SynAVal together to describe the overall SynAVal framework.

5.2.1 Gene Copy Detection

Gene copies within a genome are the corresponding paralogous proteins. Therefore, a single-copy gene is a gene with no paralogs, and a multi-copy gene is a gene with paralogs. SynAVal detects the sets of gene copies in each input genome. A weighted similarity graph of a proteome P is given by $G(P)$ in which the vertices are the set of all the proteins in P and the edges connecting the proteins

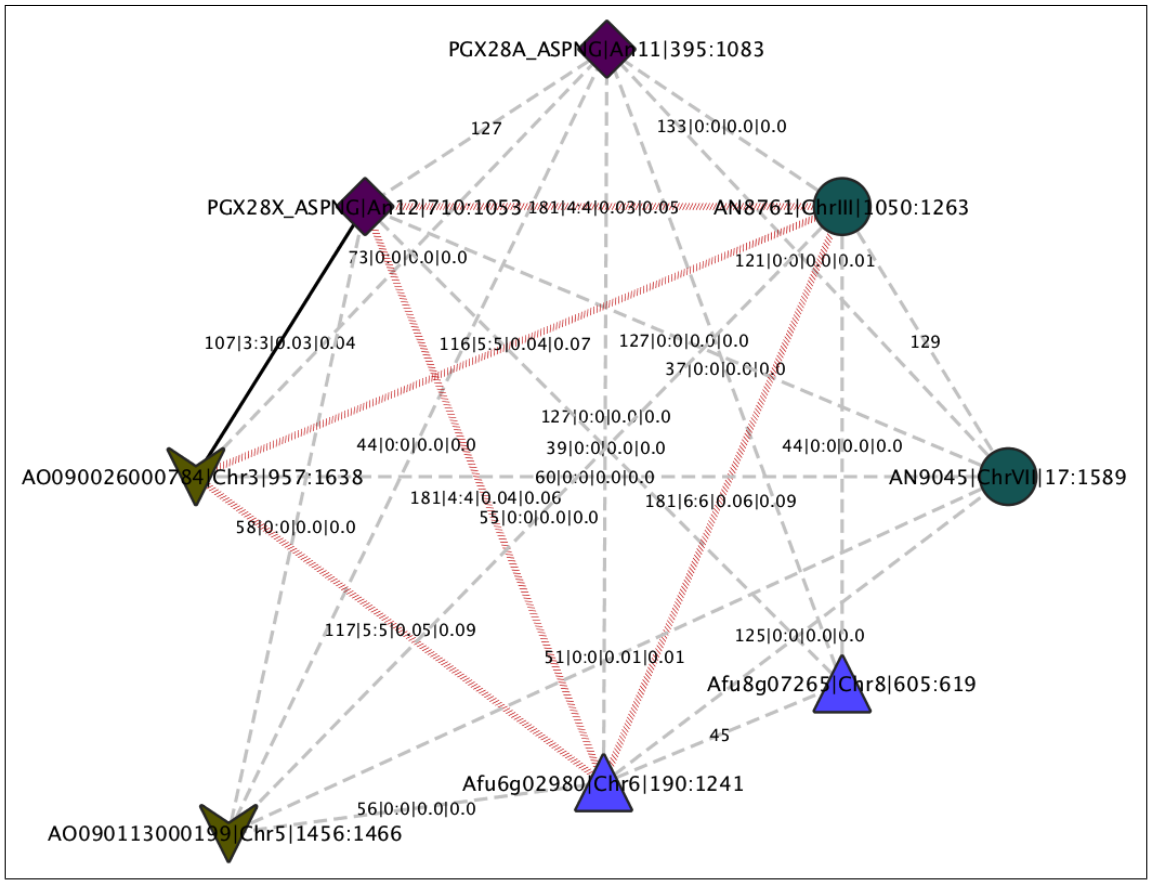

Figure 21: A cluster with GH family 28 sequences from the mycoCLAP database. Nodes represent proteins and edges represent their relationship types. There are three types of edges; `hit` edges, `rbh` edges, and `syn_rbh` edges. `hit` edges are depicted by grey dashed lines, `rbh` edges are depicted by solid black lines, and `syn_rbh` edges are depicted by red slashed lines. A node label of a protein `p` is a pipe-delimited string of the source database identifier, `p.chromosome`, and the location of the `p.head` on the gene ordered list of `p.chromosome` joined by the total number of genes on `p.chromosome`. Species are shape coded. Sequences from *A. fumigatus* are depicted as triangles, *A. niger* as diamonds, *A. nidulans* as ellipses, and *A. oryzae* as arrow heads. An edge is labeled with a pipe-delimited string $x|i:j|y|z$. x corresponds to their similarity value. i and j correspond to the number of RBHs and syntenic RBHs in their neighborhood, respectively. y corresponds to the neighborhood edge density, and z corresponds to the neighborhood normalized edge weights.

are the self all-versus-all Blast hits defined in Definition 2.5. The weights of the edges are the Blast similarity values defined in Definition 4.2. The gene copy detector then computes the connected components of $G(P)$ to detect the sets of gene copies. The standard Hopcroft and Tarjan connected component algorithm is used [HT73]. We refer to a connected component as a *bin*. Therefore, every pair of proteins in a bin are paralogs. The bin size indicates the number of gene copies in that bin.

A relaxed similarity value threshold generates a similarity graph with fewer connected components. These connected components may contain proteins that are not gene copies. To ensure connected components with only gene copies, we run a similarity value threshold experiment where we compute the connected components of graphs generated from a consecutive series of similarity values ranging from 3 to 181.

We evaluate the connected components with respect to Glycoside Hydrolase (GH) protein families. We first run the eight fungal genomes against Pfam [PCE+12] domain Hidden Markov Models (HMMs) with the *hmmscan* program from the HMMER 3.0 [Edd98] package, and consider only connected components with at least one protein with a GH family domain. For each genome, we compute the purity of the connected components. We use the purity metric $purity(\Omega, C)$ as defined in Definition 2.2, where Ω is the set of all the connected components with at least one protein with a GH family domain, and C is the set of GH families in the connected components. Proteins with no Pfam domain hits are counted in the purity measures. We assign the highest scoring GH family Pfam HMM for proteins with multiple GH family Pfam HMM hits.

Figure 22 depicts the scatter plots of gene copy bin purity with respect to GH families of the eight fungal genomes with similarity value range 3 to 181. Purity values for each genome varies. We do not observe a monotonic increase in purity with all genomes except for *S. cerevisiae* and *N. crassa*. For most genomes, similarity values in the range 3 to 5 result in low purity. *A. fumigatus* and *A. nidulans* do not reach a purity value of 1. We choose the similarity value threshold following two steps: (1) for each genome, we report the minimum similarity value that achieves 0.9 purity, and (2) from the list of minimums we report the maximum similarity value. *A. fumigatus* reaches 0.9 at 18, *A. nidulans* at 24, *A. niger* at 8, *A. oryzae* at 35, *C. albicans* for all similarity values, *S. cerevisiae* and *S. pombe* at 5, and *N. crassa* at 16. The set of the minimum similarity values that reach a purity value of 0.9 for the eight genomes is $\{18, 24, 8, 35, 5, 16\}$, and the maximum of these values is 35. We therefore set the similarity value threshold for the gene copy bin detector to 35, and report the connected components of the corresponding graph as the sets of gene copies.

5.2.2 RBH and Syntenic RBH Clique Detection

Reciprocal best hits are more similar to each other than they are to any other sequence when their corresponding proteomes are compared. For this reason, they are considered as potential orthologs. The confidence increases with syntenic RBHs, i.e., RBHs in syntenic blocks. The functional annotations of RBHs and syntenic RBHs of proteins in the gold standard mycoCLAP database proved this premise. In the eight fungal genomes, there are 42 RBHs out of which six are syntenic RBHs. The proteins on the both sides of the 42 RBHs have similar functional annotations as annotated by the mycoCLAP database. Although these numbers cover a small proportion of the proteins in the

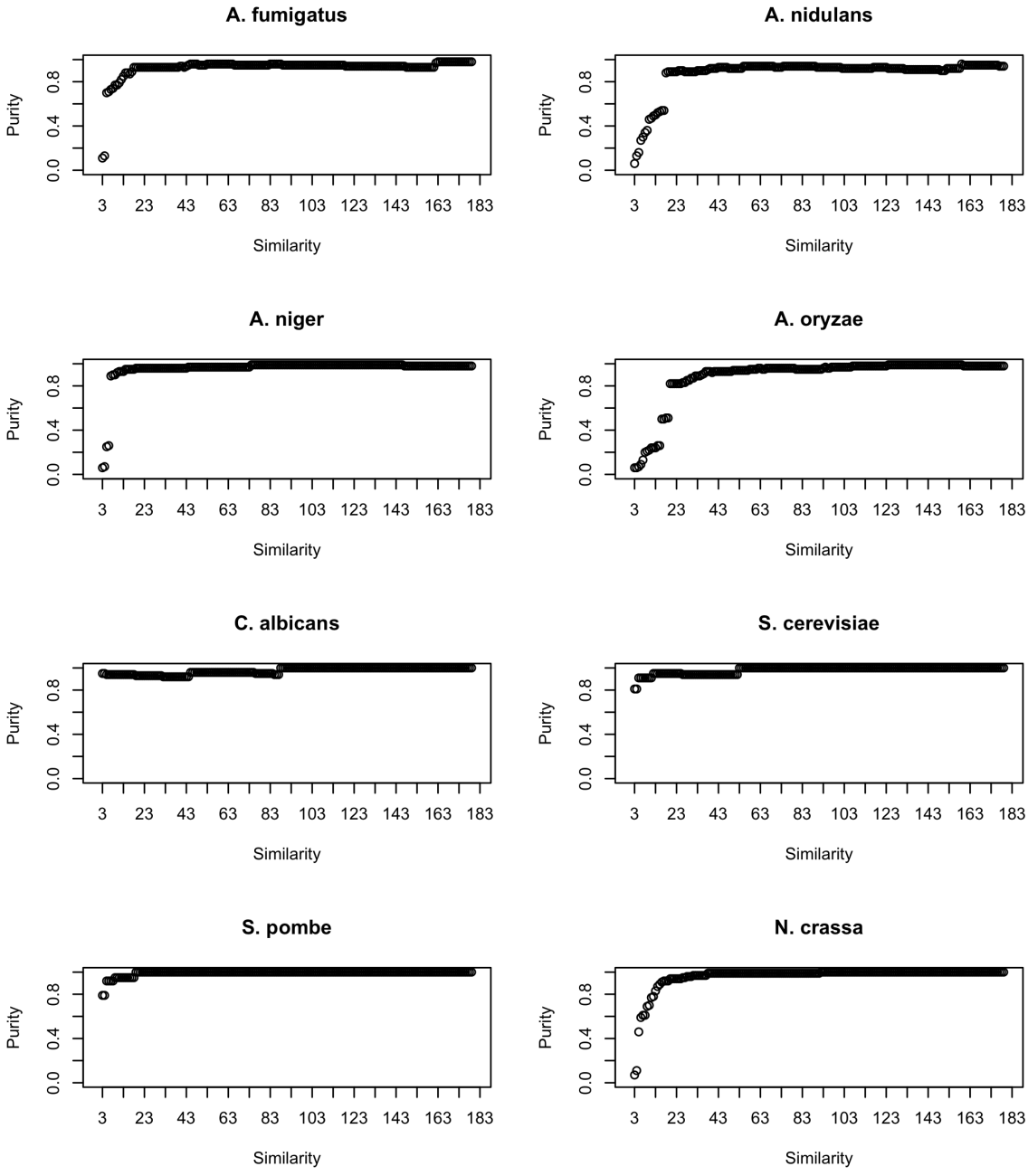

Figure 22: Gene copy bin purity with respect to glycoside hydrolase protein family domains for eight fungal genomes

eight genomes, they demonstrate the ability of RBHs and syntenic RBHs to identify orthologs.

When orthology is to be predicted for more than two genomes, RBH and syntenic RBH cliques can be used. An RBH clique is a subgraph in which any two vertices are connected to each other by a `rbh` edge. A syntenic RBH clique is a subgraph in which any two vertices are connected to each other by a `syn_rbh` edge. Such structures are highly likely to be orthologous clusters in which any two vertices are orthologs. When a clique has a member protein from each input genome, it represents a high confidence orthologous cluster, i.e., the orthologs of each protein in the other input genomes are identified with high confidence.

SynAVal computes RBH cliques and syntenic RBH cliques separately. For RBH cliques, the graph $G(V, E)$ has the set V of vertices that form RBH connections, and the set E of edges that are the RBH connections. For syntenic RBH cliques, the graph $G(V, E)$ has the set V of vertices that form syntenic RBH connections, and the set E of edges that are the syntenic RBH connections. To compute the RBH and syntenic RBH cliques, SynAVal applies the Bron–Kerbosch algorithm [BK73] on both graphs separately, and reports the cliques.

