Ontology Alignment in the Presence of a Domain Ontology

Finding Protein Homology

by

Andrew August Carbonetto

B.Sc., McGill University, 2005

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

The Faculty of Graduate Studies

(Computer Science)

The University Of British Columbia

(Vancouver)

May, 2008

Abstract

Cheap electronic storage and Internet bandwidth has increased the amount of online data. Large quantities of metadata are created to manage this wealth of information. Methods to organize and structure metadata has led to the development of ontologies - data that is organized to describe the relation between elements. The creation of large ontologies has brought forth the need for ontology management strategies. Ontology alignment and merging techniques are standard operations for ontology management.

Accurate ontology alignment methods are typically semi-automatic, meaning they require periodic user input. This becomes infeasible on large ontologies and the accuracy and efficiency drops significantly when these algorithms are forced to align without human interaction. Bioinformatics, for example, has seen the influx of large ontologies, such as signal pathway sets with thousands of elements or protein-protein interaction (PPI) databases with hundreds of thousands of elements. This drives the need for a reliable method of large-scale ontology alignment.

Many bioinformatics ontologies contain references to domain ontologies - manually curated ontologies describing additional, general information about the terms in the ontologies. For example, more than 2/3 of proteins in PPI data sets contain at least one annotation to the domain ontology the Gene Ontology. We use the domain ontology references as features to compute similarity between elements. However, there are few efficient ways to compute similarity from structured features. We present a novel, automatic method for aligning ontologies based on such domain ontology features.

Specifically, we use simulated annealing to reduce the complexity of the domain ontology's structure by finding approximate relevant clusters of elements. An intermediate step performs hierarchical clustering based on the similarity between elements of the ontology. Then the mapping between clusters across aligning ontologies is built. The final step builds an alignment between matched clusters.

To evaluate our methods, we perform an alignment between Human (Homo Sapiens) and Yeast (Saccharomyces cerevisiae) signal pathways provided by the Reactome database. The results were compared against reliable homology studies of proteins. The final mapping produces alignments that are significantly more accurate than the traditional ontology alignment methods, without any human involvement.

Table of Contents

List of Tables

List of Figures

Acknowledgements

The thesis requires a huge acknowledgment to my second supervisor: Francis Ouellette. Without his help, his time and his suggestions, this project would not have happened. He provided expert advice on protein-protein interaction networks, GO annotation and everything else biological.

This project is dedicated to my father who wanted me to complete writing my thesis as much as I did. Also to my puppy, Evolution, who passed away in such an untimely manner.

Chapter 1

Introduction

This chapter explains the background knowledge necessary to understand the algorithms and methods detailed in this thesis. A rudimentary knowledge of Computer Science is assumed, but no biology is needed. This overviews a formal definition of ontologies and some of the methods using in designing out method.

1.1 Overview of Ontologies

In recent years, there has been an increasingly large amount of data because of cheap storage space and ease of acquiring large amounts of data from various sources. There has also become a large market for metadata - data that organizes and annotates data. Medical applications use a combination of digital information and expert analysis. For example, hospitals now keep track of a variety of different data sources, such as patient data, staff data, diagnosis, treatment and solution data. These different data sources need to be annotated with metadata to develop relationships. Patients are treated by doctors and medical staff, are diagnosed with a specific illness or injury, and a specific treatment is applied. Other doctors can use this knowledge to help treat further problems: however they need to read, modify and understand the metadata relationships.

Simple annotation is often not enough to fully characterize a data element and the relationships it might own. For example, a patient might have a red blood cell anemia: something is causing the patient to have deficient oxygen transport by their hemoglobin caused by either a low blood cell count or deficient oxygen-iron binding. A seemingly simple hemoglobin protein complex can be annotated by *molecular weight*, *protein subunits*, *host species*, etc. A protein complex's function are often categorized by families, and metadata is needed to cluster protein families by function. To annotate family of individuals, such as proteins similar to hemoglobin, one needs to develop a relationships between data elements: *hemoglobin* has similar function to *myoglobin* and symmetrically *myoglobin* has related function to *hemoglobin*. The functional deficiencies of the patient could be better understood if related information, such as the myoglobin tp hemoglobin relationship, is readily accessible by medical staff to make the best possible decision for treatment.

Metadata describing the relationships between elements of a database can be of an order of magnitude larger than the actual data. So metadata is often stored in a separate database.

One way to organize metadata databases is with ontologies. Ontologies are structurally loose data models, ideally suited to represent relationship metadata in an appropriate manner. They are ordinarily investigated in artificial intelligence, the semantic web, linguistics, and philosophy to infer (reason) relationships between entities. The effect is to provide a defined set of relationships between elements of a database. For example, *hemoglobin* and *red blood cell* terms are synonymous, and should be treated as synonymous by any application.

1.2 Structure of Ontologies

Ontologies are first and foremost collections of relations that represent the relationships between elements, and [19] describes their composition in detail. In general, these elements are either:

- **Classes** are abstract elements that do not describe individual instances of the world. Like object-oriented design, classes can be arranged hierarchically with sub-classes inheriting attributes from super-classes. For example metalloproteins are a set of proteins that have metallic binding potential. When describing the class of *metalloproteins*, it is necessary to make it a sub-class of the class *proteins*. Metalloproteins inherit all the attributes of proteins. Using this method, we can use classes to either describe every term in the universe, or within a specific domain.

- **Individual** elements represent instances that exist in the real world. Each individual is described by a class. For example, *hemoglobin* is a member of the class *metalloproteins*. In inherits all the attributes from the abstract class, as well as all the super-classes (such as from *[proteins*).

Classes can be described by any number of **attributes**. Attributes can be of a number of data types, including strings, integers or characters. They are used to uniquely describe the characteristics of all members of the class.

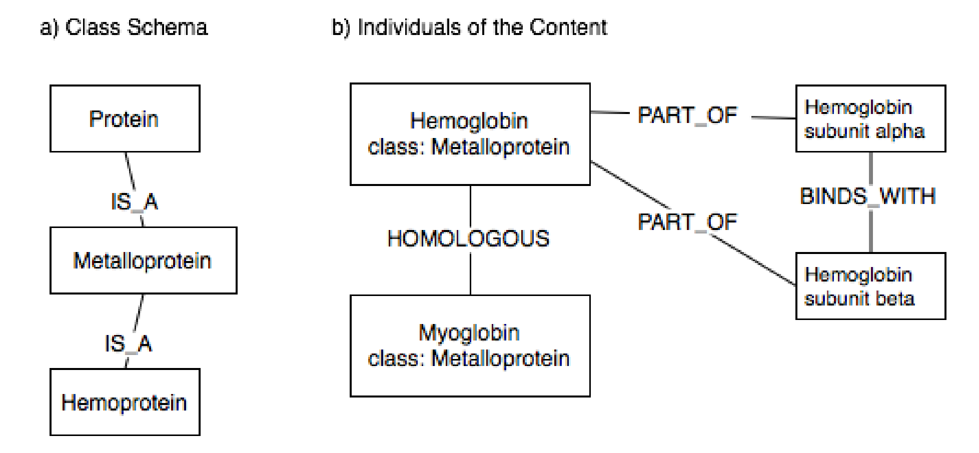

Figure 1.1: An example of the relationships between hemoglobin molecules and related proteins.

For example, our class of proteins might be characterized by their *label*, *molecular weight* and *sequence of amino acids*.

Ontologies use relationships to describe how elements relate to one another. We now describe relationships as properties and as predicates:

- Relationships between class elements are called **properties**. They describe how elements of a class relate to elements of another class. For example, the class of metalloproteins *is_subclass_of* (or more commonly, *is_a*) the class of proteins. Another common property is *part_of*, which describes components of classes. Class properties are hierarchical, in the sense that 'child-classes' are typically more specific classes of their 'parent-classes'. The resulting graph is directed and acyclic. The root (i.e., the super class to which all classes belong to) of the graph is often called a 'thing'.

- **Predicates** in ontologies are relationships between two individual elements. They are often much less restrictive than properties, as they often have no hierarchical ordering. Properties can be described in other manners, for example *transitive*, *asymmetric*, *symmetric* or have a *cardinality*. Figure 1.1 has several examples of prodicates: hemoglobin is *HOMOLOGOUS* to myoglobin; Hemoglobin subunit alpha *BIND_WITH* Hemoglobin subunit beta, and both are *PART_OF* hemoglobin molecules.

For our hemoglobin example (Figure 1.1) we know: The class *Metalloproteins* **is_a** *protein*, which is a hierarchical subclass property. The *hemoglobin* complex is an individual is of class type *metalloprotein*. Both *Hemoglobin subunit alpha* **is_part_of** *hemoglobin* and *Hemoglobin subunit beta* **is_part_of** *hemoglobin*, which describes the two globin subunits of hemoglobin. The predicate Hemoglobin subunit alpha **binds_with** Hemoglobin subunit beta describes the 3D binding activity that is required to form a hemoglobin molecule.

1.3 Domain Ontologies

Due to the explosion of data, it is not uncommon to find ontologies that attempt to describe an abstract concept. Dictionaries have catalogued and described languages for years, but now linguists have converted dictionaries to digital formats with 2nd or 3rd degrees of metadata. Ontologies that describe an entire domain are called domain ontologies. When these domain ontologies are reliable and complete, they are excellent devices for metadata since they are reliable and complete. Unfortunately, it can be difficult to decide when an ontology is reliable or complete because of the ever changing nature of domains. For example, English dialects from region to region can be radically different.

Regardless of the ever changing nature of ontologies, the great advantage of utilizing domain ontologies lies in their composition. We use the versatility of these domain ontologies to describe classes, and understand how the metadata of these classes relate. For example, we have already described that hemoglobin and myoglobin are homologues (Figure 1.1). If we investigate how related the attributes of these protein complexes are, we find that both hemoglobin and myoglobin are *oxygen binding*, which describes their similar molecular function. However, only hemoglobins are described as *oxygen transporter activity*, which describes the difference between functions.

Domain ontologies can also be a curse, in the sense that they, by definition, contain error. Domain ontologies represent real world domains, their static attempt to imperfectly model a changing and imprecise world means that domain ontologies need to be constantly updated and corrected. There are several errors that might originate from domain ontologies:

- Domain ontologies can contain synonyms and mistakes. Synonyms arise because they are often described using English words. For example, hemoglobin and red blood cell can be used interchangeably. Mistakes arise from incorrect data entry. Both can be corrected by using conventional data cleaning methods. For example, [18] tries to correct synonyms and errors found in taxonomic domain ontologies.

- New discoveries in research often increase or contradict existing definitions. As these discoveries are defined, more specific terms can be added to a domain ontology or can replace existing branches. By using out of date information, inconsistencies might arise between outside data and the domain ontology.
- There will always be demand for domain ontologies that cover multiple domains. For example, research that includes genomic, molecular and chemical information might require a domain ontology in both chemistry and biology.

Some of these problems can be solved, at least in theory, by upper ontologies. Upper ontologies try to explain every concept from every domain. Some examples of upper ontologies include SUMO [41], and Cyc [29].

1.4 Aligning and Merging Ontologies

When a single, existing, ontology is not adequate, it might be necessary to manipulate several ontologies at the same time to combine their information ([11] and [3]). For example, hemoglobin molecules are well studied in mammalian species, but the oxygen transportation system of a new model species is less well understood. We could predict well the functions of the new model species' hemoglobin by merging the two ontologies: human hemoglobin and the new model species' hemoglobin. This would reveal the similarities between the two species.

There are generally two types of automatic ontology manipulation: ontology alignment and ontology merging ([44] and [43]). The process of alignment creates a mapping between two input ontologies. The mapping is a set of anchors between the two ontologies (an edge connecting two elements of the ontologies), with confidence values that rate the algorithm's confidence level in associating the ends of the anchors. For example, if ontology alignment was used across two ontologies of the same domain, it could be used to find similar labels or synonyms between the ontologies. An algorithm could give a confidence weight to each mapping based on its level of confidence. [42] describes some methods for ontology alignment. Ontology merging follows from alignment. Merging builds an ontology based on how the ontologies aligned. In this thesis, we are only concerned with ontology alignment.

The confidence score of the alignment mapping is derived from formulas that rate similarity. There are two main methods to rate similarity between elements, *internal structure similarity* and *external structure similarity*.

Internal structure similarity finds how similar two elements are based on their attributes, such as a string comparison of their labels or even the number of attributes. For example, hemoglobin proteins have many attributes, including the label name and amino acid sequence. A reliable way to find similarity might be to find the homology (see Section 2.1) between two molecule's amino acid sequences.

The *external structure similarity* defines the similarity between two elements based on how the element relates to its neighbouring (or related) elements. Some measures might include the element's out-degree, in-degree or distance from a root of the graph. For example, hemoglobin might be more related to myoglobin than other oxygen transportation molecules. Despite having similar function, both hemoglobin and myoglobin are members of the hemoproteins class and both contain *heme* parts - the oxygen binds to iron. Other oxygen transport molecules tend to use metals other than iron.

A *mapping* $M_{S \to T}$ between a source and target ontology S and T, respectively, consists of a set of 3-tuples edges representing a relationship between elements of S and T. In the case of $M_{S \to T}$, the set of edges of M are of the form $< s, \textit{weight}\ , o >$, where $s \in I_S$ is a subject element in the source schema's set of individuals I_S, $o \in I_T$ is an object element in the target schema's set of individuals I_T and *weight* is the confidence score of the mapping edge.

1.5 Aligning Methods Using Contexts and Sets

Contexts are defined here as the background domain for a given element. It is used to differentiate ambiguous terms. For example, bat is synonymous for a flying rodent, a sports object in baseball, a verb that uses the sports object or a verb meaning to flutter one's eyelashes. We can attach a context to the word, that makes it dis-ambiguous, such as into a sentence: *The bat flew away.* We do the same thing with elements of an ontology. Elements become less ambiguous when in groups with a common *theme*.

Other areas of data management have used contexts to increase the accuracy of their methods: query search [2], the semantic web [53], and schema matching ([32], [51], and [22]). Our method uses contexts defined by domain ontologies (or more specifically, a sub-ontology of the domain ontology) that takes into account the structure of the ontology. Context in this case is defined as the set of annotation terms of the elements, the significance of individual terms and a distance measure between pairs of terms.

To find a context within an ontology O (as will be applied to O_1 and O_2), we find the groups of elements that have identical annotation terms. We can relax the definition of identical annotation terms by the terms' definition: we will say that term classes intersect with their super-classes, but to a milder extent; we will say that term classes intersect with all their sub-classes, but to a milder extent. Then we can further define what an annotation term is by describing learning how much that term contributes to the context.

The final product is a well defined context described by a sub-domain ontology. The resulting contents of the sub-domain ontology would contain all the terms that contribute to the context. We can distribute the elements of the ontology to the various contexts, and thus define the contexts of which the ontology terms are owned to.

Extending the metallo-protein example, in Figure 1.2 we see a sample of the GO terms describing hemoglobin and myoglobin and a possible context pairing both of these proteins would contain the description of the terms common to both proteins.

1.6 Related Work

Ontology alignment is a specific problem of generalized schema matching: finding mappings between elements of multiple schemas ([30]). The input schemas might consider the same data, but encode multiple representations (such as XML and relational schemas). Schema mapping considers two important structures of the schemas: the internal attributes of individual elements and the external structure describing the overall relationship of the elements of the schema.

For the past two decades, there as been serious work produced for creating schema matchings. [15] lists many of the important recent discoveries, and categories these algorithms based on the target area. However, many of these programs cannot address some of the key challenges for schema matching. Mainly, schemas tend to have error produced by inaccurate or outdated sources, and thus the schema data, clues or elements can be incomplete. Also, it can be difficult to determine the optimal or ideal matching between schemas because matching can often be subjective. To address the problem of outdated or incorrect data sources, [17] reviews techniques that

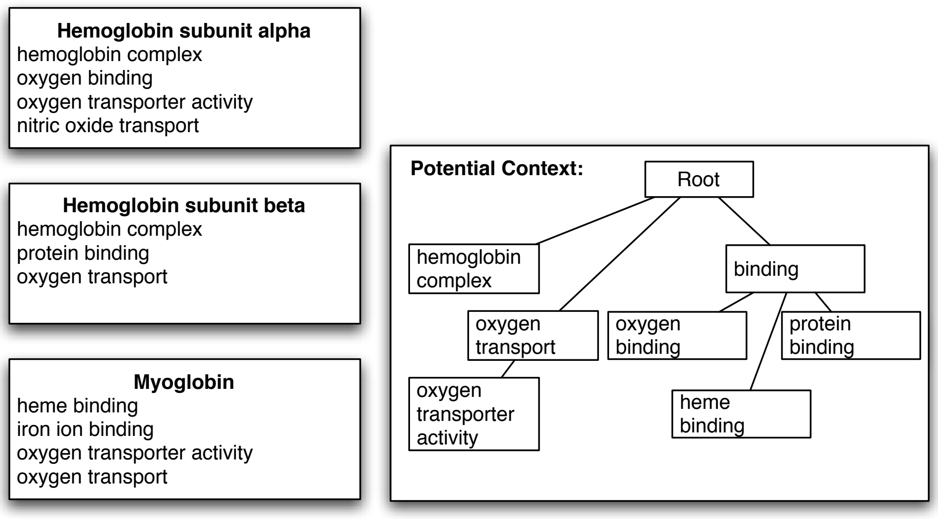

Figure 1.2: Given the three proteins on the left, with their respective GO term annotation. All three have similar function, but not exactly the same function. We provide a potential context, on the right, when all three proteins are clustered together. The overall context could be described by oxygen transport (with slight oxygen transporter activity) and general binding potential.

perform data integration specifically on data that contains error. They take into consideration the quality of the data source before integrating.

Ontology alignment is a similar problem to schema integration, but with another rich dimension that makes the problem using [46]. In schema integration, we are typically less concerned with the relationships between elements. In fact, in some schema integration mappings, the relationships are combined. For example, in one schema, we might have *street number* and *street* as elements that map together to *address* in another schema. Ontology alignment is a difficult problem that can be automated by the PROMPT suite [45].

Proteomics use ontologies to represent how groups of proteins interact. The interactions are often described as networks of proteins (such as signal pathways, protein-protein interaction networks, genomic interaction networks as described in Section 2.2). One of the main reasons for creating these networks, is to align the networks across multiple species. The problem is conveniently solved by ontology alignment methods.

There are several methods to merge protein network ontologies as described by the survey of [28]. Many of these methods are specific to the investigated ontologies, perform above and beyond traditional ontology alignment algorithms as described above, but cannot be generalized to the general ontology alignment problem. They perform very well because of their specific nature [31], however they often still suffer from the same problems as the general ontology alignment problems [39].