5.2.3 Framework

Figure 23 illustrates the SynAVal framework. SynAVal has three components: gene copy detector, clique detector, and orthology resolver. The gene copy detector detects the sets of gene copies in each input genome. The input to the gene copy detector is the similarity graph $G(P)$ of a proteome P built from the result of a self all-versus-all Blast search. The gene copy detector then applies the standard Hopcroft and Tarjan connected component algorithm on $G(P)$ to detect the connected components of $G(P)$. The output of the gene copy detector is the sets of gene copies or bins.

The clique detector algorithm detects RBH and syntenic RBH cliques. As described in Section 5.2.2, the clique detector component of SynAVal builds two graphs, one for RBH connections and one for syntenic RBH connections. The RBH connections are the output of the RBH detector of SynAPhy. The syntenic RBH connections are the output of the syntenic RBH detector of SynAPhy. The clique detector component then applies the Bron–Kerbosch algorithm [BK73] on the two graphs to compute the RBH and syntenic RBH cliques.

The inputs to the orthology resolver are the sets of gene copies, RBH and syntenic RBH cliques, and orthologous clusters generated by an orthology prediction system. The orthology resolver reports three findings: (1) paralogs that may raise confusions, (2) RBH and syntenic RBH cliques with respect to single- and multi-copy genes, and (3) clusters generated by an orthology prediction system with and without confusions.

The orthology resolver classifies two types of RBH and syntenic RBH cliques. The classification is based on gene copy data. The two types are: (1) cliques with only single-copy members, and (2) cliques with at least one multi-copy member. Members of cliques classified into the first type do not introduce confusion because there is no choice of paralogs. Members of cliques classified into the second type introduce confusion because the multi-copy member has multiple choices to be predicted as ortholog. The orthology resolver detects these two types of cliques in output orthologous clusters, and if the paralogs of the multi-copy members are clustered with the multi-copy cliques, it reports

confusion. In addition, SynAVal reports the frequency of multi-copy genes in RBH and syntenic RBH cliques that may raise confusions, and resolves the confusions by assigning the orthologs to be the members of RBH or syntenic RBH cliques.

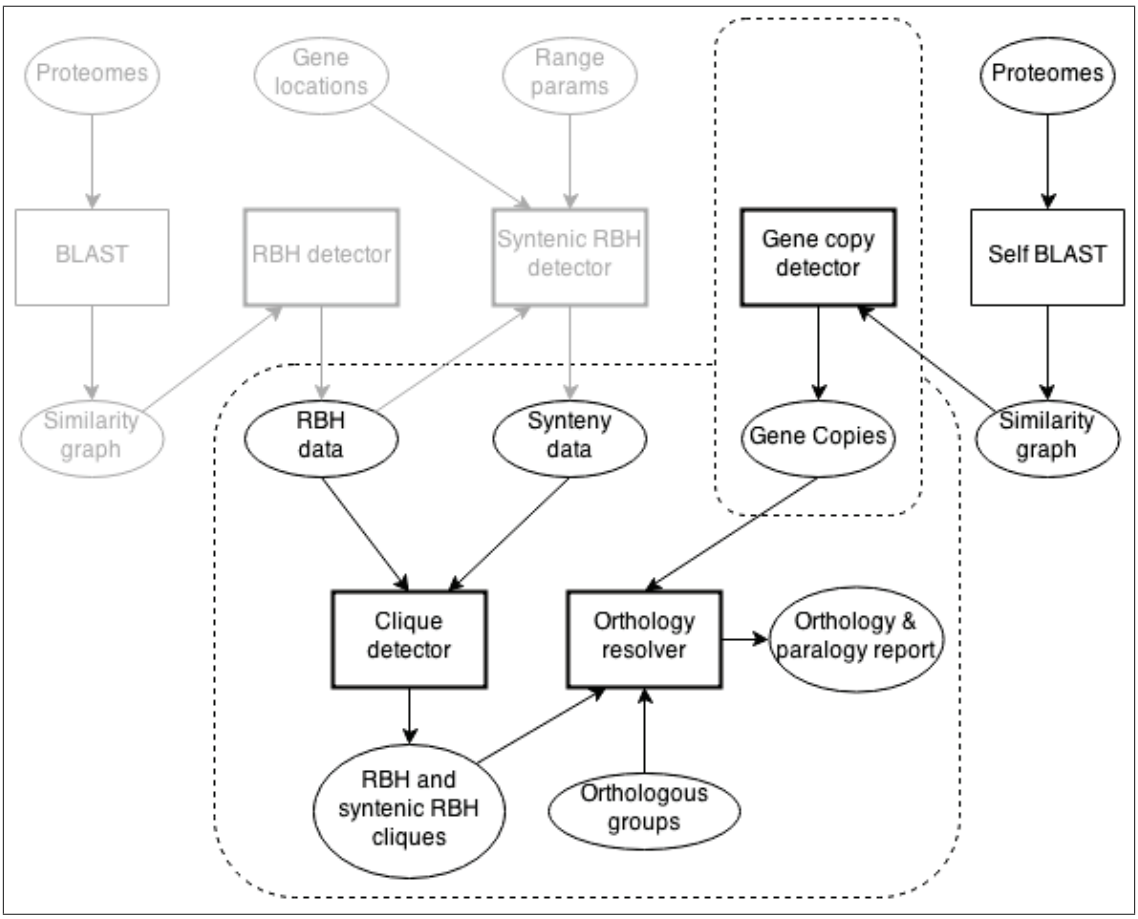

Figure 23: The SynAVal framework. SynAVal has three components: gene copy detector, clique detector, and orthology resolver. The input to the gene copy detector is the similarity graph built from the result of self all-versus-all Blast search. The output is the sets of gene copies. The inputs to the clique detector are the RBH and syntenic RBH connections detected by SynAPhy, and the outputs are RBH and syntenic RBH cliques. The inputs to the orthology resolver are the sets of gene copies, the RBH and syntenic RBH cliques, and orthologous clusters generated by an orthology prediction system. The output is a report of the (1) paralogs that raise confusion, (2) high confidence orthologs based on RBH and syntenic RBH cliques, and (3) set of output orthologous clusters with and without confusions.

5.3 Results

In this section, we present and discuss the results of SynAVal on the eight fungal genomes. First, we present the results of the gene copy detector. We show the proportion of single- and multi-copy genes in each of the eight genomes. Second we present the proportion of single- and multi-copy

Genome	# single-copy	% single-copy	# multi-copy	% multi-copy	Bin size			
					min	*max*	μ	σ
A. oryzae	6,928	58.21	4,974	41.79	2	114	4.35	7.98
A. nidulans	6,908	64.55	3,793	35.44	2	105	4.07	6.77
N. crassa	7,052	65.39	3,733	34.61	2	59	2.88	2.94
S. cerevisiae	3,843	65.58	2,019	34.44	2	86	3.08	4.61
A. fumigatus	6,505	66.49	3,278	33.51	2	98	3.79	5.96
A. niger	9,386	66.76	4,674	33.24	2	109	4.3	7.66
S. pombe	3,736	72.63	1,408	27.37	2	49	3.12	3.36
C. albicans	4,577	73.62	1,640	26.38	2	34	3.38	3.25

Table 10: Single- and multi-copy gene frequency in eight fungal genomes. The number and percentage of proteins corresponding to single-copy genes are shown in the columns labeled *# single-copy* and *% single-copy*, and the number and percentage of proteins corresponding to multi-copy genes are shown in the columns labeled *# multi-copy* and *% multi-copy*. The minimum, maximum, mean and standard deviation of bin sizes are shown in the columns labeled *min*, *max*, μ, and σ, respectively. The rows are sorted in descending order based on the *% multi-copy* column.

genes that are in RBH and syntenic RBH cliques. Finally, we analyze the five different techniques MCL, SynAPhy_1_5, SynAPhy_3_5, orthoMCL, and OMA with respect to (1) the presence of RBH and syntenic RBH cliques with paralogous copies in the generated clusters, and (2) the precision and recall of the techniques in clustering RBH and syntenic RBH connections together.

5.3.1 Gene Copies

Table 10 shows the frequency of single- and multi-copy genes in eight fungal genomes. The number and percentage of proteins corresponding to single-copy genes are shown in the columns labeled *# single-copy* and *% single-copy*, and the number and percentage of proteins corresponding to multi-copy genes are shown in the columns labeled *# multi-copy* and *% multi-copy*. The last four columns of the table show data on bin sizes. The bins are the connected components of the graphs for each proteome built from the results of self all-versus-all Blast searches. The minimum, maximum, mean and standard deviation of the bin sizes are shown in the columns labeled *min*, *max*, μ, and σ, respectively. The rows are sorted in descending order based on the *% multi-copy* column.

We observe that the percentage of proteins corresponding to multi-copy genes ranges from 26.38 for *C. albicans* to 41.79 with *A. oryzae*. *A. oryzae* has the largest bin size 114. The minimum bin size for each genome is two, but the maximums vary.

We notice that for all the eight genomes the maximum bin sizes are large. This led us to examine the Pfam domains of the proteins in the largest bins of all the eight genomes. The largest bins of all four *aspergilli* genomes are mainly composed of sugar transporters. The bin composition is different with the other four genomes, but we observe sugar transporters present in large bins. *N. crassa* and *S. pombe* have a majority of protein kinase domains in their largest bins. *S. cerevisiae* has a majority of yeast transposon domain in its largest bin, and *C. albicans* has a majority of leucine-rich repeat domains. The Pfam domain data of the large bins allows us to confirm the membership consensus across the eight genomes in terms of functional domains.

We observe that approximately 25% to 40% of the proteins in the eight fungal genomes correspond

Gene copy	# proteins	# proteins with hits	# proteins in RBH cliques	# proteins in syntenic RBH cliques
Single-copy	48,935	39,535	36,683	16,355
Multi-copy	25,519	24,744	19,460	5,774
Total	74,454	64,279	56,143	22,120

Table 11: Single- and multi-copy gene frequency in relation to protein sequence relation type. The total number of proteins in the eight fungal genomes is shown in the column labeled *# proteins*. The total number of proteins with hits in other proteomes is shown in the column labeled *# proteins with hits*. The total numbers of proteins that form part of RBH and synteny RBH cliques are shown in the columns labeled *# proteins in RBH cliques* and *# proteins in syntenic RBH cliques*, respectively.

to multi-copy genes. The large presence of multi-copy genes shows the presence of potential orthology and paralogy confusions for significant proportion of the input proteomes.

5.3.2 Gene Copies in RBH and Syntenic RBH Cliques

This section presents the results of RBH and syntenic RBH cliques. In particular, the presence of single- and multi-copy genes in RBH and syntenic RBH cliques, and how SynAVal helps resolve the orthology of cases with confusion. We first present the frequency of cases that may raise confusion with respect to multi-copy genes only. We then present the frequency of cases that may raise confusion with respect to single- and multi-copy genes.

5.3.2.1 Confusions Raised with Respect to Multi-copy Genes

Table 11 shows the frequency of single- and multi-copy genes in RBH and syntenic RBH cliques in the eight fungal genomes. The column labeled *# proteins* shows the total number of single- and multi-copy genes. The column labeled *# proteins with hits* shows the number of protein sequences with hits in other proteomes. The columns labeled *# proteins in RBH cliques* and *# proteins in syntenic RBH cliques*, respectively, show the number of protein sequences in RBH and syntenic RBH cliques. 86.33% of the input proteins from the eight fungal genomes have hits in other proteomes. These are the proteins that may have orthologs, the other 13.67% may not have orthologs because orthology comes with some level of sequence similarity. Out of the 86.33% of the input proteomes, 61.50% are single-copy genes, and 38.50% are multi-copy genes. The single-copy genes with hits do not introduce confusion as there is no choice of which copy is the ortholog. However, the multi-copy genes do introduce confusion. We consider two resolutions, one for proteins in RBH cliques, and another for proteins in syntenic RBH cliques. The syntenic RBH cliques are of higher confidence than the RBH cliques. But if RBH clique resolution is required, 78.65% of the multi-copy genes with hits are in RBH cliques. Therefore, the orthology of 78.65% of the proteins with confusion may be resolved. If syntenic RBH resolution is required, the orthology of 23.33% of the proteins with confusion may be resolved. With syntenic RBH cliques, the orthology of less proteins with confusion may be resolved than with RBH cliques. This is expected, as less proteins are found in syntenic RBH cliques.

5.3.2.2 Single- and Multi-copy Genes in RBH and Syntenic RBH Cliques

Recall from Section 5.2.3 that the orthology resolver reports two types of cliques: (1) cliques with only single-copy gene members, and (2) cliques with at least one multi-copy gene member. We present the frequency of RBH and syntenic RBH cliques with at least one multi-copy gene member. These cliques are the structures where the orthology of the multi-copy gene(s) to the other members of the clique is resolved. These members can be single- or multi-copy genes.

The clique detector component of SynAVal detected 18,324 RBH cliques in the graph built from the RBH connections of the eight fungal genomes. The members of these cliques form about 78.86% of the eight fungal proteomes. Out of the 18,324 RBH cliques, 7,546 are single-copy RBH cliques with the members forming about 39.44% of the eight fungal proteomes, and 10,778 are multi-copy RBH cliques with the members forming about 39.42% of the eight fungal proteomes. We now move to present the frequency of syntenic RBH cliques. There are 8,192 syntenic RBH cliques in the graph built from the syntenic RBH connections of the eight fungal genomes. The members of these cliques form about 29.86% of the eight fungal proteomes. Out of the 8,192 syntenic RBH cliques, 5,645 are single-copy syntenic RBH cliques with the members forming about 20.76% of the eight fungal proteomes, and 2,547 are multi-copy syntenic RBH cliques with the members forming about 9.1% of the eight fungal proteomes.

Therefore, with RBH cliques, SynAVal can resolve confusions raised by 39.42% of the proteins in the eight genomes, and with syntenic RBH cliques, SynAVal can resolve confusions raised by 9.1% of the proteins in the eight genomes.

We next show the frequency of the single- and multi-copy genes in RBH and syntenic RBH cliques of different sizes for each of the eight genomes. Table 12 shows the frequency of the multi-copy genes that are in RBH cliques of different sizes. The number of proteins in each genome is shown in the column labeled *# proteins*, and the percentage of multi-copy genes in each genome is shown in the column labeled *% multi-copy*. The percentage of multi-copy genes that are in RBH cliques of sizes ranging from 2 to 8 are shown in their respective columns. The percentage of multi-copy genes that are in all detected RBH cliques is shown in the column labeled *Total*. The totals of the columns are shown in the row labeled *Total*.

A. fumigatus has the largest fraction of multi-copy genes in RBH cliques with 86.00%, and *S. cerevisiae* has the lowest fraction with 67.90%. Consider the row labeled *Total*. The 34.27% of all the sequences in all the eight genomes are multi-copy genes. Most of the multi-copy genes are in cliques of size 2 and 8. Cliques of size 8 are strong indication of orthology because there is an identified ortholog in each of the remaining input genomes. This table shows strong presence of multi-copy genes in RBH cliques.

Table 13 shows the percentage of multi-copy genes in syntenic RBH cliques. *A. fumigatus* has the highest fraction of multi-copy genes in syntenic RBH cliques with 47.86%, and *S. pombe* has the lowest with 0%. Note that *S. pombe* does not have any multi-copy gene that is in syntenic RBH clique of any size. The other three *yeasts* (*N. crassa*, *C. albicans*, and *S. cerevisiae*) have significantly less than the four *aspergilli* genomes. The percentage of multi-copy genes in the four *aspergilli* genomes ranges from 27.75% to 47.86%. Although these percentages are significantly lower

Genome	# proteins	% multi-copy	RBH clique size							Total
			2	3	4	5	6	7	8	
A. fumigatus	9,783	33.51	12.54	15.89	19.89	12.97	6.35	5.74	12.63	**86.00**
A. nidulans	10,701	35.44	16.0	15.19	17.4	11.23	5.01	4.93	10.81	**80.57**
A. niger	14,060	33.24	19.53	15.15	14.96	9.2	4.21	4.13	8.99	**76.17**
A. oryzae	11,902	41.79	19.14	15.38	14.62	9.33	4.12	3.96	8.75	**75.29**
N. crassa	10,785	34.61	7.98	7.66	8.17	15.03	7.53	9.7	17.71	**73.77**
S. pombe	5,144	27.37	6.53	4.97	4.26	4.69	9.59	10.44	31.68	**72.16**
C. albicans	6,217	26.38	13.6	4.57	4.51	4.82	7.5	10.37	24.09	**69.45**
S. cerevisiae	5,862	34.44	13.87	6.29	4.06	4.85	5.75	10.45	22.63	**67.90**
Total	**74,454**	**34.27**	**14.8**	**12.26**	**12.77**	**9.99**	**5.7**	**6.49**	**14.26**	**76.26**

Table 12: Multi-copy gene frequency in RBH cliques of different sizes. The genomes are shown in the column labeled *Genome*. The number of proteins in each genome is shown in the column labeled *# proteins*. The percentage of multi-copy genes in each genome is shown in the column labeled *% multi-copy*. The percentages of multi-copy genes in RBH cliques of different sizes indicated by column headings are shown in the merged column labeled *RBH clique size*. The totals of the rows and columns are shown in the column and row labeled *Total*, respectively.

Genome	# proteins	% multi-copy	Syntenic RBH clique size				Total
			2	3	4	5	
A. fumigatus	9,783	33.51	12.48	17.91	17.24	0.24	**47.86**
A. nidulans	10,701	35.44	6.64	13.05	14.92	0.24	**34.85**
A. oryzae	11,902	41.79	7.02	10.86	12.26	0.14	**30.28**
A. niger	14,060	33.24	6.01	9.2	12.39	0.15	**27.75**
N. crassa	10,785	34.61	0.43	0.56	0.46	0.27	**1.74**
C. albicans	6,217	26.38	0.43	0.06	0	0	**0.49**
S. cerevisiae	5,862	34.44	0.4	0	0	0	**0.4**
S. pombe	5,144	27.37	0	0	0	0	**0**
Total	**74,454**	**34.27**	**5.18**	**8.13**	**9.16**	**0.16**	**22.63**

Table 13: Multi-copy gene frequency in syntenic RBH cliques of different sizes. The table structure of this table is similar to that of Table 12, but it shows the percentage of multi-copy genes in syntenic RBH cliques of different sizes.

than the multi-copy genes in RBH cliques, they still account for a good proportion of the multi-copy genes.

Table 14 and Table 15 show the percentages of single-copy genes in RBH and syntenic RBH cliques, respectively. We show the single-copy results for completeness. We are, in particular, interested whether single-copy genes have more presence in syntenic RBH cliques in the *yeast* genomes than the multi-copy genes. As shown in Table 15, *S. pombe* has no single-copy gene that is in a syntenic RBH clique. The other three *yeast* genomes do not have a high percentage of single-copy genes in syntenic RBH cliques. However, all the eight genomes have significant percentages of their single-copy genes in RBH cliques. The results of the single-copy genes align with the results of the multi-copy genes presented in Table 10. The orthology of single-copy genes is trivial to resolve as there is no choice of which gene copy is the ortholog. Therefore, when present in single-copy RBH or syntenic RBH cliques, there is no confusion. However, when present in multi-copy RBH or syntenic RBH cliques, there is confusion.

Genome	# proteins	% single-copy	Clique size							Total
			2	3	4	5	6	7	8	
A. fumigatus	9,783	66.49	7.16	8.55	12.88	16.02	9.1	11.18	23.04	**87.92**
A. nidulans	10,701	64.55	7.53	8.37	12.19	15.2	9.05	10.35	21.76	**84.44**
S. pombe	5,144	72.63	3.48	3.05	2.92	5.09	11.67	14.64	39.59	**80.43**
A. oryzae	11,902	58.21	10.67	7.02	9.93	13.6	7.75	8.62	21.33	**78.91**
S. cerevisiae	5,862	65.58	9.0	3.43	2.42	3.51	6.3	14.57	38.2	**77.44**
C. albicans	6,217	73.62	8.08	2.95	2.77	4.37	8.5	15.53	33.25	**75.46**
N. crassa	10,785	65.39	4.01	4.25	6.1	13.81	8.03	8.47	19.7	**64.36**
A. niger	14,060	66.76	6.59	5.53	8.32	10.95	6.12	7.2	15.91	**60.62**
Total	**74,454**	**65.72**	**7.1**	**5.76**	**7.99**	**11.36**	**8.09**	**10.48**	**24.18**	**74.96**

Table 14: Single-copy gene frequency in RBH cliques. This table is similar to the table structure of Table 12, but it shows the percentages of single-copy genes in RBH cliques.

Genome	# proteins	% single-copy	Clique size				Total
			2	3	4	5	
A. fumigatus	9,783	66.49	13.21	26.46	29.47	0.32	**69.45**
A. nidulans	10,701	64.55	10.02	22.22	27.61	0.29	**60.13**
A. oryzae	11,902	58.21	8.3	18.43	26.83	0.32	**53.88**
A. niger	14,060	66.76	6.26	13.31	20.16	0.23	**39.96**
N. crassa	10,785	65.39	0.34	0.85	0.64	0.28	**2.11**
C. albicans	6,217	73.62	0.59	0.02	0	0	**0.61**
S. cerevisiae	5,862	65.58	0.6	0	0	0	**0.6**
S. pombe	5,144	72.63	0	0	0	0	**0**
Total	**74,454**	**65.72**	**5.7**	**11.94**	**15.57**	**0.21**	**33.42**

Table 15: Single-copy gene frequency in syntenic RBH cliques. This table is similar to the table structure of Table 12, but it shows the percentages of single-copy genes in syntenic RBH cliques.

5.3.3 Gene Copies in Clusters

Gene copies and RBH and syntenic RBH cliques allow SynAVal to evaluate techniques that generate orthologous clusters. The evaluation we present in this section is (1) confusion cases with respect to clusters, and (2) confusion cases with respect to single- and multi-copy genes. We evaluate the results of the five techniques presented in Chapter 4: MCL, SynAPhy_1_5, SynAPhy_3_5, orthoMCL, and OMA.

5.3.3.1 Confusions Raised with Respect to Clusters

SynAVal detects and reports protein clusters that have multi-copy RBH or syntenic RBH cliques together with the paralogs of the multi-copy gene(s). These clusters are the ones that raise confusion as to which copy is the true ortholog of the RBH or syntenic RBH members. Recall that when a multi-copy RBH or syntenic RBH clique is clustered with the paralogs of the multi-copy gene(s), the clique members are the proteins forming high confidence orthologous relations.

We present the clustering results of MCL, SynAPhy_1_5, SynAPhy_3_5, orthoMCL, and OMA techniques. We classify the generated clusters into three categories; (1) clusters that do not have RBH or syntenic RBH cliques present; (2) clusters with RBH or syntenic RBH cliques in which none of the copies of a multi-copy gene member of the cliques is in the cluster; and (3) clusters with RBH or syntenic RBH cliques in which at least one copy of a multi-copy gene member of the cliques is in the cluster. SynAVal can not evaluate any member of the clusters in the first category as there is no RBH or syntenic RBH presence. The RBH or syntenic RBH members of the clusters in the second category do not raise confusion because the copies of the multi-copy genes are not in the cluster. SynAVal for this category does not report any confusion. However, it does report a confusion for clusters in the third category as the copies of the multi-copy genes in RBH or syntenic RBH cliques are in the clusters.

Table 16 shows the number of clusters in the three categories for the five techniques we are analyzing with respect to RBH cliques. The number of clusters generated by each technique is shown in the column labeled *# clusters*. The number of clusters classified into category one is shown in the column labeled *# clusters with no RBH cliques*. The number of clusters classified into category two is shown in the column labeled *# clusters with RBH cliques no multi-copy gene(s) (no confusion)*, and the number of clusters classified into category three is shown in the column labeled *# clusters with RBH cliques and multi-copy gene(s) (confusion)*. We show the number and percentage of clustered sequences that are in category three clusters in the column labeled *sequences with RBH cliques with confusion – # (%)*.