Since domain ontologies are now being used as annotation in number of problems, it seems obvious to take advantage of the rich relationships available in these schemas to perform ontology alignment. There is no trivial manner to use ontologies as annotation, and so we present a solution to align ontologies that have a domain ontology as annotation. We show that domain ontologies contain additional information specific to that domain, and that the use of biological domain ontologies perform comparatively to ontology alignment algorithms specific to biology without being restricted to only bio-ontologies.

1.7 Overview of Contributions

Ontology alignment is a hard topic to solve, because of both the complexity of the system and uncertain reliability of the application domains. We provide a method that aligns ontologies that already have structured metadata in the form of ontologies (details in Chapter 4). The structured metadata is annotation on the input ontologies (the ontologies to be aligned) that is itself described by an ontology and a mapping to that ontology. This describing ontology is a domain or upper ontology (the description of domain and upper ontologies are somewhat subjective). This is useful for application areas where metadata is described using ontologies. For example, in the domain of bio-ontologies, various standards for annotation and metadata are emerging from popular use. One such example of a well used standard is the GeneOntology (GO) bio-ontology [1]. As explained further in Chapter 5, we show how some of these bio-ontologies can make use of their GO metadata for ontology alignment.

Our method provides the following contributions to ontology alignment methods:

- Aligning ontologies is a hard question and can require a lot of time to process for large ontology sets. This is because (in the worst case) each pair combination of elements of the aligning networks needs to be compared with every other element to be mapped. When using a domain ontology for annotation, comparing a pair of elements from the input ontologies commonly requires scanning through the entire domain ontology for each possible pair of annotation - depending on the comparison method used. This can take $O(|O_1||O_2||O'|^2)$ time for two input ontologies O_1 and O_2 and domain ontology O' as annotation. For large input ontologies (such as protein networks) or large domain ontologies (such as GO) with hundreds of thousands of nodes, this problem can become infeasible. Instead, we offer a method to approximate the mapping by using domain ontology contexts, which reduces the problem to expected time complexity $O(|O_1||O'|+|O_2||O'|)$ for preprocessing (using an iterative refined approach called simulated annealing, described in Section 3.3 and 4.3), $O(|O_1|^2 + |O_2|^2)$ for clustering and $O(|O'|)$ for constant sizes number of clusters (as described in Section 4.3).

  We hypothesize that preprocessing does not need to be completely recomputed so long as the given ontologies do not change significantly (changes need to be re-computed). We leave the validation of this hypothesis for future work.

- We have developed a novel method to compare two elements with annotation in the same domain ontology. This method takes time $O(|O'|)$ to compute by counting the number of intersecting elements shared between both elements. We have extended this method to take into account human error found when most domain ontologies are built and maintained.

- We demonstrate a novel framework for clustering elements (see Section 3.1) from both input ontologies into contexts. These contexts are described by ontologies, to preserve the relationships of the ontology. This is useful for the following 3 reasons: 1) the clustering approximation of the problem does not drop relationships, and thus does not lose information within the ontology relationships. 2) contexts have been used to improve the accuracy of schema matching, and we use contexts to improve the accuracy of ontology alignment. 3) contexts can be also be used to describe the function when using proteins. This is convenient to easily understanding the purpose of contexts and matches when evaluating the results subjectively.

- Our method is easily generalizable to any alignment of ontologies with a mapping to a common domain ontology. The results, analysis and scoring methods used are specific to biological our input set. The problem is defined in Section 4.1.

- We apply our method to finding protein similarity across multiple species. This area is well suited to be tested by our method. Proteins are often annotated using the GeneOntology domain ontology, and their interactions are described using ontologies. The GeneOntology is curated by experts and has become a standard used throughout biology. We expect that the relationships described therein have minimal error. Two sets of non-intersecting proteins are taken from different species with a known common function. A mapping, representing similar proteins, is resulted. Further details are explained in Chapter 2.

- To demonstrate our method on protein networks, we develop a scoring method that does not directly use protein homology (described in Section 2.1) as means of finding similar proteins. There could be indirect influence. Some proteins are mapped to GO terms by prediction through protein homology. We selected proteins networks with fewer GO terms predicted from protein homology. We could thus use protein homology to test for the accuracy of our method.

- We evaluate the use of GO terms as context for groups of proteins, and make claims for the significance of GO terms (see Section 5.1.5). That is to say, some GO terms are a less impact on the context of the protein. In other words, we attach a weight to each GO terms mapping based on it's significance to the protein. This produces contexts with only the significant terms, to maximize the description. We show that proteins with known groupings have much more significant shared GO terms then proteins from random groupings. This helps confirm that the use of significance improves results.

- The final mapping produced contains confidence scores: these are weighted mappings that rank the confidence our method has in the mapping produced. Confidence scores are standard results of mapping across ontologies or schemas, such as in Harmony [36]. Boolean mappings between ontologies are often defined using a cutoff threshold. The results from methods that use confidence scores are easy to sum together if desired. This is useful for subjective analysis or ontology merging.

Chapter 2

Aligning Proteins of Bio-Ontologies

2.1 Protein Homology

Proteins evolve in a similar manner to species (the exact differences are beyond the scope of this thesis). Protein evolution is often displayed as a *Tree of Life*, such as 2.1. One measure of similarity between proteins is by the separating distance in the tree of life (sum of all edges between the two proteins). The nearest common ancestor of two proteins is the highest internal in the graph that connects the two proteins: this node represents the speciation event that caused the separation of proteins.

Protein *homology* is a biological method that matches proteins by their common ancestors. There are many implications that can be made about two proteins that share common ancestors, all of these implications have exceptions. Some implications include shared function roles in signal pathways, common mutation rates, similar protein translation and more. We use protein homology to as comparison to test out method.

The most common method to detect homology between proteins is through pairwise or multiple sequence alignment. These produce scores that measure the similarity of two or more proteins. We make the implication that higher scores relating proteins implies that proteins have a greater chance of having similar function.

This helps understand the diverging event that causes the common ancestor to produce two distinct proteins. Commonly, these proteins evolve in two manners: orthologously and paralogously.

Orthologs (meaning: proteins that are orthologs) describe pairs of proteins that are expressed by the genomes of different species but contain a common ancestor. The two species also have a common ancestor species. Since we know that these proteins have a common ancestor protein, we can infer that both of these proteins have a common function (shared with the ancestor protein). Over time, both proteins have evolved but retained some

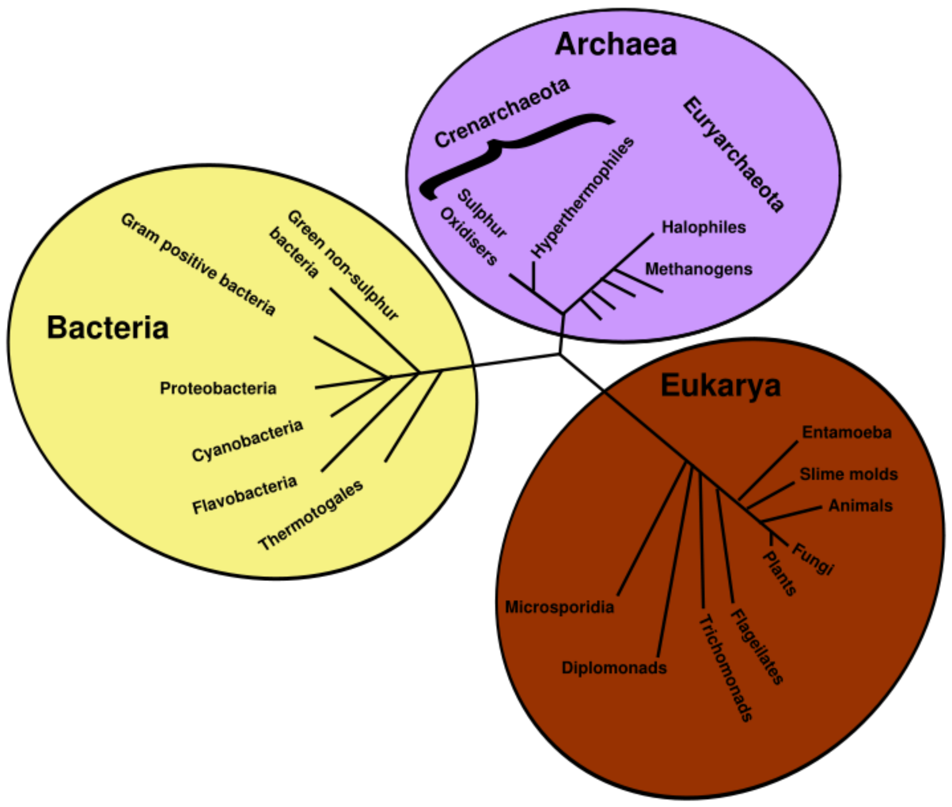

Figure 2.1: Tree of life automatically generated by [10]. The rounded tree has the root at the centre, which represents the ancestor species of all life forms. Decedents of the root evolve over time - represented as the distance from the center of the graph to the edges. Evolution involves speciation events at each internal node, creating two daughter species.

degree of similar function (this depends on a number of factors, including the 'distance' between species, the importance of the protein to the species' fitness, etc). Since we are investigating divergent species (human and yeast), we would expect many differences between proteins - even if they share a common function.

Paralogs (meaning: proteins that are paralogs) describe pairs of proteins that are expressed by a single species, but are translated from different locations on the genome. This is most commonly found from a mutation event that duplicates an entire gene. Both the original and duplicate genes express the same protein until other mutation events occur on one or both of the genes. Duplicate genes might introduce new selection pressures which could increase mutations. After time, the proteins could have divergent function.

2.2 Bio-Ontologies

Biology is a very complicated domain, and the networks that model the domain tend to be large, unpredictable and unreliable. Structured metadata schemes are becoming increasingly necessary to the science. Recently, a wealth of ontologies that attempt to explain sub-domains of the biological universe have been created. These ontologies are commonly referred to as bio-ontologies.

Bio-ontologies deal with many domains, including protein function, molecular interactions, cell construction and species relatedness (to name a few). These ontologies can be large, covering tens or hundreds of thousands of elements. All the possible interactions can be verified by experimentation since biological information is more reliable when verified by experts or experimentation. Interactions can be predicted accurate by computational methods, which can narrow the search space.

There are several well used bio-ontologies formats:

1. Protein-protein interaction networks try to understand the complicated networks, or complexes, that proteins work. Individual proteins tend to have very little function, but in groups they can form networks that can perform actions such as the reproduction of DNA and RNA, and run complicated cells such as neural synapses. DIP [48], IntAct [24] and BIND [4] are the three large protein-protein interaction databases, and contain hundreds of thousands of proteins and protein-protein interactions.

2. Signal pathways take an alternative approach to protein-protein interaction. They are networks of a particular function and describe all interacting molecules that, combined, provide that function. For example, RNA translation involves a complicated network of proteins, ligands and RNA molecules. The number of proteins and molecules involved is much smaller than what is in protein-protein interaction networks, but the description of interactions is better described. KEGG [21] and Reactome [57] are some examples of signal pathway network repositories.

3. The manually curated, and well maintained bio-ontology, the Gene Ontology (GO) [1], is a domain ontology for molecular genetic terms. GO has provided a set of annotations, and the known corresponding relationships, for molecular function terms, cellular component terms and biological process terms of molecules. The molecular function sub-domain hierarchically categorizes molecular functions (for example, oxygen transportation). The cellular component hierarchically categorizes the components of cells so that researchers can associate components with function (for example, inter- or intra-cellular). The biological process lists the general process the molecule is involved in (for example transcription or cell death).

2.3 Finding Protein Function in Protein Networks

The objective of all protein networks: be they protein-protein interactions, signal pathways or otherwise, is to describe the function of proteins. By function, we often mean the manner that protein interacts with its environment, including other proteins and molecules. Like the gears of a clock that individually only turn, proteins have very simple functions individually. Combined, they have greater functions such as duplicating DNA and RNA, metabolize molecules or transport oxygen. The act of forming protein complexes is a conserved quality: these complexes will be of similar size across species [50].

We can refer to the *context* of a protein element as the background function in a cluster of proteins. Contexts (as described by [56]) have been used in schema matching [14] and ranking database queries [2]. We will be using context is a related manner to evaluate and rank the matching potential of multiple clusters of protein elements.

We further describe the use of annotation to find protein function with respect to our data set in Section 5.1.4.

2.4 Aligning Protein Network Ontologies using a Genomic Domain Ontology

Ontology alignment is a difficult problem, and is very related to a similar problem: protein homology in biological networks. For example, the Ulysses project [23] and [33] use protein homology to find similar networks of proteins. [40] uses protein homology and a list of GO annotation for protein function homology. However, the match-and-split algorithm does not take the structure of the annotation into consideration - but uses a established method by [25]. Protein homology is less than 100% accurate at finding proteins with similar function [20], and alternative methods are needed to complement protein homology to increase the accuracy of finding proteins with similar function.

We use our method to demonstrate how protein function can be predicted across species in a complementary manner than protein homology. This uses the metadata ontology GeneOntology (GO) that is common to many known proteins. The GO is ideal for our purposes because:

1. much of the GO is manual curated, which reduces the amount error.
2. there are enough protein products that have enough GO metadata to find protein function
3. GO is divided into 3 sub domains that describe protein function: molecular function, biological process and cellular component. Both the biological function and cellular component sub-domains are relevant to protein function ([47], [54]).

For an example, Tables 2.4 and 2.4 shows proteins aligned together by [40]. It is well established that these proteins work together. Their function or context of either group of proteins could be described by the approximate intersection of their GO annotation: A *mitochondrial inner membrane* protein that *binds to* and *targets proteins* for *transport activity* and *cellular component organization and biogenesis.* These terms were selected because:

1. All proteins share the cellular component *mitochondrial inner membrane presequence translocase complex* GO term.

2. Most of the proteins share the either *protein binding* or *protein targeting* terms.

3. The Terms *protein import into mitochondrial matrix* and *protein targeting to mitochondrion* are sub-class terms of *intracellular protein transport.*

4. The Terms *protein import into mitochondrial matrix*, *protein targeting to mitochondrion* and *mitochondrial membrane organization and biogenesis* are sub-class terms of *cellular component organization and biogenesis.*

yeast protein	GO terms
TIM17	Cellular component: mitochondrial inner membrane presequence translocase complex Molecular function: protein binding Molecular function: protein transporter activity Biological process: protein import into mitochondrial matrix
TIM23	Cellular component: mitochondrial inner membrane presequence translocase complex Molecular function: protein binding Molecular function: protein transporter activity Biological process: protein import into mitochondrial matrix
TIM50	Cellular component: mitochondrial inner membrane presequence translocase complex Molecular function: protein binding Biological process: protein import into mitochondrial matrix

Table 2.1: Yeast proteins aligned by the match-and-split algorithm. The GO annotation is extracted by the swiss-prot ExPASy database [16].

human protein	GO terms
TIMM17A	Cellular component: integral to membrane Cellular component: mitochondrial inner membrane presequence translocase complex Biological process: protein targeting to mitochondrion
TIMM17B	Cellular component: integral to membrane Cellular component: mitochondrial inner membrane presequence translocase complex Biological process: protein targeting to mitochondrion
TIMM23	Cellular component: integral to membrane Cellular component: mitochondrial inner membrane presequence translocase complex Cellular component: mitochondrial intermembrane space Molecular function: protein binding Biological process: protein targeting to mitochondrion
TIMM50	Cellular component: mitochondrial inner membrane presequence translocase complex Cellular component: nuclear speck Molecular function: protein binding Molecular function: protein serine/threonine phosphatase activity Molecular function: protein tyrosine phosphatase activity Molecular function: ribonucleoprotein binding Biological process: mitochondrial membrane organization and biogenesis Biological process: protein amino acid dephosphorylation

Table 2.2: Human proteins aligned by the match-and-split algorithm. The GO annotation is extracted by the swiss-prot ExPASy database [16].

Chapter 3

Methods Background

This chapter provides a brief background on several methods used in our Ontology Alignment method. We use clustering (see Section 3.1) to find related elements in an ontology based on a provided distance metric. Some commonly used clustering methods, including hierarchical clustering and k-means clustering are provided for reference. The Hungarian Algorithm is described in Section 3.2, which provides a method to match maximum weighted bipartite graphs. The final matchings produced by our method are matched using the Hungarian Algorithm based on the confidence scores produced. Simulated Annealing (see Section 3.3) is lastly described for the simulation of models with thermodynamic constraints. Simulated annealing was used for approximating contexts and the pre-processing of our method.

3.1 Clustering Techniques

Contextualization in database management is the process of categorizing database elements into contexts [56]. This process can aid in query search ([2], [53]). This can be interpreted as metadata. The metadata could easily be construed as an ontology.

By using clustering techniques, one can cluster terms with related contexts. We later use this technique to cluster proteins with similar function together, depending on the context of the protein.

The objectives of clustering a set of elements into k distinct subsets, based on a *distance* or *similarity* between the elements. Distances between elements can be in the form of a symmetric matrix with zeros in the diagonal, each row or column corresponding to a separate element.

Supervised techniques are methods that require an initial number of clusters k as input. The advantage is that the elements are optimally partitioned into the desired number of clusters. Outliers are sometimes placed in incorrect or less than desirable clusters, because the number of clusters restriction.

Unsupervised techniques do not require an initial number of clusters. Instead, the method either produces an ideal number of clusters, or results are provided for variable number of clusters.

3.1.1 K-means clustering

Supervised clustering techniques require a k input. **K-means** clustering is a machine learning, supervised clustering technique that requires an initial set configuration of the elements into clusters. The technique iteratively updates the configuration until a 'stable' set is reached, or when the cluster centroids no change significantly from step to step.

1. For each cluster, get the cluster centroid:*mean* of the elements of that cluster.
2. For each elements, find the cluster mean that is closest, and put element into that cluster.
3. Continue steps 1 and 2 until there is no 'visible' change in the cluster means.

This techniques relies heavily on the a priori configuration of the clusters. It also depends on the initial number of clusters, the method for calculating the cluster centroid, and the distance measure. Outliers tend to heavily affect the *mean* of the cluster, which can influence the resulting elements of the clusters.

3.1.2 Hierarchical Clustering

Hierarchical clustering is an unsupervised technique that ranks n elements based on their distance measure, and then builds a hierarchy of clusters. The resulting hierarchy is a cluster configuration from 1 cluster to n clusters. Users can specify any number as the desired number of clusters. There are two methods to approach hierarchical clustering: Agglomerative and Divisive methods.

The **Agglomerative** method proceeds as follows:

1. Every elements starts as its own cluster.
2. The two *most similar* clusters are joined together.
3. Repeat step 2 until there is only one cluster left.
4. Output the results of all steps.