The results of MCL, SynAPhy_1_5, and SynAPhy_3_5 show that by integrating the RBH resolution to the MCL clustering algorithm, a smaller percentage of the clustered sequences are in clusters with confusion. SynAVal was able to evaluate the performance increase of SynAPhy_3_5 over SynAPhy_1_5 and SynAPhy_1_5 over MCL. SynAPhy_1_5 has more output clusters than MCL, and in turn SynAPhy_3_5 has more than SynAPhy_1_5. This is because SynAPhy_1_5 generates finer grained clusters than MCL with syntenic RBH connections clustered together, and in turn

Technique	# clusters	# clusters with no `rbh` cliques	# clusters with `rbh` cliques no multi-copy gene(s) (no confusion)	# clusters with `rbh` cliques and multi-copy gene(s) (confusion)	sequences in `rbh` cliques with confusion – # (%)
MCL	18,796	9,724	6,326	2,746	22,493 (30%)
SynAPhy_1_5	18,870	9,540	6,408	2,922	22,282 (29%)
SynAPhy_3_5	20,611	9,461	8,145	3,005	17,855 (24%)
orthoMCL	13,133	1,647	9,646	1,840	11,477 (19%)
OMA	12,389	1,152	11,237	NA	NA

Table 16: Proportion of confusions reported in clusters with RBH clique resolution in five orthology prediction systems

SynAPhy_3_5 generates finer grained clusters than SynAPhy_1_5 with RBH and syntenic RBH connections clustered together. Although SynAPhy_1_5 and SynAPhy_3_5 have more clusters than MCL and SynAPhy_1_5, respectively, the number of clusters with no RBH cliques in SynAPhy_1_5 against MCL and SynAPhy_3_5 against SynAPhy_1_5 decreases. Another positive result with SynAPhy_1_5 over MCL and SynAPhy_3_5 over SynAPhy_1_5 is that the number of clusters with no confusion increases. Although the number of clusters with confusion increases with SynAPhy_1_5 over MCL and SynAPhy_3_5 over SynAPhy_1_5, the percentage of sequences in these clusters decreases from 30% to 24%.

orthoMCL and OMA have smaller number of clusters than MCL, SynAPhy_1_5, and SynAPhy_3_5. Consequently, they have smaller number of clusters with no RBH cliques. OMA compared to orthoMCL has more clusters with no confusion. This is expected as OMA does not generate clusters with more than one member from a specific genome. For this reason, the results of clusters with confusion, and the total number of sequences in clusters with confusion is Not Applicable (NA) for OMA. orthoMCL has significantly smaller percentage of sequences that are in clusters with confusion compared to any of MCL, SynAPhy_1_5, or SynAPhy_3_5. This is expected because orthoMCL filters out a large number of the input proteins.

Table 17 shows the number of clusters in the three cluster categories for the five techniques with respect to syntenic RBH cliques. The table structure is the same as that of Table 16. The results follow the same pattern as that of RBH cliques in the number of sequences that are in clusters with confusion. SynAPhy_3_5 has smaller percentage of the clustered sequences that are in clusters with confusions than SynAPhy_1_5, and SynAPhy_1_5 in turn has smaller percentage than MCL. Syntenic RBH resolution provides higher confidence than RBH resolution. If syntenic RBH resolution is required, then the results of the Table 17 can be used, and if RBH resolution is required, then the results of the Table 16 can be used.

5.3.3.2 Confusion Raised in Clusters with Respect to Single- and Multi-copy Genes

In this section, we present the frequency of confusions raised and detected by SynAVal in the clusters of the five techniques with respect to single- and multi-copy genes. The following notations help us present the results in tabular form.

Let P be the union of the proteomes.

Technique	# clusters	# clusters with no `syn_rbh` cliques	# clusters with `syn_rbh` cliques no multi-copy gene(s) (no confusion)	# clusters with `syn_rbh` cliques and multi-copy gene(s) (confusion)	sequences in `syn_rbh` cliques with confusion – # (%)
MCL	18,796	12,880	4,992	924	4,561 (6.1%)
SynAPhy_1_5	18,870	12,329	5,490	1,051	3,406 (4.5%)
SynAPhy_3_5	20,611	14,121	5,723	773	2,475 (3.3%)
orthoMCL	13,133	6,814	6,002	317	1,204 (2.1%)
OMA	12,389	6,148	6,241	NA	NA

Table 17: Proportion of confusions reported with syntenic RBH clique resolution in five orthology prediction systems

Let $\mathcal{C}$(technique) be the proteins clustered by the *technique*, that is, the union of the clusters.

Let O be the subset of single-copy genes in P.

Let M be the subset of multi-copy genes in P.

For X being P, O, or M let $R_X = \{p \in X \mid p \textit{ in RBH clique}\}$.

For X being P, O, or M let $S_X = \{p \in X \mid p \textit{ in syntenic RBH clique}\}$.

Let $\bar{P} = \{p \in P \mid p \textit{ has a hit in another proteome}\}$.

Let $\bar{O} = O \cap \bar{P}$.

Let $\bar{M} = M \cap \bar{P}$.

Recall from Table 11 that the total number of proteins in the eight fungal genomes is 74,454. MCL, SynAPhy_1_5, and SynAPhy_3_5 cluster 99.99% of the proteins in the eight fungal genomes. orthoMCL clusters 78.57%, and OMA clusters 66.18%. We show the total number of single- and multi-copy genes with hits in other genomes in Table 11. MCL, SynAPhy_1_5, and SynAPhy_3_5 cluster 100% of the proteins with hits in other proteomes. orthoMCL clusters 89.27% of the single-copy gene with hits in other genomes, and 89.02% of the multi-copy genes with hits in other genomes. While OMA clusters 77.36% of the single-copy genes with hits in other genomes, and 75.21% of the multi-copy genes with hits in other genomes. These data show that both orthoMCL and OMA are leaving out a significant proportion of the input proteins with hits in other genomes, with orthoMCL leaving out about 10% of both single- and multi-copy genes, and OMA leaving out about 25% of both single- and multi-copy genes. With a large proportion of both single- and multi-copy genes being unclustered with both orthoMCL and OMA, there is less room for errors as we show later in the chapter. The situation is more critical with multi-copy genes because multi-copy genes are the ones that raise confusion.

In Table 18 we show the frequency of single- and multi-copy genes clustered by the five techniques we are analyzing. In what follows, we look at the frequency of multi-copy genes in RBH cliques clustered with at least one of their copies. In this case, the paralogs are clustered with the orthologs. The total number of multi-copy genes in RBH cliques is shown in Table 11 and it is 19,460. Table 19

Technique	$\|\mathcal{C}(\text{technique})\|$	$\|\mathcal{C}(\text{technique}) \cap \bar{O}\|$	$\|\mathcal{C}(\text{technique}) \cap \bar{M}\|$
MCL	74,443	39,535	24,744
SynAPhy_1_5	74,443	39,535	24,744
SynAPhy_3_5	74,443	39,535	24,744
orthoMCL	58,499	35,293	22,027
OMA	49,270	30,588	18,609

Table 18: Frequency of clustered single- and multi-copy genes with hits in other proteomes. The number of proteins clustered by a technique is shown in the column labeled $|\mathcal{C}(\text{technique})|$. The number of single-copy proteins clustered with hits in other proteomes is shown in the column labeled $|\mathcal{C}(\text{technique}) \cap \bar{O}|$. The frequency of multi-copy proteins clustered with hits in other proteomes in shown in the column labeled $|\mathcal{C}(\text{technique}) \cap \bar{M}|$.

Technique	$\|\mathcal{C}(\text{technique}) \cap R_{\bar{M}}\|$ without copies	$\|\mathcal{C}(\text{technique}) \cap R_{\bar{M}}\|$ with copies	$\|\mathcal{C}(\text{technique}) \cap R_{\bar{M}}\|$ not clustered
MCL	11,083	8,377	0
SynAPhy_1_5	12,824	6,636	0
SynAPhy_3_5	17,110	2,350	0
orthoMCL	16,783	2,590	87
OMA	17,078	0	2,382

Table 19: Frequency of multi-copy genes in RBH cliques clustered with and without copies. The frequencies of multi-copy genes forming part of RBH cliques that are clustered with their copies, without their copies, and not clustered are shown in the columns labeled $|\mathcal{C}(\textit{technique}) \cap R_{\bar{M}}|$ *without copies*, $|\mathcal{C}(\textit{technique}) \cap R_{\bar{M}}|$ *with paralogs*, and $|\mathcal{C}(\textit{technique}) \cap R_{\bar{M}}|$ *not clustered*, respectively.

shows the frequency of multi-copy genes in RBH cliques clustered with their copies, without their copies, and not clustered by the five techniques we are analyzing. Consider MCL, SynAPhy_1_5, and SynAPhy_3_5. The three techniques cluster all the mutli-copy genes in RBH cliques. MCL, SynAPhy_1_5, and SynAPhy_3_5 cluster $11,083$, $12,824$, and $17,110$, of the multi-copy genes in RBH cliques without their copies, respectively. SynAPhy_3_5 clusters more multi-copy genes without their copies than SynAPhy_1_5, and in turn SynAPhy_1_5 clusters more. This is a good indication as with SynAPhy_3_5 less multi-copy genes are being clustered with their copies. Consider orthoMCL and OMA. orthoMCL is clustering $16,783$ multi-copy genes without their copies, and $2,590$ with their copies. The latter number is more than that of SynAPhy_3_5, which is $2,350$. OMA does not cluster paralogs together, therefore the number of multi-copy genes clustered with their copies is 0. Note that similar to MCL, SynAPhy_1_5, and SynAPhy_3_5, orthoMCL includes proteins from the same genome that are very similar to each in the protein sequence similarity graph for the MCL algorithm to cluster. Therefore, it is expected that orthoMCL puts together copies or in-paralogs. The comparison presented here is not a fair comparison as to whether the correct copy of the paralogs are being put together in a cluster to be considered as orthologs by a technique. We will present a fair comparison of the techniques later in the chapter.

Table 20 shows the frequency of multi-copy genes in syntenic RBH cliques clustered with their copies, without their copies, and not clustered by the five techniques we are analyzing. The table structure is similar to that of Table 19, except that this one shows multi-copy genes in syntenic RBH cliques. The total number of multi-copy genes in syntenic RBH cliques is $5,774$ shown in Table 11.

Technique	$\|\mathcal{C}(\text{technique}) \cap S_{\bar{M}}\|$ without copies	$\|\mathcal{C}(\text{technique}) \cap S_{\bar{M}}\|$ with copies	$\|\mathcal{C}(\text{technique}) \cap S_{\bar{M}}\|$ not clustered
MCL	4,356	1,418	0
SynAPhy_1_5	5,762	12	0
SynAPhy_3_5	5,763	11	0
orthoMCL	5,564	209	1
OMA	5,675	0	99

Table 20: Frequency of multi-copy genes in syntenic RBH cliques clustered with and without copies. The frequencies of multi-copy genes forming part of syntenic RBH cliques that are clustered with their paralogs, without their paralogs, and not clustered are shown in the columns labeled $|\mathcal{C}(technique) \cap S_{\bar{M}}|$ *without copies*, $|\mathcal{C}(technique) \cap S_{\bar{M}}|$*with paralogs*, and $|\mathcal{C}(technique) \cap S_{\bar{M}}|$ *not clustered*, respectively.

The same behavior as that of multi-copy genes in RBH cliques is observed with the multi-copy genes in syntenic RBH cliques by the five techniques. SynAPhy_3_5 clusters one less multi-copy gene with its copy than SynAPhy_1_5, and SynAPhy_1_5 clusters $1,406$ less than that of MCL. orthoMCL clusters 198 more multi-copy genes with their copies than SynAPhy_3_5. OMA does not cluster copies.

Table 19 and Table 20 show the frequency of multi-copy genes in RBH and syntenic RBH cliques, respectively, clustered by the five techniques with and without their copies. This comparison, however, does not show whether the true copies, i.e., the ones that form part of RBH or syntenic RBH cliques, are clustered with their ortholog. Table 21 and Table 22 show the mis-clustered multi-copy genes that are in RBH and syntenic RBH cliques, respectively, by the five techniques we are analyzing. A mis-clustered multi-copy gene is a multi-copy gene that is clustered with the RBH or syntenic RBH clique member of one of its copies. In Table 21, consider the techniques MCL, SynAPhy_1_5, and SynAPhy_3_5. MCL has $14,133$ mis-clustered multi-copy genes in RBH cliques, SynAPhy_1_5 has $12,831$, and SynAPhy_3_5 has $7,220$. We notice that the number of mis-clustered multi-copy genes with SynAPhy_3_5 is significantly less than that of SynAPhy_1_5, and that of SynAPhy_1_5 is less than MCL. This is because we are scaling the similarity values of the RBH connections with SynAPhy_1_5 and SynAPhy_3_5 directing MCL to cluster the RBH connections together. Consider the techniques orthoMCL and OMA. OMA has less mis-clustered multi-copy genes in RBH cliques than orthoMCL, and it has less than any of the MCL, SynAPhy_1_5, and SynAPhy_3_5. We present the percentages of the mis-clustered multi-copy genes in the column labeled *% wrt* $|\mathcal{C}(technique) \cap R_{\bar{M}}|$ with respect to the multi-copy genes clustered by a technique in the column labeled $|\mathcal{C}(\text{technique}) \cap R_{\bar{M}}|$. We notice an improvement in SynAPhy_1_5 over MCL, and a significant improvement in SynAPhy_3_5 over SynAPhy_1_5. SynAPhy_3_5 dropping to 37.10% mis-clustered multi-copy genes. However, this number is less with orthoMCL and OMA, with OMA leading with only 15.32% of mis-clustered multi-copy genes in RBH cliques.