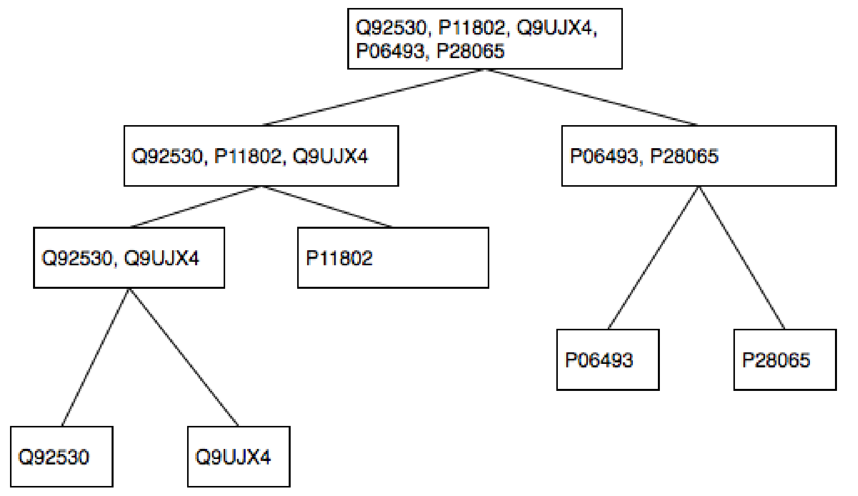

Figure 3.1: Hierarchical clustering of proteins, taken from the swiss-prot ExPASy database [16]. Proteins are clustered into sets. The top step has 1 cluster, the second has 2 clusters, and so on until the last step has 4. Agglomerative methods run from top to bottom, while divisive methods from from bottom to top.

The **Divisive** method proceeds as follows:

1. All the elements are placed in the same cluster.
2. A desired cut is found, that takes one cluster and divides it into two clusters.
3. Repeat step 2 until each element is in its own cluster.
4. Output the results of all steps.

The result is a hierarchy of clusters. Each layer i of the hierarchy has the elements clustered into i clusters. Figure 3.1 represents the progression of joinings or divisions that take place during each method.

3.2 Hungarian Algorithm

Given a weighted bipartite graph $G = (V_1, V_2, E)$ with two sets of vertices that are not connected with edges in E. The edges in E separating the sets

V_1 and V_2 are complete and weighted, that is for every pair $v_1 \in V_1$, $v_2 \in V_2$, there exists a weighted edge (v_1, v_2, w) with weight w connecting the pair. The maximum weighted matching finds a matching between V_1 and V_2 that maximizes the sum of the weights.

This is useful once our method has created a confidence scoring for all pairs of elements in our aligning ontologies. A mapping from input ontology O_1 to O_2 is created using this matching.

The Hungarian Algorithm was originally developed by [38]. It chooses a maximum weighted matching between two sets of vertices V_1 and V_2 in time $O(V^2E)$. We make the assumption that both V_1 and V_2 are not necessarily the same size, and allow for unmatched vertices (these results are minimized). We also relax duplicates, so that if there are multiple possible matchings with equal maximum weight, we output all combinations.

3.3 Simulated Annealing

Simulated Annealing is the process of simulating overly complicated models using a sampling method ([35]), and is a generalization of Monte Carlo and sampling methods ([8]). It has been shown to be beneficial for predicting thermodynamic and computational models [26]. It is an iterative refinement method. At each iteration, the system determines a random action to take, and performs the action given a specific probability. If this increases the stability of the system, then the action is always accepted. If this decreases the stability of the system, the action is taken with a probability equal to this destabilization amount. At random, the system performs random hops to prevent the system from getting stuck in local maximums.

We use simulated annealing to estimate the distance between elements in our input ontologies O_1 and O_2. An appropriate model for the simulated annealing is required to run the system. We give an example of an appropriate model using proteins and their GO annotation and an intersection score in Chapter 6.

Chapter 4

Methods: Aligning Ontologies using a Domain Ontology

We demonstrate a mostly automatic method that appropriately increases the accuracy of conventional ontology alignment algorithms when the input ontologies take into account the structured metadata that is in the form of an ontology. Metadata has become increasingly common and this is also true for structured metadata. Finding similarity between ontologies that contain structured metadata is a hard problem, because of the size of the ontologies and the necessary variability of these ontologies. To solve this, we have built a two stage method that performs ontology alignment. In the first stage, the method reduces the complexity the metadata that annotates the input ontology data. After which, the method then predicts mappings of high confidence between the input ontologies.

As a convenient consequence to our method, we can also investigate the context sets of elements. Context has been shown to be useful in making matches in other areas [52], including schema matching [14], semantic web [53] and query ranking [2]. These areas a related enough to ontology alignment that we have included some of ideas of forming contexts to ontology alignment and reduce the complexity of the method.

The resulting mapping between ontologies is a complete set of weighted edges between the nodes of the ontologies. Each edge contains a confidence weight that expresses the methods confidence that the connecting nodes are similar. These confidence weights can be combined or compared with confidence weights produced by conventional ontology alignment algorithms to amalgamate the results. This method can be used on its own, or as middleware with a combination of other ontology alignment methods to produce satisfactory mappings.

Confidence scoring methods produce a complete matching between the elements of the ontologies. We use a maximum weighted matching algorithm to find the best matchings. However, for common applications, a cutoff threshold used to distinguish between 'good' matchings and 'bad' matchings.

An example is shown in Figure 4.1. We might be very interested in finding a mapping between the proteins Hemoglobin, Myoglobin, Chlorocruorin and Hemocyanin. Since all of these proteins have similar functionality, we would favour a results that strongly maps these proteins together. However, it is evident that matching by labels would not be adequate enough (ex, Hemoglobin and Chlorocruorin have very dissimilar labels). Matching by attribute information might not be good enough since Chlorocruorin binds to copper ions. Matching by external structure would be inadequate because none of these proteins produce a similar complex of proteins (ex, Hemoglobin is a complex of 4 proteins, and Myoglobin has only 1).

If we approach the example by looking at the similarity between Gene Ontology elements, we notice that *Iron Ion Binding* and *Copper Ion Binding* elements are siblings. Thus, we would be able to include Hemocyanin as a strong mapping to the other proteins.

In this chapter, we explain how to align two input ontologies that contain a structurally complicated domain ontology. The algorithm takes as an input two ontologies and their corresponding overlapping domain ontology. By *overlapping*, we mean that there exists a mapping between the input ontologies and the domain ontology - this is further explained in Section 4.2. The algorithm then reduces the complexity of the attributes of the input ontologies to pairwise distance measures. The pairwise distance measures can then be clustered by conventional clustering methods. Section 4.3 explains the process for clustering the elements. Clusters of individual elements of an input ontology model are then described by the subset of the domain ontology. The describing subset is used to match elements across multiple input ontologies. Section 4.4 explains the process to match clusters across ontologies. Finally, Section 4.5 describes the process with which one retrieves a mapping from the matched clusters. The result is a complete mapping with confidence weights between the elements of the input ontologies, which can be used to produce an alignment between the ontologies. See Figure 4.2 for more details.

4.1 Problem Statement

Our problem can be formalized as follows:

Given two input ontologies O_1 and O_2 and a domain ontology O', where there exists a direct mapping $M_{O_1 \rightarrow O'}$ and $M_{O_2 \rightarrow O'}$, return a mapping from

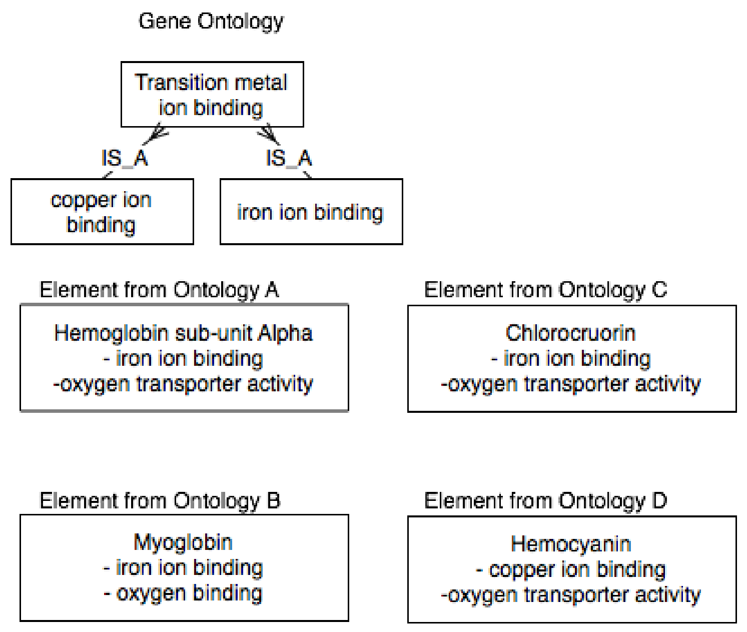

Figure 4.1: Shown is input ontologies A, B, C, and D. Proteins *Hemoglobin subunit-Alpha*, *Myoglobin*, *Chlorocruorin* and *Hemocyanin* represent oxygen binding proteins from various species or parts of the body. Hemoglobin are red blood cells common in humans. Myoglobin are proteins found in human muscles responsible for extra oxygen storage. Chlorocruorin and Hemocyanin are oxygen transport proteins in some annelids and mollusks respectively. The Gene Ontology representation of *metal-ion binding* elements in the molecular function domain.

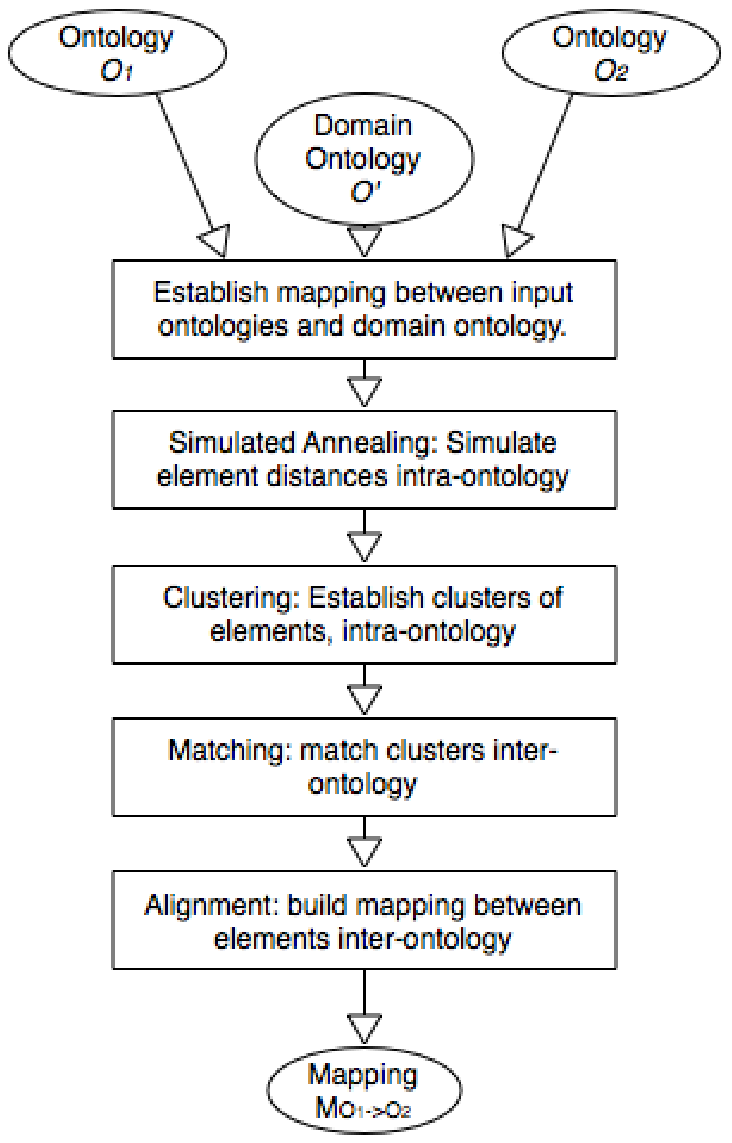
Ontology O1
Ontology O2
Domain Ontology O'
Establish mapping between input ontologies and domain ontology.
Simulated Annealing: Simulate element distances intra-ontology
Clustering: Establish clusters of elements, intra-ontology
Matching: match clusters inter-ontology
Alignment: build mapping between elements inter-ontology
Mapping MO1->O2

Figure 4.2: Overview of our method. The first step, mapping the input to domain taxonomy is only performed if needed.

$M_{O_1 \to O_2}$ with weighted edges such that the edge weights are the minimum distance between significant metadata elements in O' from $M_{O_1 \to O'}$ to $M_{O_2 \to O'}$ for each element in O_1 and O_2. Distance is the minimum number of edges between elements of O'. The significance of elements in O' are optionally weighted by the mappings $M_{O_1 \to O'}$ to $M_{O_2 \to O'}$ or are otherwise all constant.

We can produce an approximation of this problem, defined for multiple elements of O_1 mapped to multiple elements of O_2, to map the context of elements in O_1 to O_2. This approximation takes the same two input ontologies O_1 and O_2 and a domain ontology O' with a mapping $M_{O_1 \to O'}$ and $M_{O_2 \to O'}$. We define a cluster C of an ontology O to contains a mapping to O' that is defined by the respective elements in $M_{O \to O'}$. We return a mapping from $M_{O_1 \to O_2}$ with weighted edges such that the edge weights are the distance between significant metadata elements in O' from clusters $C_1[1], ..., C_1[n]$ from ontology O_1 matching to clusters $C_2[1], ..., C_2[m]$ from ontology O_2. The size of n and m is given. The matching between two clusters minimizes the distance between significant metadata elements in O' contained in all elements of the clusters.

Similarity and significance of metadata elements in O' is described by the *intersection score*, see Section 4.3.

4.2 Mapping Ontology to Domain Taxonomy

If the mapping $M_{O_1 \to O'}$ and $M_{O_2 \to O'}$ from the input ontologies to the Domain Ontology is not given, the mapping can be found in a naive manner. Occasionally the mapping is obvious, where the input ontologies contain a unique identifier to elements in the domain ontology. When these mappings are not obvious, we can use several techniques to estimate the mapping. As a disclaimer, if the mapping is incorrect, it will affect the results of our method, because our method assumes that the mappings $M_{O_1 \to O'}$ and $M_{O_2 \to O'}$ are correct.

4.3 Intra-Model Clustering for Common Domain Taxonomy and Structure

The objective of our method is to find similar elements between multiple input ontologies based on the elements' mapping to a domain ontology and the external structure similarity in these mapped domain ontology elements.

However, to process the external structure between every grouping of elements of the input ontologies would take a long time (an exhaustive search would require $O(|M_{O_1 \to O'}||M_{O_2 \to O'}||O'|)$ time). For large ontologies, this time requirement is infeasible. Instead, we take an approximate approach that takes into account external structure similarity of O', by using simulated annealing (see Section 3.3).

We use simulated annealing to reduce the complexity of structured models of ontologies to a simple pairwise model by sampling. The objective is to find the similarity between pairs of individuals in the model, and not the similarity between the attributes of individuals (as is the objective of feature selection algorithms). The simulated annealing process performs iterations, slowly modifying the clusters of the system. At each iteration, the system accepts the movement of a single element from one cluster to another if the intersection score of the system improves or a random chance if the score of the system does not improve. Individual elements and clusters chosen at each iteration are chosen in a random way (usually from a uniform distribution). We label this a *move* subroutine.

We divide the individuals I of a single ontology $O =< S, I, P >$ into N clusters $\{C_1, ..., C_n\}$. The clusters of the simulation are evaluated based on an intersection score $f(C_i)$. The intersection score of a cluster $f(C_i)$ determines the precedence of the cluster. Clusters with low scores are broken, while clusters with higher scores remain in the simulation.

One of the requirements for simulated annealing to process successfully is for the system to be *ergotic*. That is to say, any state in the universe of the simulation must be able to eventually reach any other state after x number of iterations, where x is a finite number. To maintain ergocity in the system, we include a second state change that can be executed when the system approaches a local maximum, we call it a *break* subroutine. The *break* function randomly cuts a cluster into two smaller clusters. The break selects elements at random to be distributed between the two new clusters.

We also include another subroutine, which we call a *merge*, for completeness. The *merge* function combines a cluster with insignificant score (such as 0) into another cluster without reducing the total intersection score of the system in a significant way. This function maintains the consistency of the number of clusters in the system.

It is trivial to see how this system would be ergotic, because of the amount of uniformly random processes. However, to simulate the distance measures within a system by using so many random iteration functions can take a long time. A system designer using this method can reduce the randomness of each iteration without disrupting the ergocity by tuning the weights attached to selection probability for each function. We show how this is done in the evaluation (Section 6.2.1).

Parameter tuning simulated annealing can affect speed and accuracy. The two parameters that most significantly affect the speed and accuracy of the simulation are 1) the chance of accepting a *move* operation that decreases the systems score and 2) the chance of a *break* operation. Since simulated annealing uses a hill-climbing strategy, and both of these operations cause the system to either climb down or make a random jump, it will slow the system down. Since time was of little issue for the preprocessing step of our method, we were very conservative and made sure our system climbed to the summit of all local maximums.

It is simple to see how the intersection score can affect the clusters sizes in an undesirable way. If the intersection favors larger clusters, the system will gradually shift towards fewer, larger clusters when x gets large. Similarly, with an intersection score that favors smaller clusters, the system will eventually shift towards one element per cluster and $n = |I|$ when x gets large. Thus, it is important to use an unbiased intersection score. An example of an unbiased intersection score can be seen in Section 6.2.1, we normalize the intersection score to reduce the bias in the size of the clusters.

A matrix of size $|N_1|$ x $|N_2|$ (where N_1 and N_2 are the number of clusters produced from O_1 and O_2 respectively) of pairwise distances can then be built by gathering the frequency of the presence of pairs of individuals in clusters in the simulated system. The resulting matrix is an upper triangular matrix with 1's on the diagonal. These matrices are ideal for conventional clustering algorithms, such as hierarchical clustering. We can also get a very good estimate for the number of clusters by looking at the simulation's cluster frequency.

Thus, we reduce the structural complexity of the domain ontology into an upper triangular matrix containing pairwise distance measures which we call a *distance matrix*. We also have a set of clusters of the simulation obtained from hierarchical clustering on the distance matrix.

This divides the problem into sub parts of complexity $O(x(M_{O_1 \to O'} + M_{O_2 \to O'}))$ for preprocessing, which is highly dependent on the number of iteration steps x. $O(|O_1|^2 + |O_2|^2)$ for clustering, but with a small computational constant. $O(N_1 N_2 |O'|)$ for cluster size constants N_1 and N_2 to compute the matching between models (see Section 4.4).

4.4 Inter-Model Matching by Similar Domain Taxonomy

The Inter-Model Matching step in the methods attempts to produce a mapping between the two input ontologies. In previous steps we have estimated the distance measures between individuals in the ontologies, and grouped the individuals into clusters based on these distances. Now, we take the clusters and the domain ontology information that describes these clusters and attempt to perform matching across different input ontologies.

We can abstract the Inter-Model Matching problem to the maximum weighted bipartite set matching problem. This takes a graph $G = (V, E)$ with vertices V and weighted edges E. The clusters formed in each ontology are abstracted to the vertices of the model, and the weighted edges are abstracted as the distance measure, as determined from the describing domain ontology information, between clusters.

Bipartite set matching is well studied problem within algorithms, and several efficient algorithms for finding maximum weighted bipartite matching have been proposed. We used the Hungarian algorithm [38] which solves the maximum weighted bipartite matching problem in $O(V^2E)$ running time.

The result is a weighted mapping between clusters $C_{11}, ..., C_{1n}$ from O_1 and $C_{21}, ..., C_{2m}$ from O_2, where n is the number of clusters from O_1 and m is the number of clusters from O_2. The weight of the mapping between C_{1i} and C_{2j} is produced from the intersection score of $f(C_{1i} \cup C_{2j})$.

4.5 Individual Element Alignment using Internal Cluster Structure

Thus far, we have built an approximation of the original problem. We have assigned weights to edges between clusters, but we require weights assigned to edges between the elements of O_1 and O_2 to produce the mapping $M_{O_1 \to O_2}$. This can be done in one of two ways:

1. Direct transformation: For each weighted edge E_{ij} between clusters C_{1i} and C_{2j}, we create a weighted edge between each element of C_{1i} to each element of C_{2j}. The result simply creates groups of elements with similar *context*. The resulting alignment can easily be combined with other alignment methods to increase accuracy. This approximation becomes more accurate as the number of clusters (N_1 and N_2)

increases.

2. Performing sub-alignments between clusters: For each weighted edge E_{ij} between clusters C_{1i} and C_{2j}, we use a conventional ontology alignment method (such as label matching) with source ontology C_{1i} and target ontology C_{2j}. The unionized mapping from each weighted edge produces a complete mapping between O_1 and O_2. The advantage in using this strategy over conventional ontology alignment methods is that the alignments are divided into *contexts in the domain ontology* before producing the alignment.

Both strategies are evaluated in Chapter 6. The domain is explained in Chapter 5.

Chapter 5

Applications to Protein-Protein Interaction Networks

Proteomics, the study of proteins, is a research field that has invested largely in data and metadata. Proteins are responsible for many biological functions in the human body. Understanding how they work and how they interact, called interacomics, could lead to better understanding of various diseases. Proteins often interact with other proteins, and with other molecules. They can react differently when mutated, or when in different environments such as intra- or inter-cellular. In the end, the amount of data that is being recorded on proteins is immense, all with the goal of understanding how proteins work and how malfunctioning proteins lead to disease.

Signal pathways are maps of proteins and their interacting functions into a specific domain or function. For example, pathways have been designed to illustrate how HIV infects human cells, or how DNA replicates. The maps are detailed as specifically possible, based on the current understanding of the disease and its proteins. Pathways are very important in understanding how proteins play a specific role in a larger function. It is very important when studying disease. For example, hemoglobin require two different proteins to function. The absence of one would cause a deficiency in red blood cells.

There is a great deal of descriptive information that needs to be represented about interactions, and with the large number of interactions, Ontologies are ideally suited to represent these protein networks. Indeed, there exist many forms of biological networks that are already represented by ontologies, such as protein-protein interaction networks, taxonomies and signal pathways. The number of proteins involved can be in the thousands. The size can be burdensome.

All interactions between molecules, be it protein-protein or otherwise, requires experimental verification. Complimentary verification can increase the confidence of a particular interaction. Sometimes, experimental evidence is contradictory, lowering the confidence. Confidence can also be determined by the 'validity' of the experiment, where high-throughput experimentation generally leads to lower confidence levels than low-throughput experiments. We can paint a rough picture of the confidence of various interactions using this model.

However experimental evidence is costly, and we can reduce the amount of error by first predicting interactions *in silico*. The most common and productive method for predicting *in silico* is by comparative analysis with a model species, such a mouse, worm or fruit fly.

To perform prediction of interactions, we need to be able to manipulate the networks. Since the networks are designed using ontologies, we need ontology manipulation techniques to perform serious predictions. Ontology alignment is absolutely necessary if ontologies are to be compared, which is absolutely necessary for *in silico* prediction.

5.1 Overview of Signal Pathways

5.1.1 Analysis and Prediction in Biological Networks

The main purpose of biological networks is to understand how simple molecules, such as proteins, interact to perform some larger function. These networks tend to be similar between similar species. For example, humans and chimpanzees have more similar networks than humans and fruit flies. However, even distantly related networks have some similarities - since the larger function of the combined network have similarities. If given an unknown species, and their associated molecules, one excellent way to predict the molecules' interactions and functionality is with comparative analysis.

The process to find similar proteins between species usually involves sequence alignment - called protein homology. A DNA sequence alignment between a new model species and a known species is first processed; this finds potential coding regions or genes on the genome. Open Reading Frames of the coding regions are predicted to find the translated sequence of amino acids. Homology studies are run to predict similarity of amino acids between species, this gives an idea how the amino acids fold into a protein, and what the functional roles of that protein consist of. [34] uses an established interaction network to produce an interaction network in a new species using sequence alignment.

Once protein homology with high expectation has been discovered, evidence is produced from wetlab experiment. Experimentation is usually classified as either **High-throughput (HTP) analysis** or **low-throughput (LTP) analysis**. HTP analysis discovers interaction between hundreds of proteins, such as Yeast or E. coli-2 Hybrid screening which discovers binding potential between proteins. High throughput analysis is scalable and cheap, but often gives only a first glance impression of interactions ([5]). This is because of the high false positive rate of HTP methods. For example, 2-hybrid screening can have false positive rates as high as 50% [12].

Interactomics are often interested in finding methods to increase the confidence level of protein interactions. We can increase confidence in an interaction by performing complimentary experimentation. Two or more independent experiments compliment each other, increasing the confidence in the described interaction. [6] explains methods to increase confidence in protein networks. Double linkage [23] labels interactions that have higher confidence from complimentary references.

5.1.2 Signal Pathway Networks

As described, Signal Pathways are biological maps of the interactions of proteins and their environment within a specific functional domain. They exist in all species, with subtle differences. For example, both humans and fruit fly have very similar pathways for RNA processing. With the introduction of new model species, we can immediately make predictions about their proteins and pathways. Vice versa, it is cheaper to find experimental evidence on model species, and then predict how this evidence will affect human pathways.

Reactome ([57]) and Kyoto Encyclopedia of Genes and Genomes (KEGG - [21]) are two signal pathway networks. They contain functional representations of pathways for various species, including human, mouse, worm and fruit fly. Both networks store these biological networks in ontologies, mainly the BioPax format.

BioPax (http://www.biopax.org/index.html) is a superset of OWL. BioPax elements are described using one of four main classes: **Entity**, **PhysicalEntity**, **Interaction** and **Pathway**. These superclasses are described by Figure 5.1.

- **Entity**: This is the root class of the BioPax class structure (schema). This is an abstract class, and no individuals exist of type Entity. It exists only to relate its subclasses. All classes inherit the attributes of Entity, which include *NAME*, *SHORT-NAME*, *COMMENT* and *REFERENCE*.

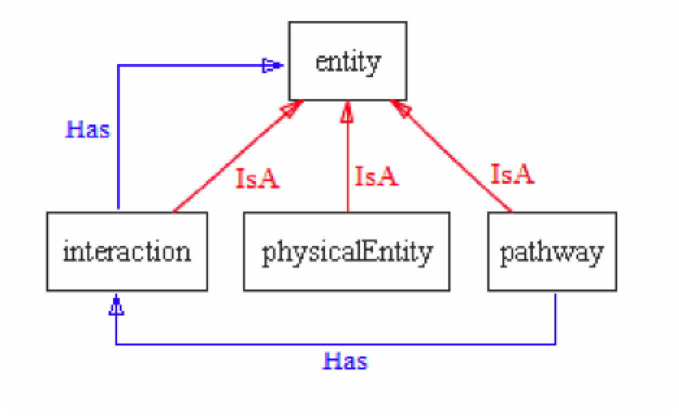

Figure 5.1: Describes the superclass relationships. Everything is an **Entity** (root of the graph). **PhysicalEntity** describes physical biological units. **Interactions** describe how physical entities interact. **Pathways** describe collections of interactions and their overlaying function. There are various subclasses for interactions and physical entities.

- **PhysicalEntity**: Physical entities are biological units that can be 'seen'. They include DNA, RNA and proteins. Researchers are most interested in the function of physical entities, how they react, interact and how they can be created and destroyed.

- **Interaction**: Interactions consist of two parts, the *Relationships* and *Participants*. A relationship describes how the participants react to each other. In other words, interactions describe the relationships that are known between any number of participants. Each interaction is supported by one or more pieces of *EVIDENCE*. Some examples of interactions include 3D protein-protein interactions, and enzyme catalysis.

- **Pathway**: Pathways represent collections of interactions with a common function or goal. For example, RNA processing and glycolysis are both pathways.

5.1.3 Related Biological Networks

While protein networks provide an excellent source for interacome maps, there are many other sources for protein interactions with large sets of proteins, but the interactions are less confident and characterized. BIND [4], IntAct [24] and DIP [48] are large protein-protein interaction databases. Clusters of Orthologous Groups (COG - [55]) clusters groups of proteins based on their homologous evidence. Molecular interactions based on genomic information is characterized in Genomic Interaction networks (GI networks).

5.1.4 Theory and Reality of using Gene Ontology as Annotation for Protein Function

While biological networks are one way to describe protein function, there have been several attempts to describe protein function using protein annotation ([20], [40]). The Gene Ontology (GO - [1]) is a domain ontology that describes molecular function elements that can be referenced by proteins. These functional annotations can be used to further describe the function of the protein and its role in pathways. Functional annotation can also be used to find *de novo* function [54]. [47] describes methods to functionally annotate proteins for protein networks. Efforts have been made to cluster and classify proteins in protein-protein networks [7].

However, discovering *de novo* protein function is difficult from protein annotation - such as GO annotation. This is because (i) protein annotation is inconsistent between proteins and (ii) the domain ontology itself is incomplete. Protein annotation is inconsistent because proteins are individually annotated by humans, and humans have different views of the meaning of some GO annotations. GO is constantly changing as new annotations need to be added from further experimentation. That means that new protein functions are discovered all the time.

GO is an ontology described by the Open Biomedical Ontologies (OBO - http://obofoundry.org/), and is organized as a Directed Acyclic Graph (DAG), which means that it is hierarchical except that nodes of the graph can have multiple parents. Edges of the graph describe the parent to child relationships that are either *is_a* or *part_of* relationships. Nodes of the graph describe GO terms, which can be used as annotation for proteins and can describe either their *molecular function*, *cellular component* or *biological process*:

- Molecular Function: Descsribes functions of a gene in intercellular and intracellular activities.
- Cellular Component: Subcellular and macromolecular localities where genes can be expressed.

- Biological Process: Recognized processes that describe a series of events or functions.

5.1.5 Specificity and Significance of GO Terms

Not all GO terms are equal - GO terms that are more specific better describe the role of the molecule. Child nodes are always more specific than their parents, as the meaning of *is_a* and *part_of* relationships describe. In general, child node that are farther away from the root tend to become more specific. We can use the height of the GO term (maximum distance from root or minimum distance to any leaf) to describe the specificity of the term. This height gives only an approximation of the specificity of the term, since i) not all sibling nodes (two GO terms with the same parent term) are of equal specificity, ii) the total height of the ontology is not consistent and iii) because of the DAG construction some terms have multiple parents and thus several heights and several meanings. Most GO terms at a similar height tend to have similar specificity [13].

For a given protein with GO metadata, not all of the GO annotation plays as heavy a role in its function. This amount is influenced by the specificity of the GO term - the amount we describe as the significance of the GO term to the protein's overall function. The sum of the significance of all GO terms describes the GO term function.

Chapter 6

Evaluation

We use two protein networks in ontology formats that have annotation from a domain ontology as input to our method. In doing so, we demonstrate our ability to map proteins between these two protein ontology networks. Then we show how it compares to conventional protein mapping methods. We also show how this compares orthogonally to conventional label matching and string matching ontology alignment methods.

This chapter explains how our method can be applied to a sample data set. We show that proteins in two signal pathways can be mapped together based on their metadata from the Gene Ontology (GO [1]). We use simulated annealing (see Section 3.3) to determine the clustering of proteins based on the similarity of their GO terms. Similarity is determined by an *intersection score* described in Section 6.2.1. The clusters are then matched using hierarchical clustering. The results are described in Section 6.2.2 and discussed in Section 6.2.3.

We performed four experiments to show the capabilities of our method to map proteins compared to homology studies that find both paralogs and orthologs (see Section 2.1). [47] and [54] both explain how GO terms can describe function and co-expression of proteins. We assume that proteins that are more homologous tend to have more similar function, and vice versa. Our assumption holds in most cases, however [20] and [49] say that there are some exceptions to this assumption. We show that there exists a correlation between our method's matching an homology's expectation value across human (Homo sapiens) and yeast (Saccharomyces cerevisiae) in general. We also show the same correlation within a species, comparing two human pathways - RNA Capping and RNA Editing pathways. The results are described in Section 6.2.4 and discussed in Section 6.2.5.

We aim to map proteins from yeast to proteins from human. The proteins from human exist within a known protein network, thus the interactions between these proteins are well known. The interactions between the proteins from yeast are not known. What is known (though not necessarily verified from experimentation), is that the proteins being investigated are well conserved, their function is well conserved, and thus the proteins networks that comprise these proteins are conserved. See Figure 6.1.

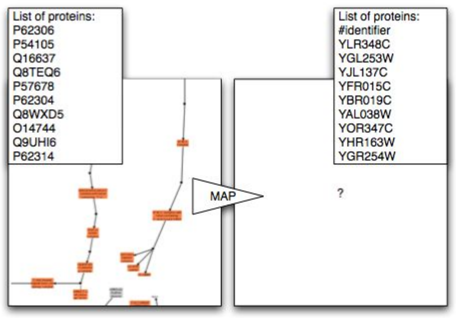

Figure 6.1: We aim to build a mapping between the proteins from human and yeast. The proteins on the left represent human proteins, and the proteins from the right represent yeast proteins. The human proteins have been extensively studied, and the protein pathway is known. The yeast proteins are known to function within the protein pathway, but we do not know the exact pathway. We can build a mapping between the two pathways.

6.1 Data Set

6.1.1 Reactome.org BioPax Pathway Ontologies

Biological pathways are the known protein networks that we have chosen to work with. Reactome (www.reactome.org - [57]) is a repository for biological pathways, which can be downloaded in a number of formats including BioPax (http://www.biopax.org/). BioPax is convenient because it defines the role of proteins within a sub-cellular context - thus BioPax let us easily extract the necessary proteins, and their known annotation, for each pathway. We use the following 4 biological pathways for evaluation:

1. Homo sapiens mRNA Capping: A sub pathway of mRNA processing. The pathway contains 269 molecules, 29 of which are proteins.

2. Homo sapiens Metabolism of ncRNA: A metabolism pathway that detects and metabolizes RNA not involved in protein coding. The pathway contains 271 molecules, 23 of which are proteins.

3. Saccharomyces cerevisiae mRNA Capping: A sub pathway of mRNA processing. The pathway contains 234 molecules, 26 of which are proteins.

4. Saccharomyces cerevisiae mRNA processing: The pathway contains 527 molecules, 47 of which are proteins.

These pathways were selected because 1) they exist in multiple species and 2) the known interactions are well validated by experimentation. Also, the pathways are similar in function (they work on RNA), but do not necessarily contain similar or homologous proteins - which could imply that they have several similar functional protein groups. They are also of similar size, both in number of molecules and proteins.

6.1.2 Gene Ontology (GO) Domain Taxonomy

All the proteins from the Reactome pathways selected contain at least one GO term cross reference (annotation) with only a couple of exceptions (no more than one protein per pathway). The GO release that we downloaded contained 23911 GO terms, 1094 were obsolete.

We used all three of the GO sub-domains (described in Section 5.1.4) to ascertain the function of a protein in a network ([54]). We use the GO terms under *Molecular Function* and *Biological Process* sub-domains describe the function of the proteins, while the *Cellular Component* sub-domain can be used for the functional locality of the protein.

6.1.3 Mapping between Protein Networks and GO

There are two challenges to difficulties to consider when using the current mapping of protein annotation to the GO network. The first arises because protein annotation is rarely updated when a new GO version is released. The second problem stems from the fact that many proteins are annotated not by scientific evidence, but by prediction studies.

If the annotation is not updated on a consistent basis, then the annotation might not be specific enough. For example, hemoglobin subunit alpha (name: HBA_HUMAN, swiss-prot accession number: P69905) should have the GO annotation *Molecular function: hemoglobin binding* (GO identifier: GO:0030492) instead of just *Molecular function: protein binding* similar to the hemoglobin subunit beta (name: HBB_HUMAN, swiss-prot accession number: P68871). We have developed a workaround for maintaining specificity that is detailed in Section 5.1.5, although maintaining an updated mapping is more accurate. Due to the definition of the relationships IS_A and PART_OF, every child node is more specific than its parent. One way to compare specificity is to notice that terms farther from the root 'tend' to be more specific. That is to say, the distance from the root node (or depth) of a node provides a rough estimate of its specificity. It is a rough estimate because specificity is not consistent between the parent-child relations. Furthermore, since one term can have multiple parents we make the assumption that the specificity of a GO term is the shortest path to the root.

The vast majority of protein annotation, in terms of GO terms, is provided by one of the following two means: data mining of PubMed texts or through homology studies to related proteins that already have GO annotation:

1. PubMed text data mining looks through paper text bodies or abstracts for key words that match GO terms ([37] [27], [9]) . This method provides an independent means of selecting GO annotation - but is highly dependent on the language of the article (i.e. the presence of key words), the article's availability, and data mining algorithm used. Since papers lack a consistent structure and key work usage, this method can contain a lot of error, which can be corrected using [18].
2. A protein that contains little or no GO annotation that is highly homologous to another protein with GO annotation, can be predicted to have the same GO annotation (see Homology in Section 2.1). This method is even less reliable than the previous method. Error is compounded. Regardless, this method is often used to predict GO annotation of non-human, model species proteins.