The same behavior is observed by the five techniques with mis-clustered multi-copy genes in syntenic RBH cliques shown in Table 22. With syntenic RBH cliques, the percentages are lower because as we saw in Chapter 4 syntenic RBHs are more similar to each other than RBHs. MCL mis-clusters 61.38% of the multi-copy genes in syntenic RBH cliques, SynAPhy_1_5 mis-clusters

Technique	$\|\mathcal{C}(\text{technique}) \cap R_{\bar{M}}\|$ mis-clustered	$\|\mathcal{C}(\text{technique}) \cap R_{\bar{M}}\|$	% wrt $\|\mathcal{C}(\text{technique}) \cap R_{\bar{M}}\|$
MCL	14,133	19,470	72.62
SynAPhy_1_5	12,831	19,470	65.93
SynAPhy_3_5	7,220	19,470	37.10
orthoMCL	5,109	19,373	26.37
OMA	2,616	17,078	15.32

Table 21: Frequency of mis-clustered multi-copy genes in RBH cliques. The number of multi-copy genes in RBH cliques that are mis-clustered is shown in the column labeled $|\mathcal{C}(technique) \cap R_{\bar{M}}|$ *mis-clustered*. The number of multi-copy genes in RBH cliques that are clustered by a technique is shown in the column labeled $|\mathcal{C}(\text{technique}) \cap R_{\bar{M}}|$, and the percentage of the mis-clustered multi-copy genes in RBH cliques with respect to the clustered multi-copy genes in RBH cliques is shown in the column labeled *% wrt* $|\mathcal{C}(technique) \cap R_{\bar{M}}|$.

Technique	$\|\mathcal{C}(\text{technique}) \cap S_{\bar{M}}\|$ mis-clustered	$\|\mathcal{C}(\text{technique}) \cap S_{\bar{M}}\|$	% wrt $\|\mathcal{C}(\text{technique}) \cap S_{\bar{M}}\|$
MCL	3,544	5,774	61.38
SynAPhy_1_5	2,104	5,774	36.44
SynAPhy_3_5	1,144	5,774	19.81
orthoMCL	828	5,773	14.34
OMA	48	5,675	0.83

Table 22: Frequency of mis-clustered multi-copy genes in syntenic RBH cliques. The number of multi-copy genes in syntenic RBH cliques that are mis-clustered is shown in the column labeled $|\mathcal{C}(technique) \cap S_{\bar{M}}|$ *mis-clustered*. The number of multi-copy genes in syntenic RBH cliques that are clustered by a technique is shown in the column labeled $|\mathcal{C}(\text{technique}) \cap S_{\bar{M}}|$, and the percentage of the mis-clustered multi-copy genes in syntenic RBH cliques with respect to the clustered multi-copy genes in syntenic RBH cliques is shown in the column labeled *% wrt* $|\mathcal{C}(technique) \cap S_{\bar{M}}|$.

36.44%, SynAPhy_3_5 mis-clusters 19.81%, orthoMCL mis-clusters 14.34%, and OMA mis-clusters 0.83%. Note that these percentages are with respect to the number of multi-copy genes in syntenic RBH cliques that are clustered by each technique. The same behavior is observed with that of mis-clustered multi-copy genes in RBH cliques. OMA has the least mis-clustered multi-copy genes, followed by orthoMCL, SynAPhy_3_5, SynAPhy_1_5, and MCL.

Table 21 and Table 22 show that of multi-copy genes in RBH and syntenic RBH cliques, respectively. We next present a comprehensive analysis of all RBH and syntenic RBH clique connections. We compute the F-measures of the five techniques we are analyzing with respect to RBH and syntenic RBH cliques., i.e., the expected results for each and every pair of proteins in the eight genomes are with respect to RBH or syntenic RBH cliques. We call a pair of proteins a

- True Positive (TP) if they are in the same clique and are clustered together in the same cluster;
- False Positive (FP) if they are not in the same clique but are clustered together in the same cluster;
- True Negative (TN) if they are not in the same clique and are not clustered together in the same cluster; and

Technique	TP	FP	TN	FN	Precision	Recall	F-measure
MCL	125,563	107,375	619,865,577	8,435	0.54	0.94	0.69
SynAPhy_1_5	126,211	73,415	619,899,537	7,787	0.63	0.94	0.75
SynAPhy_3_5	127,713	52,586	619,920,366	6,285	0.71	0.95	0.81
orthoMCL	116,698	15,652	619,957,300	17,300	0.88	0.87	0.87
OMA	91,762	1,167	619,971,785	42,236	0.99	0.68	0.81

Table 23: Fmeasures of protein clustering techniques with respect to RBH cliques. True positives, false positives, true negatives, and false negatives are shown in the columns labeled *TP*, *FP*, *TN*, and *FN*, respectively, with respect to RBH cliques.

- False Negative (FN) if they are in the same clique but are not clustered together in the same cluster.

Table 23 shows the F-measures of the techniques with respect to RBH cliques. The table shows the techniques under the column labeled *Technique*. For each system, the table shows the true positives, false positives, true negatives, and false negatives in the columns labeled *TP*, *FP*, *TN*, and *FN*, respectively. The TPs, FPs, TNs, and FNs are as defined above with respect to RBH cliques. The precision, recall, and F-measure are defined in Definition 2.3. Consider the systems MCL, SynAPhy_1_5, and SynAPhy_3_5. The precisions of MCL, SynAPhy_1_5, and SynAPhy_3_5 are 0.54, 0.63, and 0.71, respectively. The precision of SynAPhy_1_5 is higher than that of MCL, and the precision of SynAPhy_3_5 is higher than that of SynAPhy_1_5. This is expected as SynAPhy_1_5 scales the similarity values of syntenic RBHs, which are RBHs, and SynAPhy_3_5 scales the similarity values of all RBHs. This behavior allows SynAPhy_1_5 and SynAPhy_3_5 to cluster more RBHs together. The recall of MCL, SynAPhy_1_5, and SynAPhy_3_5 are 0.94, 0.94, and 0.95, respectively. The recalls of the three techniques are comparable and satisfactory. The F-measures of MCL, SynAPhy_1_5, and SynAPhy_3_5 are 0.69, 0.75, and 0.81, respectively. The F-measure of SynAPhy_3_5 is better than the other two systems.

Consider orthoMCL and OMA. OMA has a better precision at the expense of recall. OMA has a very higher precision value 0.99. This means almost all the RBHs are clustered together. orthoMCL has a precision value 0.88. The precision of orthoMCL is higher than that of SynAPhy_3_5 but lower than that of OMA. The recalls of orthoMCL and OMA are 0.87 and 0.68, respectively. orthoMCL has a better recall than OMA, and in turn SynAPhy_3_5 has a higher recall than orthoMCL.

orthoMCL has the highest F-measure among the five techniques with a value of 0.87. OMA and SynAPhy_3_5 have similar F-measure with a value of 0.81. SynAPhy_1_5 and MCL follow them with values 0.75 and 0.69. Note that MCL is not an orthology prediction system. By scaling the similarity values of the RBHs, we are directing MCL to cluster RBHs together. Therefore, SynAPhy_1_5 and SynAPhy_3_5 have better precision than that of MCL.

Table 23 shows that OMA is leading in precision at the expense of recall. orthoMCL is leading in F-measure. MCL, SynAPhy_1_5, and SynAPhy_3_5 have higher recall than both orthoMCL and OMA.

Table 24 shows the F-measures of MCL, SynAPhy_1_5, SynAPhy_3_5, orthoMCL, and OMA techniques with respect to syntenic RBH cliques. The sequence of results are similar to that of RBH

Technique	TP	FP	TN	FN	Precision	Recall	F-measure
MCL	26,298	325,832	1,603,554,203	129	0.07	1.0	0.13
SynAPhy_1_5	26,302	217,829	1,603,662,206	125	0.11	1.0	0.2
SynAPhy_3_5	26,237	127,239	1,603,752,796	190	0.17	0.99	0.29
orthoMCL	25,667	72,515	1,603,807,520	760	0.26	0.97	0.41
OMA	24,300	31,952	1,603,848,083	2,127	0.43	0.92	0.59

Table 24: Fmeasures of protein clustering techniques with respect to syntenic RBH cliques. True positives, false positives, true negatives, and false negatives are shown in the columns labeled *TP*, *FP*, *TN*, and *FN*, respectively, with respect to syntenic RBH cliques.

cliques presented in Table 23. Consider MCL, SynAPhy_1_5, and SynAPhy_3_5. SynAPhy_3_5 has a higher precision than that of SynAPhy_1_5, and in turn SynAPhy_3_5 has a higher precision than that of MCL. The recalls of the three techniques are similar. Both MCL and SynAPhy_1_5 have a recall value of 1, which means all the syntenic RBHs are clustered. SynAPhy_3_5 has a recall value of 0.99. The F-measure of SynAPhy_3_5 is the highest, with a value of 0.29, while that of SynAPhy_1_5 is 0.2 and MCL 0.13.

Consider orthoMCL and OMA. Similar to the results of RBH cliques, OMA has a higher precision at the expense of recall. OMA has a better F-measure than the rest of the techniques for syntenic RBH cliques. Similar to the conclusion presented for the RBH cliques, precision of OMA is higher at the expense of recall.

The analysis of Table 23 and Table 24 would not have been possible without the orthology resolver component of SynAVal. We showed the prospect of RBHs and syntenic RBHs to be functionally similar with the mycoCLAP gold standard dataset. Although the numbers of RBHs and syntenic RBHs present in the mycoCLAP dataset are low, all the present cases have similar functional annotation presented by the mycoCLAP dataset.

5.4 Conclusion

To evaluate an orthologous cluster, the orthology of each and every pair in the cluster has to be confirmed. In the presence of highly similar paralogs, the evaluation of orthology and paralogy is a hard problem. With the absence of gold standard genome scale dataset, the problem of evaluation becomes even harder. In this chapter, we present SynAVal, an evaluation framework for an orthology prediction system that first detects the paralogs of each input genome, and then detects the high confidence orthologs. It uses these data to identify and report the confusions in orthologous clusters generated by an orthology prediction system.

The paralogs are detected with the gene copy detector. Approximately 25% to 40% of the input proteins correspond to multi-copy genes. This range is significant, it shows that a large percentage of the input proteomes may be subject to orthology and paralogy confusion.

The clique detector builds high confidence orthologous relations. If RBH resolution is required, the orthology of 39.42% of the proteins in the eight fungal genomes may be resolved, and if the syntenic RBH resolution is required the orthology of 9.1% of the proteins in the eight fungal genomes

may be resolved. These results are based on both single- and multi-copy genes that are in multi-copy RBH cliques. Moreover, if the background data is with respect to the multi-copy genes that have hits in other genomes, SynAVal with RBH resolution is able to resolve confusions raised by 78.65% of multi-copy genes with hits in other proteomes, and with syntenic RBH cliques 23.33% of multi-copy genes with hits in other proteomes.

SynAVal detects and reports output orthologous clusters that contain multi-copy RBH or syntenic RBH cliques together with the paralogs of the multi-copy gene(s). These are the clusters that are flagged as confusion. SynAVal not only reports high confidence orthology dataset, but also evaluates the results of orthology prediction system and reports those clusters with confusion.

We applied SynAVal on MCL, SynAPhy_1_5, SynAPhy_3_5, orthoMCL, and OMA, and compared the results. MCL, SynAPhy_1_5, and SynAPhy_3_5 are compared against each other because they consider all the input proteomes. orthoMCL and OMA filter out a large number of the input proteins. Note that MCL is not an orthology prediction system, we present its results to measure the performance of SynAPhy_1_5 and SynAPhy_3_5 over the standard pairwise similarity scores. OMA does not cluster paralogs, therefore, its results are not included in comparisons where evaluation of clusters are conducted with respect to the presence of paralogs.

SynAVal allowed us to evaluated the correctness of the orthology relations in the generated clusters for MCL, SynAPhy_1_5, SynAPhy_3_5, orthoMCL, and OMA. With both RBH and syntenic RBH cliques, SynAPhy_3_5 has higher F-measure than that of SynAPhy_1_5, and SynAPhy_1_5 has a higher F-measure than that of MCL. With RBH cliques, orthoMCL has higher F-measure than that of OMA, MCL, SynAPhy_1_5, and SynAPhy_3_5. With syntenic RBH cliques OMA has the highest. We observed that the precisions of both orthoMCL and OMA in RBH and syntenic RBH cliques are higher than that of SynAPhy_3_5 at the expense of recall.

The RBH and syntenic RBH connections proved consistent with the functional annotations of the small set of experimentally characterized proteins in the mycoCLAP dataset. Although syntenic RBHs are less frequent than RBHs, they provide high confidence orthology relations for 23.33% of the multi-copy genes with hits in other proteomes that are highly likely to raise orthology and paralogy confusions.

Chapter 6

Conclusion and Future Work

Accurate protein functional annotation is important in order to understand the roles of proteins in cells as well as their application in different fields. As the number of sequenced genomes is increasing, the need for accurate automated protein functional annotation is becoming more pressing. Phylogenomic methods consider the orthology and paralogy resolution in determining evolutionary conserved proteins towards accurate protein function annotation. However, a long-standing problem in phylogenomics is distinguishing orthologs from paralogs. In this dissertation, we presented a novel graph-based algorithm to distinguish orthologs from paralogs in multiple eukaryotic species.

We first presented an evaluation of CAZyme family HMMs and subfamily HMMs. The results of the family HMMs were satisfactory to be used in a functional annotation pipeline targeted for CAZymes. We showed, however, that care must be taken when setting inclusion thresholds for HMMs. The results of the subfamily HMMs were inconclusive as there was not enough diversity in the underlying data to carry out exhaustive experiments and draw strong conclusions. We also presented a study of the effectiveness of the MCL algorithm for clustering protein families and subfamilies. The MCL algorithm was able to accurately separate protein families of an experimentally characterized dataset, however, it did not perform similarly for subfamilies. We showed that cluster metrics can be used to mark good quality clusters and outliers.

We then presented SynAPhy, a novel graph-based algorithm to predict orthologs that integrates synteny resolution. SynAPhy detects RBHs and syntenic RBHs and builds a protein sequence similarity graph from the results of all-versus-all Blast. It then scales the similarity values of RBHs and syntenic RBHs with different scalars and applies the Markov CLustering algorithm (MCL) in the resulting protein sequence similarity graph. SynAPhy was applied on eight fungal genomes and the results were compared to that of OrthoMCL and OMA. The evaluation was conducted with four cluster quality metrics. SynAPhy shows improvement when MCL is applied on the original protein sequence similarity graph. OMA outperformed in most of the metrics. It is important to note that OMA does not cluster more than one protein sequence from a specific genome in a cluster. This filtering contributes positively in the cluster quality metrics.

Although the cluster quality metrics evaluate cluster qualities in terms of sequence similarity within clusters, they are not indicators of orthology and paralogy, i.e., they do not evaluate whether

every pair within an orthologous cluster is ortholog. We proposed SynAVal, an evaluation framework for orthology and paralogy. SynAVal first detects the paralogs within each genome, and then detects high confidence orthologs. The high confidence orthologs are the RBH and syntenic RBH cliques. The results show that with synteny resolution SynAVal can resolve confusions raised by 9.1% of the proteins of the eight input genomes, and 23.33% of the proteins from the eight genomes that can raise orthology and paralogy confusions.

SynAPhy considers two signals for pairwise proteins, sequence similarity and gene neighborhood conservation. Several additional biological signals can be used to extend SynAPhy as new data emerges. For example, conformational features extracted from protein-protein interaction networks and weighted gene coexpression networks. It is expected that interactions of orthologs are conserved, and that there is a positive gene coexpression correlation between orthologs and their gene neighbors in gene coexpression networks of different genomes [TVO$^+$10]. Another signal is the pattern of presence and absence of genes in different genomes (gene phylogenetic profiles). The idea is that orthologs have similar patterns of presence and absence in different genomes [EW02]. This correlation can be integrated into the list of pairwise protein sequence features. These features, separate and combined are another avenue to consider in distinguishing orthologs from paralogs.

SynAPhy treats RBHs as candidate orthologous seeds, and then examines whether their corresponding genes are in gene neighborhoods with a predefined number of RBHs. This approach is a bottom up approach for computing syntenic blocks. Some orthology prediction systems compute syntenic blocks at the DNA level. SynAPhy can be extended to cover a top down approach in which syntenic blocks are computed at the DNA level. This approach allows one to examine the blocks in SynAPhy in which RBHs of interest have less than the required number of neighboring RBHs for the blocks to be pronounced syntenic blocks. This analysis can lead to the detection of gene model errors, where homologous genes that do not correspond to RBHs can be extended to check whether they can form RBHs or not. In addition, syntenic blocks at DNA level allows examining potential similarities in the non-protein coding regions preceding the syntenic blocks.

Bibliography

[ABGW94] Stephen F. Altschul, Mark S. Boguski, Warren Gish, and John C. Wootton. Issues in searching molecular sequence databases. *Nature Genetics*, 6(2):119–129, 1994.

[ACW+12] Henrik Aspeborg, Pedro M. Coutinho, Yang Wang, Harry Brumer, and Bernard Henrissat. Evolution, substrate specificity and subfamily classification of glycoside hydrolase family 5 (GH5). *BMC Evolutionary Biology*, 12(1):1–16, 2012.

[AD09] Adrian M Altenhoff and Christophe Dessimoz. Phylogenetic and functional assessment of orthologs inference projects and methods. *PLoS Comput Biology*, 5(1):e1000262, 2009.

[AD12] Adrian M Altenhoff and Christophe Dessimoz. Inferring orthology and paralogy. *Methods in Molecular Biology*, 855:259–279, 2012.

[AG96] Stephen F. Altschul and Warren Gish. Local alignment statistics. *Methods in Enzymology*, 266:460–480, 1996.

[AGM+90] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular biology*, 215(3):403–410, 1990.

[AMS+97] Stephen F. Altschul, Thomas L. Madden, Alejandro A. Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, 1997.

[Anf73] Christian B. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(4096):223–230, 1973.

[ARC+54] Christian B. Anfinsen, Robert R. Redfield, Warren L. Choate, Juanita Page, and William R. Carroll. Studies on the gross structure, cross-linkages, and terminal sequences in ribonuclease. *Journal of Biological Chemistry*, 207(1):201–210, 1954.

[Bak75] James Baker. The DRAGON system – an overview. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 23(1):24–29, 1975.

[BCC+13] Dennis A. Benson, Mark Cavanaugh, Karen Clark, Ilene Karsch-Mizrachi, David J. Lipman, James Ostell, and Eric W. Sayers. GenBank. *Nucleic Acids Research*, 41(D1):D36–D42, 2013.

[BK73] Coen Bron and Joep Kerbosch. Algorithm 457: Finding all cliques of an undirected graph. *Commun. ACM*, 16(9):575–577, 1973.

[But13] Greg Butler. Putting it all together: The design of a pipeline for genome-wide functional annotation of fungi in the modern era of "omics" data and systems biology. *Data Integration in the Life Sciences*, pages 113–127, 2013.

[CBBWT61] Francis H. C. Crick, Leslie Barnett, Sydney Brenner, and R. J. Watts-Tobin. General nature of the genetic code for proteins. *Nature*, 192:1227–1232, 1961.

[CCR+09] Brandi L. Cantarel, Pedro M. Coutinho, Corinne Rancurel, Thomas Bernard, Vincent Lombard, and Bernard Henrissat. The Carbohydrate-Active EnZymes database (CAZy): an expert resource for Glycogenomics. *Nucleic Acids Research*, 37(suppl 1):D233–D238, 2009.

[Coo06] Orator F. Cook. Factors of species-formation. *Science*, 23(587):506–507, 1906.

[Coo08] Orator F. Cook. Evolution without isolation. *American Naturalist*, 42(503):727–731, 1908.

[Cri58] Francis H. C. Crick. On protein synthesis. *Symposia of the Society for Experimental Biology*, (12):139–163, 1958.

[CY03] Steven B. Cannon and Nevin D. Young. OrthoParaMap: Distinguishing orthologs from paralogs by integrating comparative genome data and gene phylogenies. *BMC Bioinformatics*, 4(1):1–15, 2003.

[Day74] Margaret O. Dayhoff. Computer analysis of protein sequences. *Federation Proceedings*, 33(12):2314–2316, 1974.

[Day76] Margaret O. Dayhoff. The origin and evolution of protein superfamilies. *Federation proceedings*, 35(10):2132–2138, 1976.

[DGR+12] Christophe Dessimoz, Toni Gabaldón, David S. Roos, Erik L. L. Sonnhammer, Javier Herrero, and the Quest for Orthologs Consortium. Toward community standards in the quest for orthologs. *Bioinformatics*, 28(6):900–904, 2012.

[DMBH75] Margaret O. Dayhoff, Patricia J. McLaughlin, Winona C. Barker, and Lois T. Hunt. Evolution of sequences within protein superfamilies. *Naturwissenschaften*, 62(4):154–161, 1975.

[DSO78] Margaret O. Dayhoff, R. M. Schwartz, and B. C. Orcutt. A model of evolutionary change in proteins. *Atlas of Protein Sequence and Structure*, 5(Suppl 3):345–352, 1978.

[Edd98] Sean R. Eddy. Profile hidden Markov models. *Bioinformatics*, 14(9):755–763, 1998.

[Eis98] Jonathan A. Eisen. Phylogenomics: Improving functional predictions for uncharacterized genes by evolutionary analysis. *Genome Research*, 8(3):163–167, 1998.

[ESN97] Jason Ehrlich, David Sankoff, and Joseph H. Nadeau. Synteny conservation and chromosome rearrangements during mammalian evolution. *Genetics*, 147(1):289–296, 1997.

[EW02] Jonathan A. Eisen and Martin Wu. Phylogenetic analysis and gene functional predictions: Phylogenomics in action. *Theoretical Population Biology*, 61(4):481–487, 2002.

[FAW+95] Robert D. Fleischmann, Mark D. Adams, Owen White, Rebecca A. Clayton, Ewen F. Kirkness, Anthony R. Kerlavage, Carol J. Bult, Jean-Francois Tomb, Brian A. Dougherty, Joseph M. Merrick, Keith McKenney, Granger Sutton, Will Fitzhugh, Chris Fields, Jeannie D. Gocyne, John Scott, Robert Shirley, Li-Ing Liu, Anna Glodek, Jenny M. Kelley, Janice F. Weidman, Cheryl A. Phillips, Tracy Spriggs, Eva Hedblom, Matthew D. Cotton, Teresa R. Utterback, Michael C. Hanna, David T. Nguyen, Deborah M. Saudek, Rhonda C. Brandon, Leah D. Fine, Janice L. Fritchman, Joyce L. Fuhrmann, N. S. M. Geoghagen, Cheryl L. Gnehm, Lisa A. McDonald, Keith V. Small, Claire M. Fraser, Hamilton O. Smith, and J. Craig Venter. Whole-genome random sequencing and assembly of Haemophilus influenzae Rd. *Science*, 269(5223):496–512, 1995.

[FBB+12] Derrick E. Fouts, Lauren Brinkac, Erin Beck, Jason Inman, and Granger Sutton. PanOCT: automated clustering of orthologs using conserved gene neighborhood for pan-genomic analysis of bacterial strains and closely related species. *Nucleic Acids Research*, 40(22):e172, 2012.

[Fit70] Walter M. Fitch. Distinguishing homologous from analogous proteins. *Systematic Biology*, 19(2):99–113, 1970.

[Fit00] Walter M. Fitch. Homology: a personal view on some of the problems. *Trends in Genetics*, 16(5):227–231, 2000.

[GP06] Leo Goodstadt and Chris P. Ponting. Phylogenetic reconstruction of orthology, paralogy, and conserved synteny for dog and human. *PLoS Computational Biology*, 2(9):e133, 2006.

[HCDDG07] Jaime Huerta-Cepas, Hernan Dopazo, Joaquin Dopazo, and Toni Gabaldón. The human phylome. *Genome Biology*, 8(6):R109, 2007.

[HH92] Steven Henikoff and Jorja G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

[HHdVG06] Tim Hulsen, Martijn A Huynen, Jacob de Vlieg, and Peter M A Groenen. Benchmarking ortholog identification methods using functional genomics data. *Genome Biology*, 7(4):R31, 2006.

[HHM08] John Harris, Jeffry L. Hirst, and Michael Mossinghoff. *Combinatorics and Graph Theory*. Springer-Verlag, New York, 2008.

[HT73] John Hopcroft and Robert Tarjan. Algorithm 447: Efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378, 1973.

[JTT92] David T. Jones, William R. Taylor, and Janet M. Thornton. The rapid generation of mutation data matrices from protein sequences. *Computer applications in the biosciences*, 8(3):275–282, 1992.

[KBKS06] Nandini Krishnamurthy, Duncan P. Brown, Dan Kirshner, and Kimmen Sjölander. PhyloFacts: an online structural phylogenomic encyclopedia for protein functional and structural classification. *Genome Biology*, 7(9):1–17, 2006.

[KBM+94] Anders Krogh, Michael Brown, I. Saira Mian, Kimmen Sjölander, and David Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235(5):1501–1531, 1994.