The large remainder of protein GO annotation is curated manually by experts and validated by experimental methods. We assume that this portion of GO annotation is correct, regardless of potential human error.

6.2 Methods

Producing a mapping is a two step process.

We first need to perform preprocessing that clusters proteins into functionally related or similarly co-regulated groups. We estimated the clustering by performing simulated annealing on a dual processor machines. Each data set took approximately 4 hours of simulated annealing to provide 50 simulations, each with 5000 steps.

A frequency matrix of size $|proteins|$ x $|proteins|$ counts the number of proteins that are grouped together. The frequency matrix is used as a distance metric to cluster the proteins.

Secondly, hierarchical clustering is used to cluster the proteins based on the frequency matrix. We are justified in using clusters to perform matching for two reasons:

1. By definition, GO terms do not describe function. Instead, groups of GO functions can describe a complex functions, such as a signal pathway. Therefore, to map proteins based on function, we need to cluster the proteins together and match the combined, or clustered, GO terms.

2. [47] and [54] show that clusters of GO terms can describe relationships between genes and proteins. We used groups of GO terms to describe the context of a cluster of proteins.

6.2.1 Approximating Distance Metric

Hierarchical clustering (see Section 3.1) requires a distance metric, which we approximate by performing simulated annealing (see Section 3.3) on the data set. Simulated annealing is a iterative modeling approach. We are modeling proteins into interacting clusters based on their GO terms. Whether individuals are iteratively merged into a cluster is decided by an *intersection score*, and a break condition (sometimes referred to as the temperature of the system) decides when to reduce the size of clusters. We are confident in the use of our intersection score because it is able to distinguish between known protein networks and randomly generated proteins networks (see Section 6.2.4).

The *intersection score* itself is highly dependent on the number of GO terms in the proteins to be compared. However, the significance of GO terms to the function and context of the proteins is not uniformly distributed (see Section 2.3). We determine the *significance* of the GO terms to proteins before calculating the *intersection score.*

The *significance* of GO terms is the amount of influence a term has to the function it describes. For example, a molecule could be said to *bind with hemoglobin* - this describes its function. However, *bind with hemoglobin subunit alpha* is a more specific description of a molecules function. There are two predominant methods to determine the significance of a term. The first is using the **depth** of the term (i.e. shortest distance to root or a leaf). The second is from the **frequency** of the term in a background set. We have found that using the frequency of the term compared with a background set produces more significant results (results below), and so have used a version of **frequency** to measure the significance of the terms called *pushTermsToLeaves*.

The major drawback to calculating the significance of GO terms, is that the Gene Ontology does not have a uniform depth. This is because GO terms are not added to GO in a uniform manner. To bypass this problem, we have decided to score GO matches at the leaves only. Since children nodes of a GO ontology are more specific descriptions than their parents, GO term matches to internal nodes are assumed to be simply less specific matches to the descendant leaves. The pushTermsToLeaves method (Figure 6.2.1) scores GO terms in that manner.

The *intersection* score is a formula that is based on the number of proteins p and the number of terms, belonging to these proteins t that intersect (are equal). We can determine this naively by a sum-of-squares function:

$$\sum_{t \in T} (\sum_{p \in P} A(p,t))^2 / |T|$$

T : Set of GO Terms of the given proteins.

P : Set of given proteins.

$A(p,t) = 1$ if protein p is annotated by GO term t, 0 otherwise.

There are two issues with this naive function: (i) The number of intersections does not take into account relationships, and (ii) the significance of the GO terms is not calculated into the intersection.

In a hierarchical-like graph, each parent node of the graph is related to its children as a class to its sub class children. For a completed graph (where no new elements will ever be added), we can say that a parent is described by the combination of its children. With the PART_OF relationship, a parent contains components. These components are the parts, or children, of the parent. With the IS_A relationship, a parent can only be described in a more specific manner by one of its children. We assume that *a parent can be described fully by its children* - which is true by description if the graph is completed and fully describes the domain elements. For example, hemoglobin is fully described by its parts: subunit alpha and beta. If either of these subunits are missing in the ontology, then hemoglobin would be incorrectly described by the remainder. We take a simplified approach and assume that a parent is *equally* described by each of its children. We use a *pushTermsToLeaves* method (Figure 6.2.1) to find intersections in a less naive way, and (to a lesser extent) intersections between parent-children nodes.

If are using depth (shortest distance of node to leaf) as the level of significance of terms. In Figure 6.3, we show how two data sets, A and B,

```
PushTermsToLeaves (Proteins, Terms)
input: set of proteins, a set of terms, and a mapping of which
       proteins are annotated by which terms
output: a set of terms that have no children, weighted.
n <- ();
For each p in Proteins
      for each t in Terms[P]
            add (n, Divide(t,1));

Divide(term, weight)
n <- ();
if term is a leaf
      add (n, (term, weight));
else
      numC <- |Children(t)|;
      for each c in Children(t)
      add (n, Divide(c, weight/numC));
return n;
```

Figure 6.2: Push Terms to Leaves method: Gives less weight to matches with internal nodes than to matches with leaf nodes. This accommodates the observation that internal nodes are less specific than leaf nodes, and thus divides their intersections amongst the children nodes.

would get scored naively. Since there are no intersections, the score would be $(1 + 1 + 1)/3 = 1$. Using the pushTermsToLeaves method, Figure 6.4 would divide the intersection (uniformly) at the parent to the two children, giving each child an extra $1.5^2 + 1.5^2/3 = 1.5$. Figures 6.5 and 6.6 given a larger example. The final intersection score would be given by:

$$\sum_{t \in T} (\sum_{p \in P} w(p,t))^2 / |T|$$

T : Set of GO Terms of the given proteins.

P : Set of given proteins.

$w(p,t) :=$ the weight of the GO term annotated by the protein returned from the *PushTermsToLeaves* algorithm.

We examined how our intersection score affects protein clusters. A higher score means that the cluster has a common function described by the intersection of GO terms. We took a sample of proteins that are known to have a common function (protein network pathways) and compared the intersection scores with random clusters of proteins (with random chance of having a common function). With the exception of small protein clusters (of size ≤ 4), we observe a significant difference between the intersection score produced from known protein clusters and those produced from random protein clustres. The results are shown in Figures 6.7 and 6.8.

6.2.2 Clustering Proteins using Hierarchical Clustering

We now show how to cluster proteins based on the similarity of their GO terms. We have already established a distance metric that calculates the similarity of pairs of proteins, and we use this distance metric for clustering.

Hierarchical clustering is a simple and efficient method to cluster elements without knowing beforehand the size of the clusters (further details in Section 3.1). We used a complete method that tries to assemble nice centroid-like clusters. The clusters are built in a tree like manner, relative to the distance given by the distance metric.

The leaves of the hierarchical clustering graphs are individual proteins, and the height of connecting branches is the distance determined between proteins. These methods are unsupervised, and thus a cut off can be determined for each of these graphs to determine the number of clusters. We chose a cut off of 40 for mRNA Capping pathways, 50 for the Metabolism of ncRNA pathway, and 55 for mRNA Processing pathway proteins. These cut offs were appropriate to reach a consensus cluster size for each. We leave further investigation on the impact of cluster size to the final result

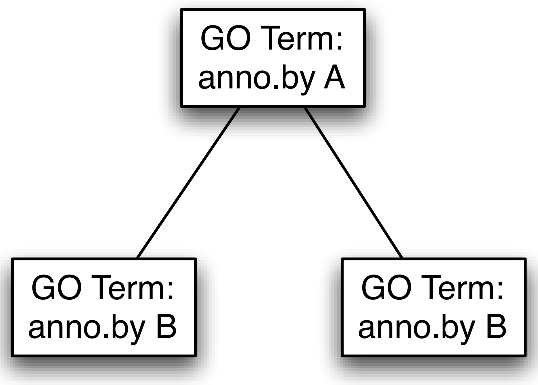

Figure 6.3: The initial result of calculating the GO terms of proteins A and B onto a DAG, where A is annotated by a GO term that is the parent of two GO terms that B is annotated by. The naive method would result in a intersection score of 1.

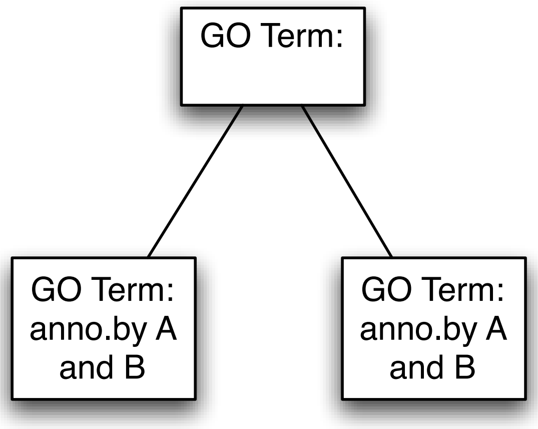

Figure 6.4: The *PushTermsToLeaves* would result in two 1.5 intersections, summing to a total intersection score of $1.5^2 + 1.5^2/2 = 1.5$ (using intersection score without significance).

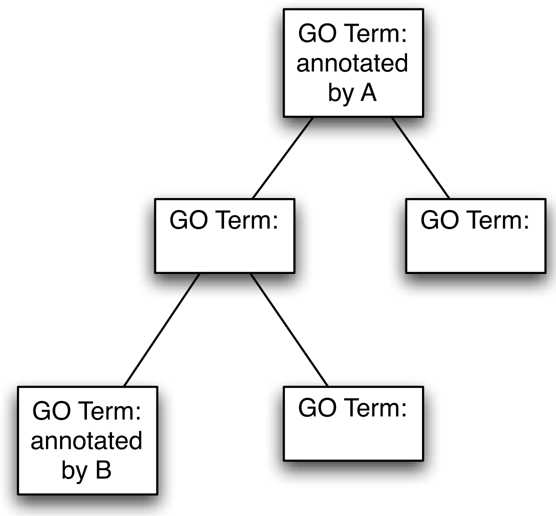

Figure 6.5: the initial results of calculating the GO terms of proteins A and B into a DAG, where A is annotated by a GO term that is the grandparent of a GO term that B annotates. The naive method would result in a intersection score of $2/5 = 0.4$.

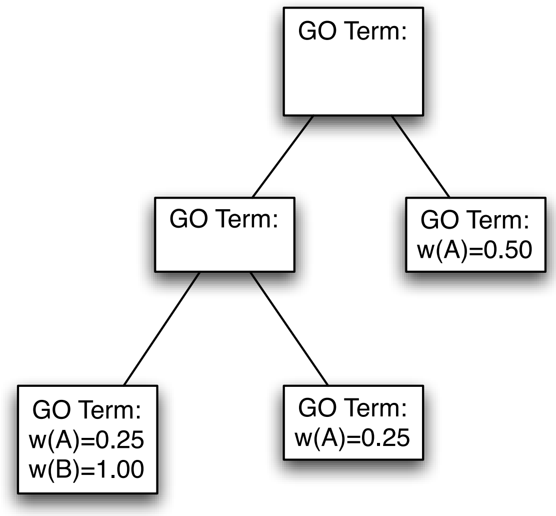

Figure 6.6: The *PushTermsToLeaves* would result in one 1.25^2 intersection, one 0.25^2 intersection and one 0.5^2 intersection, summing to a total intersection score of $(1.25^2 + 0.25^2 + 0.5^2)/3 = 1.875$.

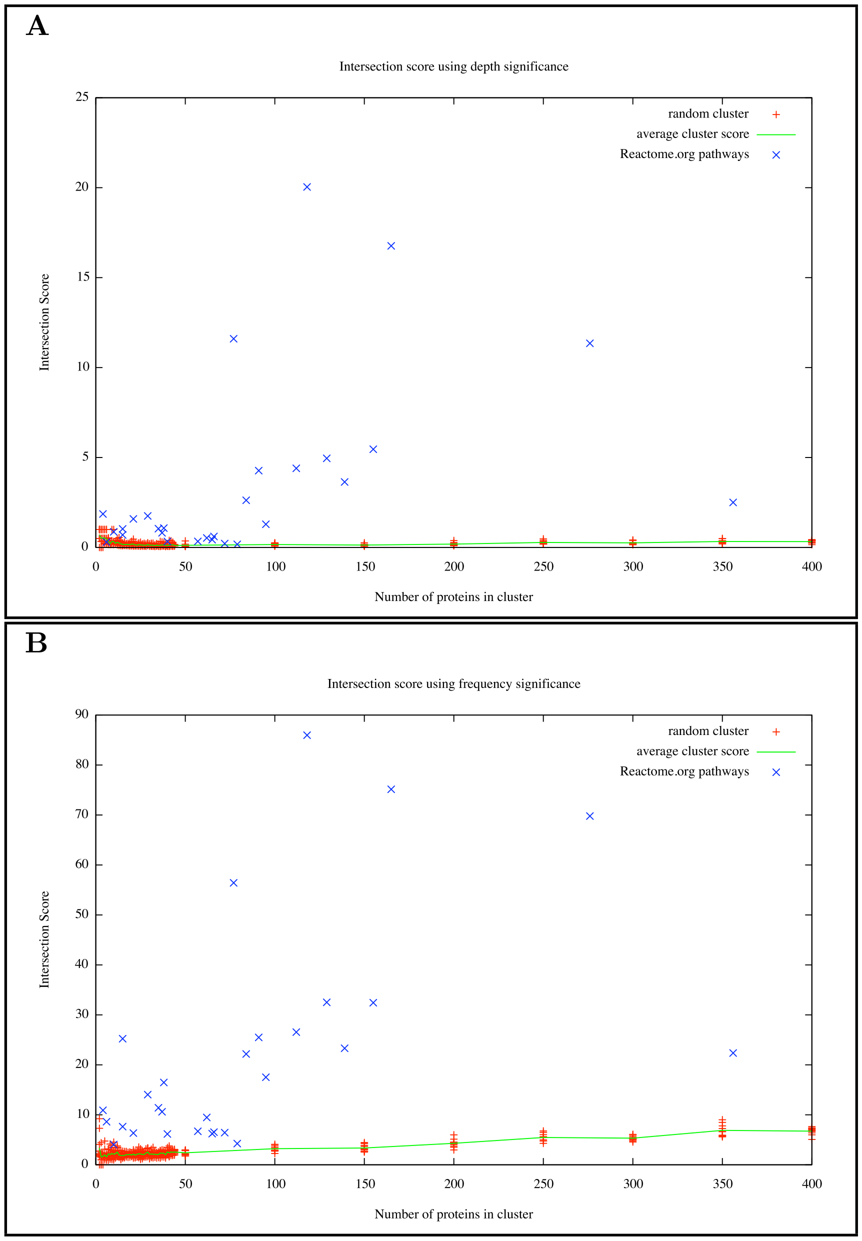

Figure 6.7: Shows the difference in intersection scores between a cluster of random proteins with little to no related GO annotation and given protein pathways from reactome.org. The protein pathways have some common GO annotation related to the pathway's function. *A* uses the distance from the root node to show the significance of a GO term. *B* uses the frequency of the GO term in a background set of all known human proteins to determine the significance of the GO term

.

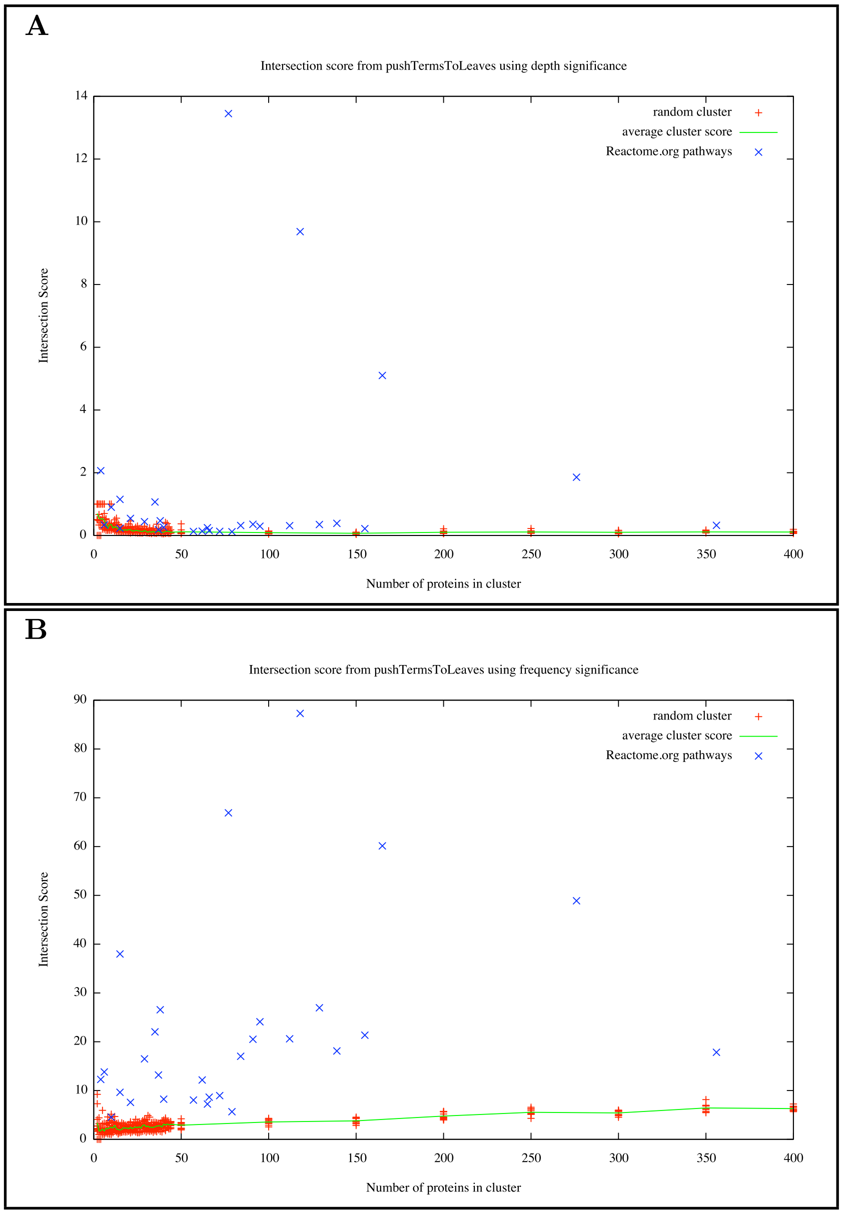

Figure 6.8: Similarly to Figure 6.7, this shows the difference in intersection scores between a cluster of random proteins with little to no related GO annotation and given protein pathways from reactome.org. However, we used the *pushDownToLeaves* method instead of the naive method. The protein pathways have some common GO annotation related to the pathway's function. *A* uses the distance from the root node to show the significance of a GO term. *B* uses the frequency of the GO term in a background set of all known human proteins to determine the significance of the GO term

.

for future work. The resulting hierarchical trees can be see in Figures 6.9, 6.10, 6.11 and 6.12.