[LAB+11] Rasko Leinonen, Ruth Akhtar, Ewan Birney, Lawrence Bower, Ana Cerdeno-Tárraga, Ying Cheng, Iain Cleland, Nadeem Faruque, Neil Goodgame, Richard Gibson, Gemma Hoad, Mikyung Jang, Nima Pakseresht, Sheila Plaister, Rajesh Radhakrishnan, Kethi Reddy, Siamak Sobhany, Petra Ten Hoopen, Robert Vaughan, Vadim Zalunin, and Guy Cochrane. The European Nucleotide Archive. *Nucleic Acids Research*, 39(suppl 1):D28–D31, 2011.

[LCR+06] Heng Li, Avril Coghlan, Jue Ruan, Lachlan James Coin, Jean-Karim Heriche, Lara Osmotherly, Ruiqiang Li, Tao Liu, Zhang Zhang, Lars Bolund, Gane Ka-Shu Wong, Weimou Zheng, Paramvir Dehal, Jun Wang, and Richard Durbin. TreeFam: a curated database of phylogenetic trees of animal gene families. *Nucleic Acids Research*, 34:D572–D580, 2006.

[LSR03] Li Li, Christian J. Stoeckert, and David S. Roos. OrthoMCL: Identification of ortholog groups for eukaryotic genomes. *Genome Research*, 13(9):2178–2189, 2003.

[MBZC+13] Aron Marchler-Bauer, Chanjuan Zheng, Farideh Chitsaz, Myra K. Derbyshire, Lewis Y. Geer, Renata C. Geer, Noreen R. Gonzales, Marc Gwadz, David I. Hurwitz, Christopher J. Lanczycki, Fu Lu, Shennan Lu, Gabriele H. Marchler, James S. Song, Narmada Thanki, Roxanne A. Yamashita, Dachuan Zhang, and Stephen H. Bryant. CDD: conserved domains and protein three-dimensional structure. *Nucleic Acids Research*, 41(D1):D348–D352, 2013.

[MPW+11] Caitlin Murphy, Justin Powlowski, Min Wu, Greg Butler, and Adrian Tsang. Curation of characterized glycoside hydrolases of fungal origin. *Database*, 2011, 2011.

[NT84] Joseph H. Nadeau and B. A. Taylor. Lengths of chromosomal segments conserved since divergence of man and mouse. *Proceedings of the National Academy of Sciences USA*, 81:814–818, 1984.

[NW70] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

[Ohn70] Susumu Ohno. *Evolution by Gene Duplication*. Springer, New York, 1970.

[OMK+13] Osamu Ogasawara, Jun Mashima, Yuichi Kodama, Eli Kaminuma, Yasukazu Nakamura, Kousaku Okubo, and Toshihisa Takagi. DDBJ new system and service refactoring. *Nucleic Acids Research*, 41(D1):D25–D29, 2013.

[ÖSF+10] Gabriel Östlund, Thomas Schmitt, Kristoffer Forslund, Tina Köstler, David N. Messina, Sanjit Roopra, Oliver Frings, and Erik L. L. Sonnhammer. InParanoid 7: new algorithms and tools for eukaryotic orthology analysis. *Nucleic Acids Research*, 38(suppl 1):D196–D203, 2010.

[PCE+12] Marco Punta, Penny C. Coggill, Ruth Y. Eberhardt, Jaina Mistry, John Tate, Chris Boursnell, Ningze Pang, Kristoffer Forslund, Goran Ceric, Jody Clements, Andreas Heger, Liisa Holm, Erik L. L. Sonnhammer, Sean R. Eddy, Alex Bateman, and Robert D. Finn. The Pfam protein families database. *Nucleic Acids Research*, 40(D1):D290–D301, 2012.

[PFS+13] Sean Powell, Kristoffer Forslund, Damian Szklarczyk, Kalliopi Trachana, Alexander Roth, Jaime Huerta-Cepas, Toni Gabaldón, Thomas Rattei, Chris Creevey, Michael Kuhn, Lars J. Jensen, Christian von Mering, and Peer Bork. eggNOG v4.0: nested orthology inference across 3686 organisms. *Nucleic Acids Research*, 2013.

[PJ01] Marco Pagni and C. Victor Jongeneel. Making sense of score statistics for sequence alignments. *Briefings in Bioinformatics*, 2(1):51–67, 2001.

[PST+12] Sean Powell, Damian Szklarczyk, Kalliopi Trachana, Alexander Roth, Michael Kuhn, Jean Muller, Roland Arnold, Thomas Rattei, Ivica Letunic, Tobias Doerks, Lars J. Jensen, Christian von Mering, and Peer Bork. eggNOG v3.0: orthologous groups covering 1133 organisms at 41 different taxonomic ranges. *Nucleic Acids Research*, 40(D1):D284–D289, 2012.

[RGD08] Alexander C. J. Roth, Gaston H. Gonnet, and Christophe Dessimoz. Algorithm of OMA for large-scale orthology inference. *BMC Bioinformatics*, 9(1):1–10, 2008.

[SK02] Erik L.L Sonnhammer and Eugene V Koonin. Orthology, paralogy and proposed classification for paralog subtypes. *Trends in Genetics*, 18(12):619–620, 2002.

[SN96] David Sankoff and Joseph H. Nadeau. Conserved synteny as a measure of genomic distance. *Discrete Applied Mathematics*, 71(13):247–257, 1996.

[Str60] Ruslan L. Stratonovich. Conditional Markov processes. *Theory of Probability and its Application*, 5:156–178, 1960.

[SW81] Temple F. Smith and Michael S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981.

[TFJ+03] Roman Tatusov, Natalie Fedorova, John Jackson, Aviva Jacobs, Boris Kiryutin, Eugene Koonin, Dmitri Krylov, Raja Mazumder, Sergei Mekhedov, Anastasia Nikolskaya, B Sridhar Rao, Sergei Smirnov, Alexander Sverdlov, Sona Vasudevan, Yuri Wolf, Jodie Yin, and Darren Natale. The COG database: an updated version includes eukaryotes. *BMC Bioinformatics*, 4(1):41, 2003.

[TKL97] Roman L. Tatusov, Eugene V. Koonin, and David J. Lipman. A genomic perspective on protein families. *Science*, 278(5338):631–637, 1997.

[TVO+10] Fadi Towfic, Susan VanderPlas, Casey A. Oliver, Oliver Couture, Christopher K. Tuggle, M. Heather West Greenlee, and Vasant Honavar. Detection of gene orthology from gene co-expression and protein interaction networks. *BMC Bioinformatics*, 11(S-3):7, 2010.

[VD00a] Stijn Van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, The Netherlands, 2000.

[VD00b] Stijn Van Dongen. Performance criteria for graph clustering and Markov cluster experiments. Technical report, National Research Institute for Mathematics and Computer Science, Amsterdam, The Netherlands, 2000.

[vdHRSvNH07] van der Heijden RT, Berend Snel, Vera van Noort, and Martijn A Huynen. Orthology prediction at scalable resolution by phylogenetic tree analysis. *BMC Bioinformatics*, 8(8):83, 2007.

[VSUV+09] Albert J Vilella, Jessica Severin, Abel Ureta-Vidal, Li Heng, Richard Durbin, and Ewan Birney. EnsemblCompara GeneTrees: Complete, duplication-aware phylogenetic trees in vertebrates. *Genome Research*, 19(2):327–335, 2009.

[WP99] Todd C. Wood and William R. Pearson. Evolution of protein sequences and structures. *Journal of Molecular Biology*, 291(4):977–995, 1999.

[WPFR07] Ilan Wapinski, Avi Pfeffer, Nir Friedman, and Aviv Regev. Automatic genome-wide reconstruction of phylogenetic gene trees. *Bioinformatics*, 23(13):i549–i558, 2007.

[YMY+12] Yanbin Yin, Xizeng Mao, Jincai Yang, Xin Chen, Fenglou Mao, and Ying Xu. dbCAN: a web resource for automated carbohydrate-active enzyme annotation. *Nucleic Acids Research*, 40(W1):W445–W451, 2012.

[ZE01] Christian M. Zmasek and Sean R. Eddy. A simple algorithm to infer gene duplication and speciation events on a gene tree. *Bioinformatics*, 17(9):821–828, 2001.

[ZE02] Christian M. Zmasek and Sean R. Eddy. RIO: Analyzing proteomes by automated phylogenomics using resampled inference of orthologs. *BMC Bioinformatics*, 3(1):1–19, 2002.

Appendix A

Enzyme Family and Subfamily Clustering

A.1 mycoCLAP Dataset

Table 25 presents a summary of the CAZymes in the mycoCLAP database as of September 2013. It shows the CAZyme families, the number of sequences in each family, the molecular functions in each family, and the number of sequences with a specific molecular function in each family. Note that there are five bifunctional CAZymes denoted as GH11_CE1 and GH32_GH43.

Family	Frequency	Enzymatic activity	Frequency
GH16	10	endo-beta-1,3-galactanase	1
		laminarinase	1
		licheninase	2
		mixed-link glucanase	6
GH17	2	laminarinase	1
		exo-1,3-beta-glucanase	1
GH15	20	glucoamylase	20
GH12	32	licheninase	1
		xyloglucanase	3
		endoglucanase	28
GH13	24	oligo-1,6-glucosidase	3
		alpha-glucosidase	4
		alpha-amylase	17
PL1	4	pectate lyase	2
		pectin lyase	2
GH11	68	xylanase	68
CE12	1	rhamnogalacturonan acetylesterase	1
GH18	34	endo-beta-n-acetylglucosaminidase	1
		chitinase	33
GH75	8	chitosanase	8
GH51	9	alpha-arabinofuranosidase	9

GH74	6	oligoxyloglucan cellobiohydrolase	2
		xyloglucanase	3
		endoglucanase	1
GH71	5	mutanase	5
GH54	13	alpha-arabinofuranosidase/beta-xylosidase	1
		alpha-arabinofuranosidase	12
GH55	10	laminarinase	3
		exo-1,3-beta-glucanase	7
GH30	6	endo-1,6-beta-glucanase	4
		galactanase	1
		xylanase	1
GH33	1	exo-alpha-sialidase	1
GH78	4	alpha-l-rhamnosidase	4
GH79	1	beta-glucuronidase	1
GH36	8	alpha-galactosidase	8
GH53	6	arabinogalactanase	6
GH61	4	polysaccharide monooxygenase	4
GH31	14	invertase	1
		alpha-xylosidase	1
		alpha-glucosidase	11
		glucoamylase	1
GH32	17	exo-inulinase	4
		invertase	10
		endo-inulinase	3
GH10	28	tomatinase	1
		xylanase	27
GH11_CE1	1	xylanase/acetylxylan esterase	1
GH65	2	trehalase	2
GH35	3	beta-galactosidase	3
GH81	3	laminarinase	3
GH85	1	n-acetylglucosaminidase	1
GH32_GH43	4	exo-inulinase	2
		endo-inulinase	2
PL4	2	rhamnogalacturonan lyase	2
GH67	6	alpha-glucuronidase	5
		xylan alpha-1,2-glucuronosidase	1
CE6	1	acetylxylan esterase	1
CE5	5	acetylxylan esterase	2
		cutinase	3
CE4	4	acetylxylan esterase	1
		chitin deacetylase	3
GH49	4	dextranase	3
		isopullulanase	1
GH62	5	arabinoxylan arabinofuranosidase	2
		arabinoxylan arabinofuranohydrolase	3
CE1	12	acetylxylan esterase	4
		feruloyl esterase	8
GH45	16	endoglucanase	16
GH47	6	alpha-1,2-mannosidase	6

GH20	12	hexosaminidase	12
GH27	14	alpha-galactosidase	14
GH26	4	beta-mannanase	4
GH43	14	exo-beta-1,3-galactanase	1
		beta-xylosidase	3
		alpha-arabinofuranosidase/beta-xylosidase	1
		exo-1,3-beta-galactanase	1
		arabinoxylan arabinofuranohydrolase	2
		endo-1,5-alpha-arabinanase	6
CE8	1	pectin methylesterase	1
GH5	71	endo-1,6-beta-glucanase	3
		exo-1,3-beta-glucanase	14
		endoglucanase	38
		galactanase	1
		endoglucanase/xylanase	1
		beta-mannanase	14
GH1	8	beta-glucosidase	8
GH6		cellobiohydrolase	16
		endoglucanase	1
GH7	17	cellobiohydrolase	25
	38	mixed-link glucanase	1
		endoglucanase	11
		xylanase	1
GH2	5	exo-glucosaminidase	2
		beta-galactosidase	1
		beta-mannosidase	2
GH3	40	beta-glucosidase	28
		tomatinase	1
		beta-xylosidase	8
		avenacinase	3
GH9	1	endoglucanase	1
AA5	1	glyoxal oxidase	1
GH28	67	rhamnogalacturonan hydrolase	4
		alpha-l-rhamnosidase	1
		exo-polygalacturonase	11
		endo-polygalacturonase	50
		xylogalacturonase	1
AA3	7	aryl-alcohol oxidase	2
		pyranose 2-oxidase	1
		glucose oxidase	4
AA2	20	lignin peroxidase	9
		peroxidase	2
		manganese peroxidase	6
		versatile peroxidase	3
GH115	1	xylan alpha-1,2-glucuronidase	1
GH93	2	exo-arabinanase	2
PL3	3	pectate lyase	3

Table 25: mycoCLAP database summary

A.2 Confusion Matrices of CAZyme Families

Table 26 shows the confusion matrices of all CAZyme families in CAZyDB when members are searched against the dbCAN family HMMs. The count of CAZymes classified as TP, FN, FP, and TN are shown in their respective columns. The actual families and the corresponding count of misclassified CAZymes are shown in the column labeled *Confusion*. The rows are sorted in descending order of F-measure and then sorted in descending order of TP. GH families 36 and 39, and AA family 1 have the lowest F-measure, which is the result of the unclassified sequences.