6.2.3 Discussion: Intra Protein Network Matching

We have produced clusters of proteins under common contexts. By context, we mean, the combination of the significant GO terms that are shared by the proteins of the cluster. This context is represented by an ontology. This context is a sub-ontology of the Domain Ontology (that is, the sub-ontology is a sub-set of the elements and relationships of the Domain Ontology). An intersection score for each cluster is calculated depending on the fraction of proteins within a cluster that are described by the context. We note that for clusters with the high intersection scores the GO terms are well conserved between proteins. In other words, the context of the cluster is well defined and shared by a majority of the proteins. Tables 6.1, 6.2, 6.3 and 6.4 show the clusters with the two highest intersection scores and the GO terms that are defined by the clusters' context.

Both of the clusters of the pathway Metabolism of ncRNA in Table 6.1 involves many proteins from involved in *spliceosome assembly* and *protein binding*. These are functions similar to the degradation of RNA molecules. One could assume that the reason both clusters are not unionized into one super-cluster, is because the algorithm contains randomized bits.

We notice that the clusters within the mRNA Capping pathway from Tables 6.2 and 6.3 for clusters with function for *DNA repairing*, *DNA transcription* and *DNA binding*. This is a strong indication that these proteins play a big role in the mRNA Capping process and that this process is conserved between humans and yeast.

Table 6.4 shows two clusters involved in RNA transcription. Cluster 1 contains many terms of regulation of polymerase II for RNA transcription: holo TFIHH complex is part of the transcription binding complex of polymerase II in yeast and nucleotide-excision repair factor 3 complex is related to the TFIHH complex. Cluster 2 contains proteins related to more general RNA transcription in polymerase I-III.

We conclude that at least the top produced clusters are accurate and relevant to their respective protein networks. We leave the analysis of the smaller, lower scoring clusters, to future work. This could be done by comparing the results against results from known data sources such as protein-protein interactions, signal pathways or genomic interactions. This could also be used to find more accurate cluster sizes, and hierarchical cluster cut-off thresholds.

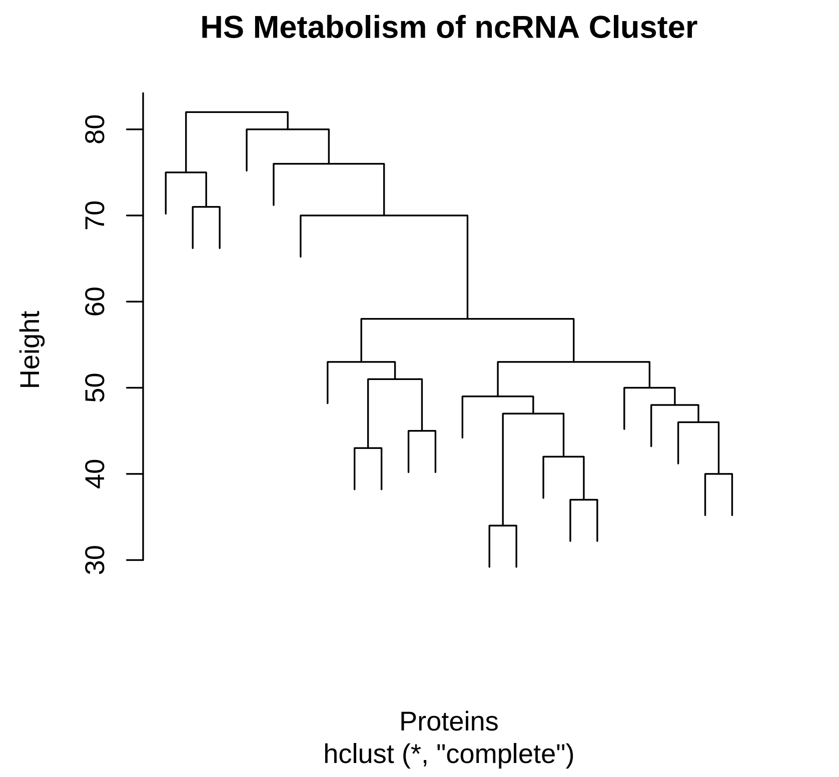

Figure 6.9: A hierarchical clustering of the proteins in the Homo sapiens Metabolism of non-coding RNA pathway from reactome.org. The result clusters into three larger clusters, with 8 smaller outlying clusters.

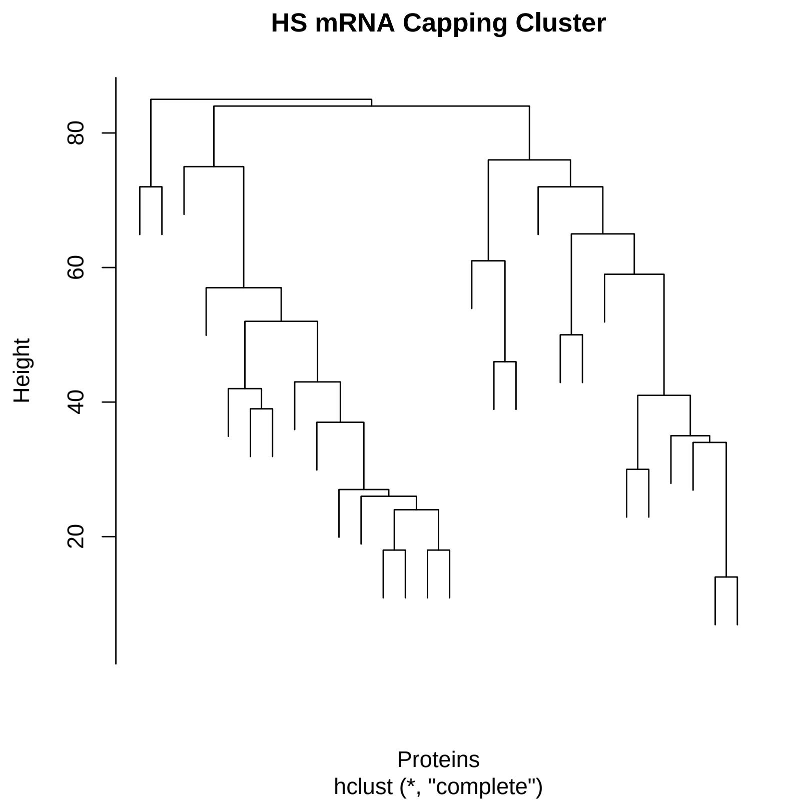

Figure 6.10: A hierarchical clustering of the Homo sapiens mRNA Capping pathway from reactome.org. The result clusters into one large cluster, with 10 smaller outlying clusters.

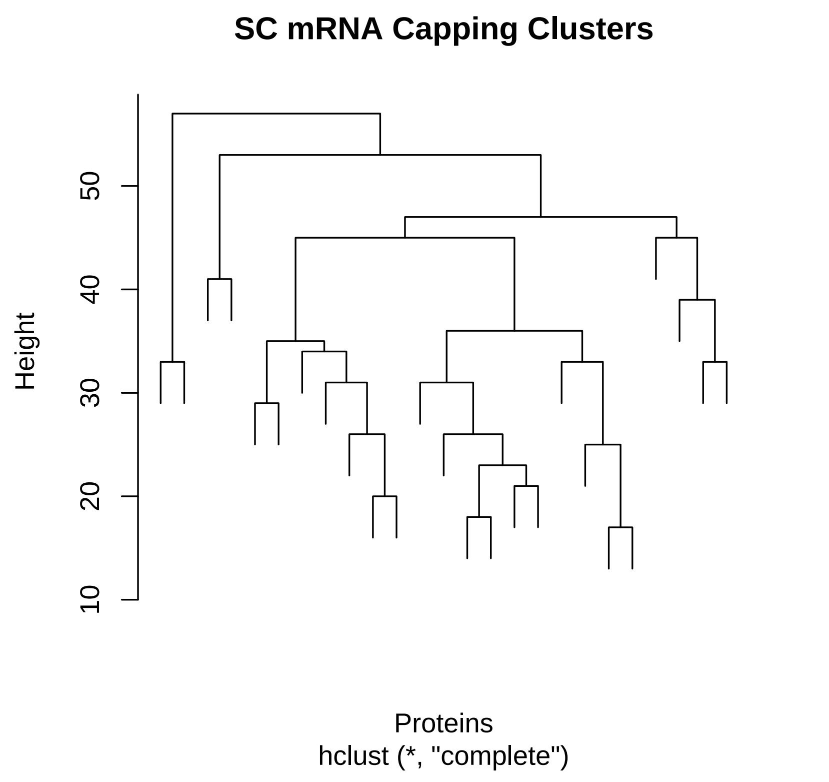

Figure 6.11: A hierarchical clustering of the yeast mRNA Capping pathway from reactome.org. The result clusters into 7 clusters.

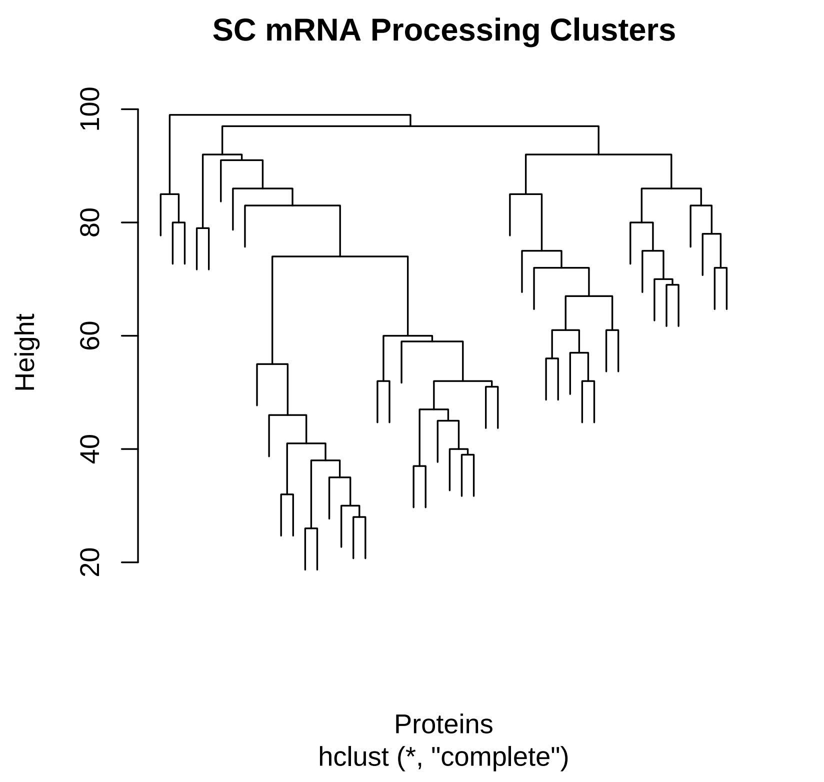

Figure 6.12: A hierarchical clustering of the yeast mRNA Processing pathway from reactome.org. The result clusters into 2 larger clusters and many loose clusters (cut off at height 55).

Proteins: P62316, O14893, P62318, P62306, P62304, P52298		
GO term	score received	Description
GO:0000245	18.0	biological process: spliceosome assembly
GO:0005515	18.0	molecular function: protein binding
GO:0005681	16.5	cellular component: spliceosome
GO:0030532	13.5	cellular component: small nuclear ribonucleoprotein complex
Proteins: P62314, P62308, P14678, Q16637, P57678		
GO term	score received	Description
GO:0000245	12.0	biological process: spliceosome assembly
GO:0005515	15.0	molecular function: protein binding
GO:0030532	12.0	cellular component: small nuclear ribonucleoprotein complex
GO:0005681	6.0	cellular component: spliceosome
GO:0005737	6.0	cellular component: cytoplasm

Table 6.1: The two best scoring clusters and their most commonly shared GO terms produced from proteins of *Homo sapiens* network Metabolism of ncRNA.

Proteins: Q92759, Q13889, P32780, P51948, P18074, P19447		
GO term	score received	Description
GO:0005675	15.0	cellular component: holo TFIIH complex
GO:0006357	15.0	biological process: regulation of transcription from RNA polymerase II promoter
GO:0006289	12.0	biological process: nucleotide-excision repair
GO:0000079	6.0	biological process: regulation of cyclin-dependent protein kinase activity
GO:0003684	6.0	molecular function: damaged DNA binding
GO:0003700	6.0	molecular function: transcription factor activity
GO:0005515	6.0	molecular function: protein binding
GO:0006283	6.0	biological process: transcription-coupled nucleotide-excision repair
GO:0006917	6.0	molecular function: induction of apoptosis
GO:0008094	6.0	biological process: DNA-dependent ATPase activity

Proteins: O15514, P24928, P52435, P62875, P19388, P30876, P52434, P62487		
GO term	score received	Description
GO:0006366	21.375	biological process: transcription from RNA polymerase II promoter
GO:0005665	19.875	cellular component: DNA-directed RNA polymerase II, core complex
GO:0003899	18.0	molecular function: DNA-directed RNA polymerase activity
GO:0005515	15.0	molecular function: protein binding
GO:0006356	8.25	biological process: regulation of transcription from RNA polymerase I promoter
GO:0003677	6.0	molecular function: DNA binding
GO:0005730	4.125	cellular component: nucleolus

Table 6.2: Shows the largest clusters and their most commonly shared GO terms produced from proteins of *Homo sapiens* network mRNA Capping.

Proteins: P06242, Q03290, P37366, P41896, Q12004, P32776, P06839, Q04673, Q00578, Q02939		
GO term	score received	Description
GO:0006367	30.0	biological process: transcription initiation from RNA polymerase II promoter
GO:0005675	27.0	cellular component: holo TFIIH complex
GO:0016251	27.0	molecular function: general RNA polymerase II transcription factor activity
GO:0007070	24.0	biological process: negative regulation of transcription from RNA polymerase II promoter, mitotic
GO:0000112	18.0	cellular component: nucleotide-excision repair factor 3 complex
GO:0000717	18.0	biological process: nucleotide-excision repair, DNA duplex unwinding
GO:0005515	15.0	molecular function: protein binding
GO:0003678	6.0	molecular function: DNA helicase activity

Proteins: P20435, P20436, P08518, P27999, P38902, P22139, P40422		
GO term	score received	Description
GO:0005665	21.0	cellular component: DNA-directed RNA polymerase II, core complex
GO:0006366	21.0	biological process: transcription from RNA polymerase II promoter
GO:0003899	21.0	molecular function: DNA-directed RNA polymerase activity
GO:0005666	12.0	cellular component: DNA-directed RNA polymerase III complex
GO:0005736	12.0	cellular component: DNA-directed RNA polymerase I complex
GO:0005515	9.0	molecular function: protein binding

Table 6.3: Shows the largest clusters and their most commonly shared GO terms produced from proteins of *Saccharomyces cerevisiae* network mRNA Capping.

Proteins: P06242, P37366, Q12004, P06839, Q00578, Q00578, Q03290, P32776, Q04673, Q02939		
GO term	score received	Description
GO:0005675	27.0	cellular component: holo TFIIH complex
GO:0006367	27.0	biological process: transcription initiation from RNA polymerase II promoter
GO:0007070	24.0	biological process: negative regulation of transcription from RNA polymerase II promoter, mitotic
GO:0016251	24.0	molecular function: general RNA polymerase II transcription factor activity
GO:0000112	18.0	cellular component: nucleotide-excision repair factor 3 complex
GO:0000717	18.0	biological process: nucleotide-excision repair, DNA duplex unwinding
GO:0005515	15.0	molecular function: protein binding
GO:0003678	6.0	molecular function: DNA helicase activity

Proteins: P08518, P38902, P20434, P40422, P20435, P22139		
GO term	score received	Description
GO:0006366	18.0	biological process: transcription from RNA polymerase II promoter
GO:0003899	18.0	molecular function: DNA-directed RNA polymerase activity
GO:0005665	18.0	cellular component: DNA-directed RNA polymerase II, core complex
GO:0005666	12.0	cellular component: DNA-directed RNA polymerase III complex
GO:0005736	12.0	cellular component: DNA-directed RNA polymerase I complex
GO:0005515	6.0	molecular function: protein binding

Table 6.4: Shows the largest clusters and their most commonly shared GO terms produced from proteins of *Saccharomyces cerevisiae* network mRNA Processing.

6.2.4 Matching Proteins across Models

Now that proteins are clustered into functional groupings, we can apply a matching solution to find which individual proteins are most similar. For a fully automated approach, we use the Hungarian Algorithm [38] to highlight the best matches. Conventional ontology alignment programs compare individual proteins and output confidence weights for each match. For comparative purposes, and we do the same.

The weights of the confidence between two proteins (from opposite ontologies) are retrieved from the combined intersection score of the belonging clusters. The resulting intersection score is high if the two clusters are more similar, and low if they have no intersection. The intersection scores are then normalized, in the [-1,1] range to easily compare it with conventional ontology alignment algorithms. Confidence scores are roughly computed as follows:

Greater than 0.9: A highly confident score

Greater than 0.5: A loose confidence

Greater than 0: Limited evidence to show similarity

Less than 0: Confident that items are not related

Since protein homology comparison is a *de facto* method for comparing similar proteins, we have used this as our accuracy measure. Cluster matching scores are produced using our method described here and Harmony label matching, and both are compared to protein homology. Homology compares the raw amino acid sequences using **blastp**
(from http://www.swbic.org/origin/proc_man/Blast/tutorial_blastp.html).

We perform analysis on 4 signal pathways from Reactome.org. Each of these sets

In our analysis, we performed 4 comparisons:

1. **Crossing human pathways**: We compared the results between the pathways mRNA (messenger RNA) Capping and Metabolism of ncRNA (non-coding RNA) to find potential paralogs. This test demonstrates intra-species similarity scoring in humans. We expect to get a few common proteins, but the majority should fall into non-matching functional groups. This is because both pathways require binding to RNA molecules, and we would expect the GO terms to represent this function. Figure 6.13 shows the results using Harmony label matching. As

expected, a vast majority of the results are impossible to interpret. Figure 6.14 shows the intersection scores using our method compared with alignment using protein homology. The result shows a lot of proteins with high confidence, that are also well conserved (from protein homology).

2. **Crossing yeast pathways**: We compared the results between the pathways mRNA (messenger RNA) Capping and mRNA (messenger RNA) Processing to find potential paralogs in yeast. This test demonstrates intra-species similarity scoring in yeast. We expect to get a very high set of matches because the signal pathways for mRNA Capping and mRNA Processing have very similar function. Actually, the majority of proteins involved are either identical, or have very similar function. Because of the high number of identical proteins involved, and the identical labels used for these proteins, Harmony does not perform in an unintelligible manner. Figure 6.15 shows that Harmony finds proteins with identical labels. Figure 6.16 shows the similarity between proteins based on confidence scores using our method. As we expected, the results show high number of proteins matching using both our method and protein homology.

3. **Crossing human and yeast mRNA capping pathway**: For this comparison, we crossed the proteins of two species, homo sapiens and Saccharomyces cerevisiae, using the same signal pathway. The pathways contain the same function, which implies that we can assume that the proteins involved would have the same functional annotation (GO terms) and high protein homology. Figure 6.17 shows the results from Harmony label matching. All the resulting confidence scores are negative, because the labels used for humans and yeast are very different, even for similar proteins. The protein homology results are high, despite being from radically different species. This implies that the protein function should be relatively conserved between human and yeast. Figure 6.18 shows the confidence scores resulting from our method. As one might expect, the proteins fall into two groups. The proteins with negative confidence scores are from groupings with less conserved function. The proteins with high confidence scores are from groups with highly conserved function. The proteins with high confidence likewise have high protein homology scores.

4. **Crossing dissimilar pathways**: The final crossing compares proteins from different pathways and different species. We use proteins

from homo sapiens Metabolism in ncRNA (non-coding RNA) pathway and the Saccharomyces cerevisiae mRNA (messenger RNA) Processing pathway. The two pathways have little in common except RNA molecule binding. Figure 6.19 again shows Harmony confidence scores all below negative. We again conclude that Harmony label matching does not work well for matching proteins across species, because of the unmeaning labels compared. Figure 6.20 shows the results of confidence scores using our method. As with the previous crossing, we see a few proteins matched with high confidence. These proteins represent proteins with highly conserved function. There are a lot more proteins with negative confidence scoring, which represent bad matches. Both of the groupings have mediocre protein homology scores.

6.2.5 Discussion: Inter Protein Network Matching

We are finally concerned with finding matching proteins between separate networks. There are several species models with known protein function, such as human, yeast, mouse and chicken. Although there are many proteins in these species that either have no known protein function or the full functionality is yet to be discovered, there are several protein pathways that are sufficiently studied. In comparison, we have no knowledge of many other species. It is extremely useful to be able to compare proteins of multiple, different species. A fast, accurate method to estimate the functionality of proteins and their networks great speeds up the collection of evidence for protein comparison.

Traditionally, protein network comparison has been done using protein homology, or using general ontology alignment methods. We discuss the differences between the results found by the Harmony label matching, protein homology and our clustering method. We show that our method can significantly increase evidence for protein network comparisons if only a small amount of information is known about both species.

6.2.6 Label Matching by Harmony

Harmony [36] uses a dictionary of common English language terms to find similarity within string sentences. Without English words, the matcher finds substring occurrences between the matching strings. We are performing matching between the protein identifiers and protein descriptions. By hypothesis, we presume that there is no information in the protein identifier to match proteins across species. Too many of the proteins do not have descriptions, so we expect little chance of finding any relevant matching between proteins by Harmony.

In the instance where proteins are identified by their protein identifier, such as *P62316* where is little to no relation between the function and the identifier, label matching is useless. Instead we need to investigate the similarity between the protein's descriptions (if any). Our current database optionally contains a description element that is used to in combination with the label, to determine the similarity between protein elements. One way to increase the potential of Harmony, or any string matching, is to provide an article describing the protein result. All proteins extracted have a reference to literature. We could have provided the article. However, the time constraints involved in the text data mining process would have been extensive.

The resulting matching is not very meaningful, especially for proteins with very different labels. In Tables 6.17 and 6.19, labels from yeast proteins are compared with labels from human proteins. The yeast proteins are referenced by their locuses, while the human proteins are not. The resulting confidence scores remain around 0 which is as expected.

The only result with greater then 0 confidence was from matching labels within yeast. This is because the two signal pathways contained identical proteins, and thus identical labels. The resulting matches are not paralogs, but identical proteins.

6.2.7 Ontology Aligning Using Domain Ontology

As shown thus far, we provide an alternate ontology alignment method that allows for annotation that has structured relationships. We tested the method on a protein network data set, to see if the method provides meaningful relationships between the proteins. Now we analyze the results, to see what meaningful relationships were discovered.

The data sets consist of 2 human and 2 yeast protein networks from reactome.org. We will first analyze the clusters created from each individual set as described in Section 6.2.2. Then we will analyze the confidence scores created form intermodel matching from Section 6.2.4 compared with those produced from blastp (protein homology) and Harmony.

Traditionally, protein network comparison has been done using protein homology, or using general ontology alignment methods. We discuss the differences between the results found by the Harmony label matching, protein homology and our clustering method. We show that our method can significantly increase evidence for protein network comparisons if only a small amount of information is known about both species. Tables 6.5, 6.6, 6.7 and 6.8 show some of the highest scoring results from our method. We see that most of these pairwise comparisons also have high homology confidence scores. Upon analysis of the results with low homology, we notice that these proteins are described to have similar roles in the literature (such as Q9H814 and P62487 from Table 6.5, which are both synthesized in the nucleus and play a role in RNA binding, and form a complex with other proteins when involved with RNA).

6.3 Conclusion of Results

It has long been established that protein homology is the *de facto* method for proteins with similar function. However, every paper that investigates the method finds exceptions to this rule. We provide a method to alternatively compare how well proteins match based on their GO terms.

Our method subjectively shows excellent potential to use existing structured protein annotation to supplement existing homology matching methods. We have shown that our method is sufficient orthologous to sequence alignment protein homology methods. But by combining the two methods, we can discover homologs between proteins with even greater accuracy. We've shown that our method works equally well for finding both paralogs and orthologs. In comparison, Harmony only works for matching identical paralogs.

We also discover in course of our method, that we develop contexts for each protein. These contexts can be used to reduce ambiguous or even synonymous functions. We show that our method works better for non-random sets, then for random sets of proteins. There are no other methods that build GO term contexts for comparison.

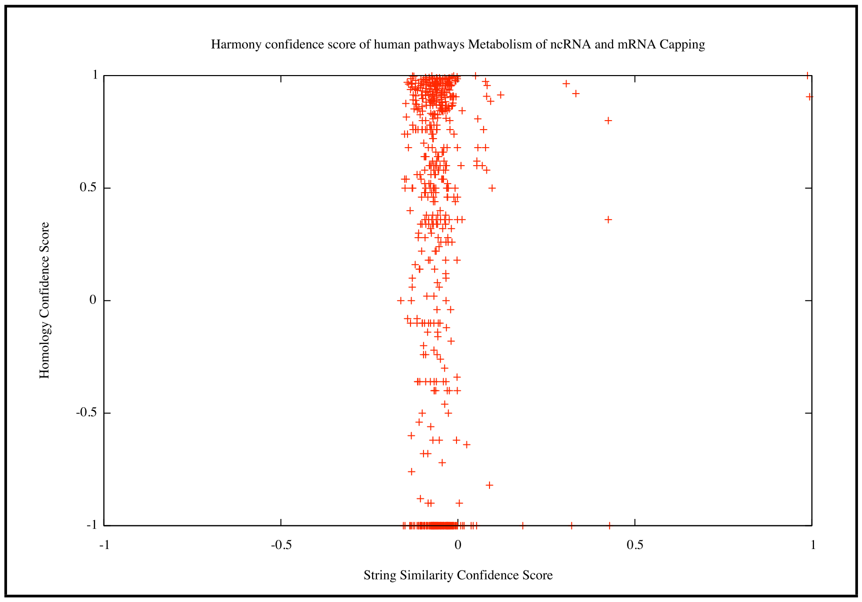

Figure 6.13: Shows the distribution between string similarity confidence score as determined by Harmony label matching and the homology between proteins as determined by BLASTp. Pairs of proteins derive from *homo sapiens* pathways mRNA Capping and Metabolism of ncRNA.

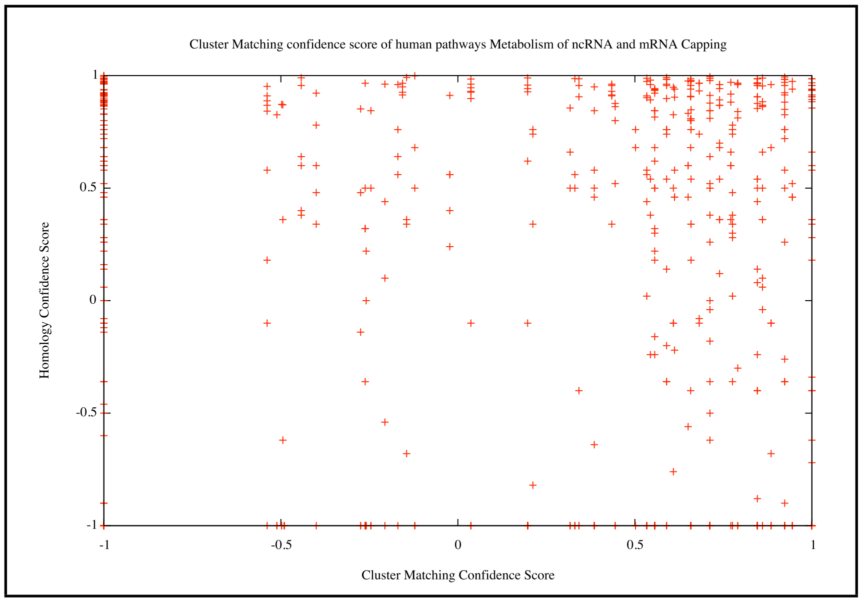

Figure 6.14: Shows the distribution between cluster matching confidence score as determined by our method and the homology between proteins as determined by BLASTp. Pairs of proteins derive from *homo sapiens* pathways mRNA Capping and Metabolism of ncRNA.

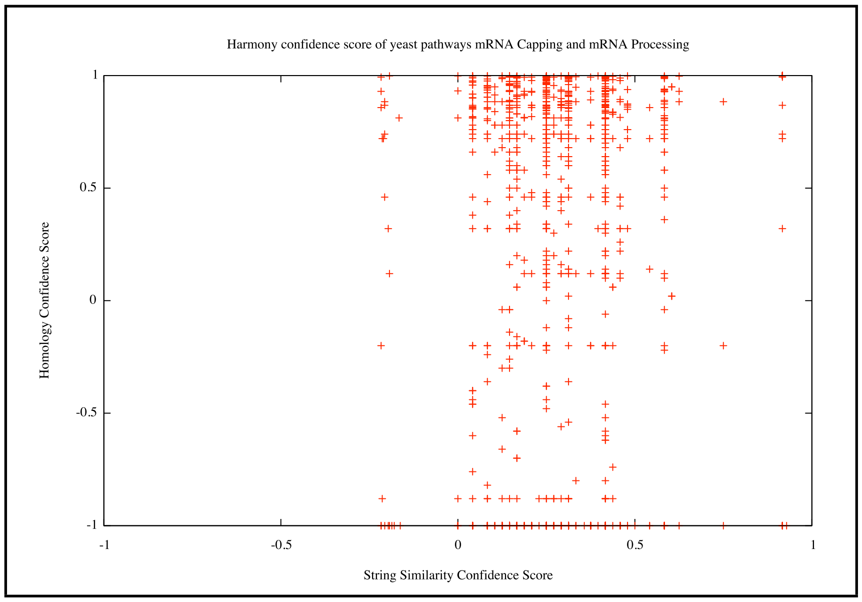

Figure 6.15: Shows the distribution between string similarity confidence score as determined by Harmony label matching and the homology between proteins as determined by BLASTp. Pairs of proteins derive from *Saccharomyces cerevisiae* pathways mRNA Capping and mRNA Processes.

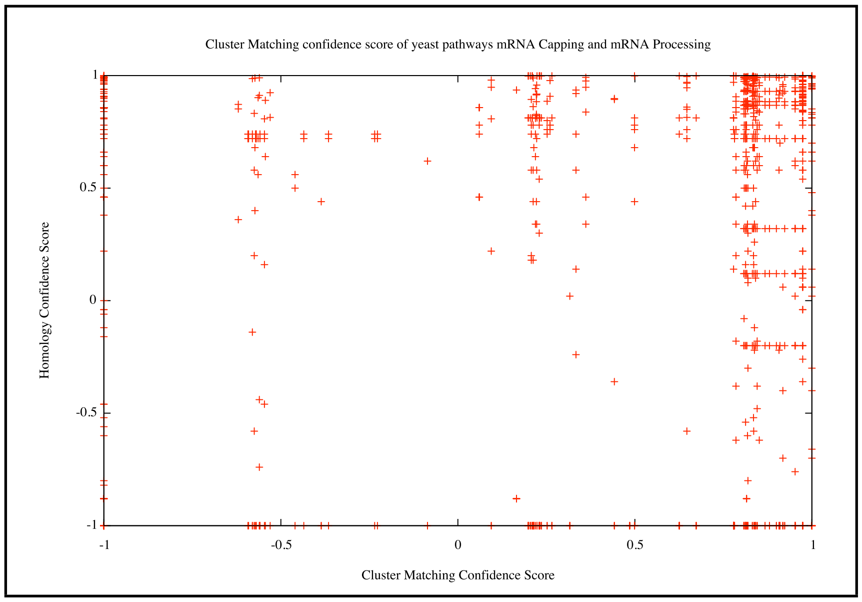

Figure 6.16: Shows the distribution between cluster matching confidence score as determined by our method and the homology between proteins as determined by BLASTp. Pairs of proteins derive from *Saccharomyces cerevisiae* pathways mRNA Capping and mRNA Processes.

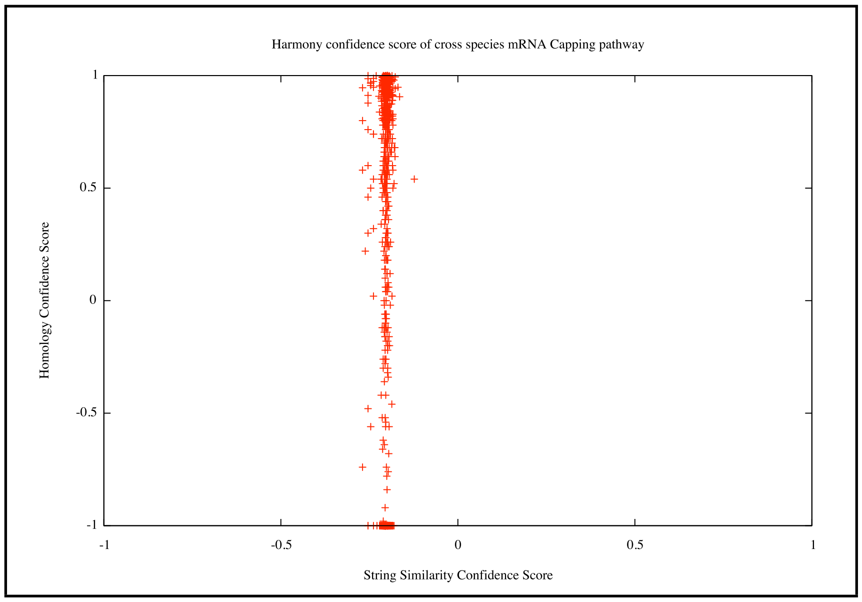

Figure 6.17: Shows the distribution between string similarity confidence score as determined by Harmony label matching and the homology between proteins as determined by BLASTp. Pairs of proteins derive from mRNA Capping pathway from *homo sapiens* and *Saccharomyces cerevisiae*.

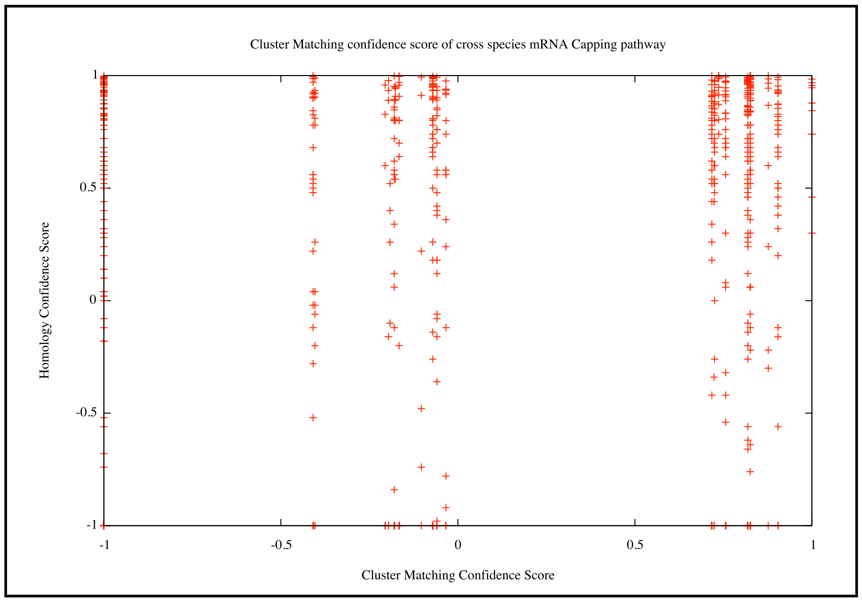

Figure 6.18: Shows the distribution between cluster matching confidence score as determined by our method and the homology between proteins as determined by BLASTp. Pairs of proteins derive from mRNA Capping pathway from *homo sapiens* and *Saccharomyces cerevisiae*.

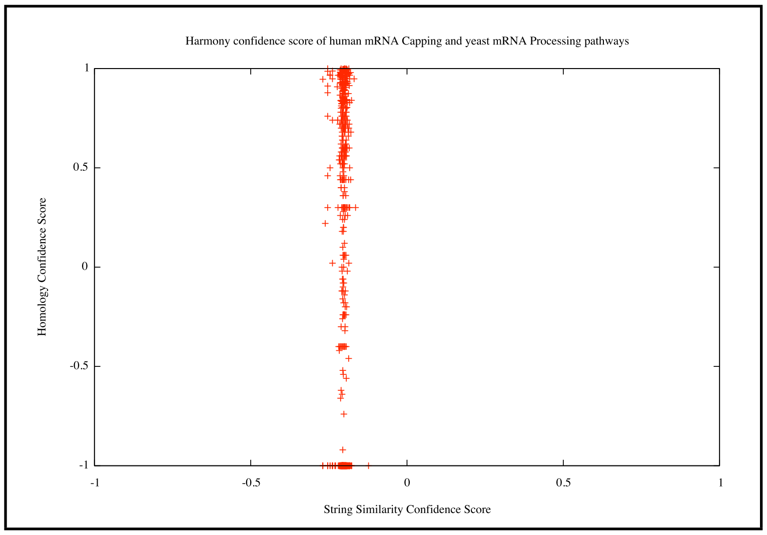

Figure 6.19: Shows the distribution between string similarity confidence score as determined by Harmony label matching and the homology between proteins as determined by BLASTp. Pairs of proteins derive from *homo sapiens* Metabolism in ncRNA pathway and the *Saccharomyces cerevisiae* mRNA Processing pathway.