Family	TP	FN	FP	TN	F-measure	Confusion
CE4	144	0	0	7937	1	-
GH132	103	0	0	7978	1	-
CE5	86	0	0	7995	1	-
AA6	74	0	0	8007	1	-
GH128	49	0	0	8032	1	-
GH125	38	0	0	8043	1	-
CE8	35	0	0	8046	1	-
CE3	30	0	0	8051	1	-
PL4	30	0	0	8051	1	-
GH75	30	0	0	8051	1	-
CE12	27	0	0	8054	1	-
GH105	25	0	0	8056	1	-
GH93	25	0	0	8056	1	-
CE16	25	0	0	8056	1	-
GH30	24	0	0	8057	1	-
CE9	23	0	0	8058	1	-
GH62	22	0	0	8059	1	-
GH114	20	0	0	8061	1	-
GH88	18	0	0	8063	1	-
GH53	18	0	0	8063	1	-
AA7	17	0	0	8064	1	-
GH67	15	0	0	8066	1	-
GH131	13	0	0	8068	1	-
GH95	12	0	0	8069	1	-
CE15	12	0	0	8069	1	-
GH64	11	0	0	8070	1	-
GH74	10	0	0	8071	1	-

GH33	9	0	0	8072	1	-
GH49	9	0	0	8072	1	-
GH9	8	0	0	8073	1	-
GH85	8	0	0	8073	1	-
PL14	8	0	0	8073	1	-
GH130	7	0	0	8074	1	-
GH29	6	0	0	8075	1	-
GH25	6	0	0	8075	1	-
GH24	6	0	0	8075	1	-
AA4	5	0	0	8076	1	-
GH23	4	0	0	8077	1	-
GH19	4	0	0	8077	1	-
PL20	4	0	0	8077	1	-
CE2	4	0	0	8077	1	-
PL9	4	0	0	8077	1	-
GH127	4	0	0	8077	1	-
PL7	3	0	0	8078	1	-
CE6	2	0	0	8079	1	-
GH48	2	0	0	8079	1	-
GH42	2	0	0	8079	1	-
GH8	1	0	0	8080	1	-
GH89	1	0	0	8080	1	-
GH84	1	0	0	8080	1	-
AA9	178	1	0	7902	0.9972	-
GH76	169	1	0	7911	0.9971	-
GH11	170	2	0	7909	0.9942	-
GH47	161	2	0	7918	0.9938	-
GH37	77	1	0	8003	0.9936	-
GH72	206	3	0	7872	0.9928	-
GH12	68	1	0	8012	0.9927	-
GH31	133	2	0	7946	0.9925	-
GH16	361	6	0	7714	0.9918	-
GH81	59	1	0	8021	0.9916	-
GH10	110	2	0	7969	0.991	-
GH20	53	1	0	8027	0.9907	-
GH1	49	1	0	8031	0.9899	-
PL3	48	1	0	8032	0.9897	-
GH92	47	1	0	8033	0.9895	-

GH6	84	2	0	7995	0.9882	-
GH18	745	18	0	7318	0.9881	-
GH38	39	1	0	8041	0.9873	-
GH43	177	5	0	7899	0.9861	-
GH65	35	1	0	8045	0.9859	-
GH51	34	1	0	8046	0.9855	-
AA5	34	1	0	8046	0.9855	-
AA3	125	4	0	7952	0.9843	-
GH55	62	2	0	8017	0.9841	-
GH71	60	2	0	8019	0.9836	-
GH78	57	2	0	8022	0.9828	-
GH2	83	2	1	7995	0.9823	-
GH5	450	17	0	7614	0.9815	-
GH54	25	1	0	8055	0.9804	-
GH115	22	1	0	8058	0.9778	-
GH79	41	2	0	8038	0.9762	-
GH35	40	2	0	8039	0.9756	-
GH63	40	2	0	8039	0.9756	-
GH17	159	10	0	7912	0.9695	-
GH3	298	19	0	7764	0.9691	-
GH28	384	27	0	7670	0.966	-
GH32	149	11	0	7921	0.9644	GH2 : 1
CE1	54	3	1	8023	0.9643	-
GH27	77	6	0	7998	0.9625	-
PL1	115	10	0	7956	0.9583	-
GH7	333	30	0	7718	0.9569	-
GH45	31	3	0	8047	0.9539	-
GH13	266	28	0	7787	0.95	-
AA2	227	30	0	7824	0.938	-
GH26	11	2	0	8068	0.9167	CE1 : 1
GH15	97	20	0	7964	0.9065	-
CE10	4	1	0	8076	0.8889	-
GH94	3	1	0	8077	0.8571	-
GH36	17	13	0	8051	0.7234	-
GH39	3	8	0	8070	0.4286	-
AA1	94	359	0	7628	0.3437	-

Table 26: Confusion matrices of CAZyDB protein families with respect to the dbCAN family HMMs.

A.3 The Quality of MCL Clusters on *mleci* Graph

Table 27 shows the cluster qualities of all the MCL clusters of the *mleci* graph. The intra-cluster density is given by the column labeled δ. The number of vertices in the cluster is given in the column labeled *# vertices*. The minimum and maximum similarities are given in the columns labeled by sim_{min} and sim_{max}, respectively. The average of the normalized edge weights is given by the column labeled $\overline{w}$, and the coefficient of variation of average normalized edge weights is given in the column labeled c_v. The rows are sorted in decreasing order of $\overline{w}$, and then in decreasing order of c_v.

ID	δ	# vertices	sim_{min}	sim_{max}	$\overline{w}$	c_v
cluster_59_3	1	3	175	179	0.9779	0.0092
cluster_88_2	1	2	171	171	0.9448	0
cluster_92_2	1	2	171	171	0.9448	0
cluster_78_2	1	2	168	168	0.9282	0
cluster_85_2	1	2	168	168	0.9282	0
cluster_95_2	1	2	164	164	0.9061	0
cluster_66_3	1	3	153	162	0.8692	0.0234
cluster_39_5	1	5	122	181	0.8586	0.1732
cluster_70_3	1	3	132	179	0.8177	0.1481
cluster_86_2	1	2	146	146	0.8066	0
cluster_100_2	1	2	144	144	0.7956	0
cluster_55_4	1	4	109	177	0.7716	0.1914
cluster_83_2	1	2	136	136	0.7514	0
cluster_93_2	1	2	136	136	0.7514	0
cluster_14_13	1	13	118	181	0.7458	0.093
cluster_84_2	1	2	132	132	0.7293	0
cluster_87_2	1	2	128	128	0.7072	0
cluster_101_2	1	2	124	124	0.6851	0
cluster_90_2	1	2	123	123	0.6796	0
cluster_43_5	1	5	97	150	0.6785	0.1442
cluster_98_2	1	2	122	122	0.674	0
cluster_46_5	1	5	106	143	0.6492	0.0836
cluster_77_2	1	2	117	117	0.6464	0
cluster_37_6	1	6	100	170	0.6287	0.1401
cluster_38_6	1	6	77	181	0.6254	0.2368
cluster_31_6	1	6	91	173	0.6254	0.2452
cluster_64_3	1	3	109	115	0.6225	0.0233
cluster_51_4	1	4	97	135	0.6225	0.1076

cluster_67_3	1	3	97	134	0.6225	0.1387
cluster_44_5	1	5	83	159	0.5983	0.1858
cluster_81_2	1	2	106	106	0.5856	0
cluster_63_3	1	3	76	164	0.5856	0.387
cluster_103_2	1	2	105	105	0.5801	0
cluster_35_6	1	6	70	173	0.5797	0.3016
cluster_79_2	1	2	104	104	0.5746	0
cluster_102_2	1	2	104	104	0.5746	0
cluster_29_7	1	7	69	181	0.5738	0.2953
cluster_91_2	1	2	103	103	0.5691	0
cluster_30_6	1	6	72	125	0.5617	0.1618
cluster_22_8	1	8	69	179	0.5345	0.2597
cluster_10_17	1	17	2	181	0.5225	0.4356
cluster_15_12	1	12	32	181	0.5192	0.3275
cluster_68_3	1	3	81	116	0.512	0.1781
cluster_54_4	1	4	57	140	0.5046	0.292
cluster_74_3	1	3	88	94	0.5028	0.0269
cluster_65_3	1	3	84	91	0.4862	0.0335
cluster_73_3	1	3	82	99	0.4862	0.0885
cluster_58_4	1	4	31	159	0.4807	0.6329
cluster_41_5	1	5	19	170	0.4801	0.65
cluster_97_2	1	2	86	86	0.4751	0
cluster_20_9	1	9	44	181	0.4731	0.3859
cluster_61_3	1	3	69	110	0.4623	0.223
cluster_72_3	1	3	79	90	0.4604	0.0574
cluster_57_4	1	4	54	174	0.4586	0.4987
cluster_13_13	1	13	27	148	0.4567	0.5296
cluster_48_4	1	4	44	175	0.4383	0.6167
cluster_69_3	1	3	47	134	0.4291	0.5136
cluster_18_10	1	10	25	179	0.4231	0.5094
cluster_80_2	1	2	76	76	0.4199	0
cluster_3_33	0.9697	33	1	181	0.4194	0.471
cluster_49_4	1	4	5	162	0.4134	0.9411
cluster_4_31	0.9441	31	1	181	0.4085	0.3992
cluster_6_22	0.8571	22	1	177	0.4062	0.6528
cluster_27_8	1	8	33	181	0.3958	0.599
cluster_24_8	1	8	51	177	0.3848	0.3534
cluster_99_2	1	2	67	67	0.3702	0

cluster_16_11	1	11	10	181	0.3624	0.5994
cluster_28_7	1	7	9	177	0.3452	0.6916
cluster_9_17	1	17	21	181	0.3418	0.5558
cluster_94_2	1	2	61	61	0.337	0
cluster_21_9	1	9	25	153	0.334	0.5328
cluster_2_35	1	35	8	179	0.3282	0.5058
cluster_8_18	1	18	7	181	0.3233	0.5299
cluster_32_6	1	6	14	125	0.323	0.6647
cluster_11_14	1	14	6	115	0.3203	0.5347
cluster_23_8	1	8	18	170	0.3122	0.5712
cluster_52_4	1	4	46	71	0.3112	0.1452
cluster_12_14	0.9341	14	1	177	0.3008	0.5151
cluster_40_5	1	5	23	99	0.3006	0.4875
cluster_5_26	1	26	1	177	0.2936	0.7904
cluster_42_5	1	5	23	133	0.289	0.6488
cluster_45_5	1	5	13	170	0.2867	0.9479
cluster_19_10	1	10	3	131	0.2689	0.9271
cluster_34_6	1	6	9	129	0.26	0.907
cluster_36_6	1	6	18	115	0.2472	0.6774
cluster_17_10	1	10	3	157	0.2442	0.9162
cluster_47_5	0.8	5	2	128	0.2417	1.177
cluster_82_2	1	2	43	43	0.2376	0
cluster_89_2	1	2	40	40	0.221	0
cluster_56_4	0.8333	4	5	128	0.2188	1.1518
cluster_1_50	0.9918	50	1	165	0.2074	0.594
cluster_53_4	1	4	4	181	0.2035	1.7524
cluster_26_8	1	8	3	179	0.2017	1.0894
cluster_76_3	1	3	14	70	0.1842	0.7782
cluster_7_19	1	19	9	150	0.1825	0.7418
cluster_62_3	1	3	13	65	0.175	0.7461
cluster_75_3	1	3	24	39	0.1639	0.2242
cluster_71_3	1	3	19	38	0.1492	0.2978
cluster_25_8	1	8	8	143	0.1433	0.9426
cluster_60_3	1	3	15	34	0.1216	0.3875
cluster_33_6	1	6	4	173	0.105	2.173
cluster_50_4	0.8333	4	1	47	0.0774	1.2172
cluster_96_2	1	2	1	1	0.0055	0

Table 27: The results of all the MCL clusters of the *mleci* graph.