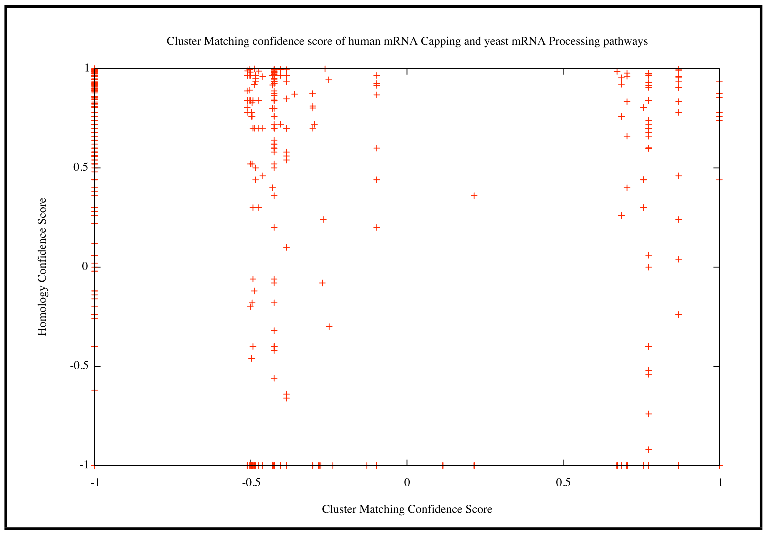

Figure 6.20: Shows the distribution between cluster matching confidence score as determined by our method and the homology between proteins as determined by BLASTp. Pairs of proteins derive from *homo sapiens* Metabolism in ncRNA pathway and the *Saccharomyces cerevisiae* mRNA Processing pathway.

Metabolism ncRNA	mRNA Capping	Harmony	Cluster Matching	Homology
Q96RS0	P30876	-0.066	0.923	0.500
Q96RS0	P24928	-0.022	0.923	0.760
Q96RS0	P52435	-0.052	0.923	0.882
Q96RS0	P19388	-0.059	0.923	0.760
Q96RS0	P62487	-0.034	0.923	0.580
Q96RS0	O15514	-0.046	0.923	0.850
Q96RS0	P62875	-0.056	0.923	0.826
Q96RS0	P52434	-0.069	0.923	0.720
P54105	P30876	-0.081	0.944	0.520
P54105	P24928	-0.102	0.944	0.460
P54105	P52435	-0.081	0.944	0.940
P54105	P19388	-0.084	0.944	0.460
P54105	P52434	-0.062	0.944	0.974
Q9BQA1	P30876	-0.084	0.923	0.970
Q9BQA1	P52434	-0.091	0.923	0.914
Q9H814	P30876	-0.058	0.923	0.924
Q9H814	P24928	-0.030	0.923	0.983
Q9H814	P19388	-0.042	0.923	0.983
Q9H814	P62487	-0.027	0.923	0.260
Q9H814	O15514	-0.016	0.923	0.993
Q9H814	P52434	-0.066	0.923	0.956

Table 6.5: The most highly confident matches between the *Homo Sapiens* pathways Metabolism of ncRNA and mRNA Capping from Figure 6.14. The columns Harmony, Cluster Matching and Homology correspond to the confidence scores from Harmony's label matching, our cluster matching method and BLASTp homology expectation values respectively.

mRNA Capping	mRNA Processing	Harmony	Cluster Matching	Homology
YOL005C	YPL122C	0.250	1.000	0.994
YOR210W	YDL108W	0.416	1.000	0.995
YPR187W	YPR025C	0.333	1.000	0.994
YGL070C	YDR460W	0.145	1.000	0.998
YGL070C	YIL143C	0.250	1.000	0.998
YGL070C	YLR005W	0.458	1.000	0.989
YGL070C	YPR056W	0.145	1.000	0.998
YGR005C	YGR005C	0.916	0.974	1.000
YGR005C	YDL108W	0.145	0.974	0.974
YGR005C	YDR311W	0.166	0.974	0.997
YPL122C	YPL122C	0.9166	0.974	1.000
YPL122C	YPR025C	0.4166	0.974	0.976
YER171W	YER171W	0.916	0.974	1.000
YER171W	YLR005W	0.250	0.974	0.980
YDR460W	YGR005C	0.312	0.974	0.997
YDR460W	YDR460W	0.916	0.974	0.993
YDR460W	YIL143C	0.145	0.974	0.993
YDR460W	YPR056W	0.416	0.974	0.993
YDL108W	YGR005C	0.145	0.974	0.974
YDL108W	YDL108W	0.916	0.974	1.000
YPR025C	YPR025C	0.916	0.974	1.000
YDR311W	YGR005C	0.166	0.974	0.997
YDR311W	YDR311W	0.916	0.974	1.000
YLR005W	YER171W	0.250	0.974	0.980
YLR005W	YLR005W	0.916	0.974	1.000
YPR056W	YPR025C	0.625	0.974	0.997
YPR056W	YLR005W	0.583	0.974	0.998

Table 6.6: The most highly confident matches between the *Saccharomyces cerevisiae* pathways mRNA Capping and mRNA Processing from Figure 6.16. The columns Harmony, Cluster Matching and Homology correspond to the confidence scores from Harmony's label matching, our cluster matching method and BLASTp homology expectation values respectively.

Human	Yeast	Harmony	Cluster Matching	Homology
Q99639	YPL122C	-0.246	1.000	0.968
Q99639	YDR460W	-0.246	1.000	0.956
Q99639	YDL108W	-0.269	1.000	0.946
Q99639	YPR056W	-0.184	1.000	0.984
O60942	YPR025C	-0.193	0.904	0.934
O60942	YIL143C	-0.204	0.904	0.936
P50613	YDR311W	-0.206	0.904	0.920
P50613	YLR005W	-0.203	0.904	0.926
P51946	YER171W	-0.213	0.904	0.981
P51946	YDL108W	-0.213	0.904	0.993
P51946	YDR311W	-0.213	0.904	0.926
P35269	YDR460W	-0.206	0.904	0.954
P35269	YPR025C	-0.197	0.904	0.926

Table 6.7: The most highly confident matches between the mRNA Capping pathways from *Homo Sapiens* (column Human) and *Saccharomyces cerevisiae* (column Yeast) from Figure 6.18. The columns Harmony, Cluster Matching and Homology correspond to the confidence scores from Harmony's label matching, our cluster matching method and BLASTp homology expectation values respectively.

Human	Yeast	Harmony	Cluster Matching	Homology
O60942	YPL178W	-0.205	1.000	0.876
O60942	YPL122C	-0.205	1.000	0.854
O60942	YPR025C	-0.193	1.000	0.934
P19447	YGR005C	-0.204	0.869	0.989
P19447	YPL178W	-0.194	0.869	0.904
P19447	YPL122C	-0.197	0.869	0.960
P19447	YER171W	-0.200	0.869	0.954
P52435	YMR125W	-0.200	0.869	0.997
P53803	YGR005C	-0.202	0.869	0.834
P53803	YMR125W	-0.205	0.869	0.934
P53803	YPL122C	-0.204	0.869	0.906

Table 6.8: The most highly confident matches *Homo Sapiens* Metabolism of ncRNA pathway (column Human) and *Saccharomyces cerevisiae* mRNA Capping pathway (column Yeast) from Figure 6.20. The columns Harmony, Cluster Matching and Homology correspond to the confidence scores from Harmony's label matching, our cluster matching method and BLASTp homology expectation values respectively.

Chapter 7

Conclusion and Future Work

Database management is faced with the problem of having large sets of unstructured data that contains large amount of metadata. Some of this metadata is organized in a manner that could be helpful in organizing the data for application use. However, there are challenges involved in defining the metadata for the best use. A parallel problem in bioinformatics deals with unorganized proteins, with structured annotation. Bioinformaticians are interested in finding meaningful methods to organize these proteins, or to find interesting associations between them.

We have developed a method that samples original structured metadata in the form of domain ontologies and clusters the annotated data to build a previously unknown organization - which we define as a context. We have applied our method to protein datasets, to find clusters of proteins with similar functional structure based on protein symbolic pathways from reactome.org. The ontology metadata is annotation from the Genome Ontology, which is organized in a hierarchical ontology. The metadata influences the proteins to form structures based on the symbolic pathways as see in reactome.org.

For complex datasets, this problem becomes increasingly difficult. This is especially true, because of the common theme of today's databases to use large amounts annotation. The need to incorporate all of this annotation requires orthogonal methods. Thus, we have made sure our algorithm is variable in the input structure - the metadata domain ontology is restricted only in that it must be hierarchical. The output is easily used as a 'confidence value', which is easily compared with other methods, such as the homology of proteins. The resulting method uses pro-processeing to reduce the time complexity to a manageable size depending on the the size of the input.

The preprocessing step is limited only by the size of the metadata. After preprocessing, the annotation space is significantly reduced from the time and space requirements, the time and space complexity is limited mostly by the size of the data set. The major drawback of this approach is that preprocessing needs to be rerun at each update of the data or metadata. There are some heuristics that could reduce the number of preprocessing reruns required when the metadata or data is updated. These heuristics could be investigated in the future.

The result on reactome proteins show that there exists much variability, but that the results contain interesting information. The speed of the method is highly dependent on preprocessing of the simulated annealing. The preprocessing step is slow because of the large size of both the data and metadata set. The output can be difficult to interpret for the same reasons. The sampling step can be time consuming - however, we add that the method is easily paralleled using divide and conquer strategies. Regardless, the results are meaningful - i.e. the application to proteins sets yielded proteins clustered via their GO annotation. There results are further meaning when combined with additional information, such as homology, as was intended.

Although the variability of the results can be seen as a drawback, we should note that running the simulated annealing process for longer than is shown can reduce this variability. This evidence is not provided.

There is much potential for this approach, and we would like to see future development in several areas. Apart from pre-processing analysis, we could investigate the use of supervised techniques for clustering (such as k-means) where the parameters could result from pre-processing. This could reduce the number of context mistakes.

We would also like to investigate the use of contexts as metadata in other areas, such as websearch, for either the semantic web or literature search. For example, for websearch in mutliple domains with related annotation, such as various media domains (video, music and book media).

Bibliography

[1] The gene ontology (go) project in 2006. *Nucleic Acids Res*, 34(Database issue):D322–6, 2006.

[2] Rakesh Agrawal, Ralf Rantzau, and Evimaria Terzi. Context-sensitive ranking. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 383–394, New York, NY, USA, 2006. ACM Press.

[3] Z. Aleksovski, W. ten Kate, and F. van Harmelen. Ontology matching using comprehensive ontology as background knowledge. In P. Shvaiko et al., editor, *Proceedings of the International Workshop on Ontology Matching at ISWC 2006*, pages 13–24. CEUR, 2006.

[4] G D Bader and C W Hogue. Bind–a data specification for storing and describing biomolecular interactions, molecular complexes and pathways. *Bioinformatics*, 16(5):465–477, 2000.

[5] Joel S Bader, Amitabha Chaudhuri, Jonathan M Rothberg, and John Chant. Gaining confidence in high-throughput protein interaction networks. *Nat Biotech*, 22(1):78–85, 2004.

[6] Joel S Bader, Amitabha Chaudhuri, Jonathan M Rothberg, and John Chant. Gaining confidence in high-throughput protein interaction networks. *Nat Biotechnol*, 22(1):78–85, 2004.

[7] Sylvain Brohee and Jacques van Helden. Evaluation of clustering algorithms for protein-protein interaction networks. *BMC Bioinformatics*, 7:488+, November 2006.

[8] V. Cerny. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51, January 1985.

[9] Hao Chen and Burt M Sharp. Content-rich biological network constructed by mining pubmed abstracts. *BMC Bioinformatics*, 5:147, 2004.

[10] Francesca D Ciccarelli, Tobias Doerks, Christian von Mering, Christopher J Creevey, Berend Snel, and Peer Bork. Toward automatic reconstruction of a highly resolved tree of life. *Science*, 311(5765):1283–1287, 2006.

[11] Olivier Dameron, Natalya F. Noy, Holger Knublauch, and Mark A. Musen. Accessing and manipulating ontologies using web services. In *Proceeding of the Third International Semantic Web Conference (ISWC2004), Semantic Web Services workshop*, 2004.

[12] Charlotte M Deane, Lukasz Salwinski, Ioannis Xenarios, and David Eisenberg. Protein interactions: two methods for assessment of the reliability of high throughput observations. *Mol Cell Proteomics*, 1(5):349–356, 2002.

[13] R Delongchamp, T Lee, and C Velasco. A method for computing the overall statistical significance of a treatment effect among a group of genes. *BMC Bioinformatics*, 7 Suppl 2:S11, 2006.

[14] Robin Dhamankar, Yoonkyong Lee, AnHai Doan, Alon Halevy, and Pedro Domingos. imap: discovering complex semantic matches between database schemas. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 383–394, New York, NY, USA, 2004. ACM Press.

[15] AnHai Doan and Alon Y. Halevy. Semantic-integration research in the database community. *AI Mag.*, 26(1):83–94, 2005.

[16] Elisabeth Gasteiger, Alexandre Gattiker, Christine Hoogland, Ivan Ivanyi, Ron D Appel, and Amos Bairoch. Expasy: The proteomics server for in-depth protein knowledge and analysis. *Nucleic Acids Res*, 31(13):3784–3788, 2003.

[17] M. Gertz and I. Schmitt. Data Integration Techniques based on Data Quality Aspects. In I. Schmitt, C. Türker, E. Hildebrandt, and M. Höding, editors, *Proceedings 3. Workshop "Föderierte Datenbanken", Magdeburg, 10./11. Dezember 1998*, pages 1–19. Shaker Verlag, Aachen, 1998.

[18] Katherine G. Herbert, Narain H. Gehani, William H. Piel, Jason T. L. Wang, and Cathy H. Wu. Bio-ajax: an extensible framework for biological data cleaning. *SIGMOD Rec.*, 33(2):51–57, 2004.

[19] J. Jannink, S. Pichai, D. Verheijen, and G. Wiederhold. Encapsulation and composition of ontologies, 1998.

[20] Trupti Joshi and Dong Xu. Quantitative assessment of relationship between sequence similarity and function similarity. *BMC Genomics*, 8:222, 2007.

[21] Minoru Kanehisa, Susumu Goto, Masahiro Hattori, Kiyoko F Aoki-Kinoshita, Masumi Itoh, Shuichi Kawashima, Toshiaki Katayama, Michihiro Araki, and Mika Hirakawa. From genomics to chemical genomics: new developments in kegg. *Nucleic Acids Res*, 34(Database issue):D354–7, 2006.

[22] Vipul Kashyap and Amit Sheth. Semantic and schematic similarities between database objects: a context-based approach. *The VLDB Journal*, 5(4):276–304, 1996.

[23] Danielle Kemmer, Yong Huang, Sohrab P Shah, Jonathan Lim, Jochen Brumm, Macaire M S Yuen, John Ling, Tao Xu, Wyeth W Wasserman, and B F Francis Ouellette. Ulysses - an application for the projection of molecular interactions across species. *Genome Biol*, 6(12):R106, 2005.

[24] S. Kerrien, Y. Alam-Faruque, B. Aranda, I. Bancarz, A. Bridge, C. Derow, E. Dimmer, M. Feuermann, A. Friedrichsen, R. Huntley, C. Kohler, J. Khadake, C. Leroy, A. Liban, C. Lieftink, L. Montecchi-Palazzi, S. Orchard, J. Risse, K. Robbe, B. Roechert, D. Thorneycroft, Y. Zhang, R. Apweiler, and H. Hermjakob. Intact–open source resource for molecular interaction data. *Nucleic Acids Research*, 35(suppl_1):D561–565, 2007.

[25] S Kiritchenko, S Matwin, and F Famili. Functional annotation of genes using hierarchical text categorization. *BioLINK SIG: Linking Literature, Information and Knowledge for Biology.*, 2005.

[26] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983*, 220, 4598:671–680, 1983.

[27] Martin Krallinger and Alfonso Valencia. Text-mining and information-retrieval services for molecular biology. *Genome Biol*, 6(7):224, 2005.

[28] Patrick Lambrix and Anna Edberg. Evaluation of ontology merging tools in bioinformatics. In *Proc. Pacific Symposium on Biocomputing*, pages 589–600, Kauai, Hawaii, USA, 2003.

[29] D. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project.* Addison-Wesley Professional., 1990.

[30] Maurizio Lenzerini. Data integration: a theoretical perspective. In *PODS '02: Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246, New York, NY, USA, 2002. ACM Press.

[31] Brenton Louie, Peter Mork, Fernando Martin-Sanchez, Alon Halevy, and Peter Tarczy-Hornoch. Data integration and genomic medicine. *J Biomed Inform*, 40(1):5–16, 2007.

[32] Hammad Majeed, Conor Ryan, and R. Muhammad Atif Azad. Evaluating gp schema in context. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 1773–1774, New York, NY, USA, 2005. ACM Press.

[33] L R Matthews, P Vaglio, J Reboul, H Ge, B P Davis, J Garrels, S Vincent, and M Vidal. Identification of potential interaction networks using sequence-based searches for conserved protein-protein interactions or "interologs". *Genome Res*, 11(12):2120–2126, 2001.

[34] L R Matthews, P Vaglio, J Reboul, H Ge, B P Davis, J Garrels, S Vincent, and M Vidal. Identification of potential interaction networks using sequence-based searches for conserved protein-protein interactions or "interologs". *Genome Res*, 11(12):2120–2126, 2001.

[35] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[36] Peter Mork, Arnon Rosenthal, Joel Korb, and Ken Samuel. Integration workbench: Integrating schema integration tools. *icdew*, 0:3, 2006.

[37] Hans-Michael Muller, Eimear E Kenny, and Paul W Sternberg. Textpresso: an ontology-based information retrieval and extraction system for biological literature. *PLoS Biol*, 2(11):e309, 2004.

[38] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of SIAM*, 5(1):32–38, 1957.

[39] Rakesh Nagarajan, Mushtaq Ahmed, and Aditya Phatak. Database challenges in the integration of biomedical data sets. In *VLDB*, pages 1202–1213, 2004.

[40] Manikandan Narayanan and Richard M Karp. Comparing protein interaction networks via a graph match-and-split algorithm. *J Comput Biol*, 14(7):892–907, 2007.

[41] Ian Niles and Adam Pease. Towards a standard upper ontology. In *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems*, pages 2–9, New York, NY, USA, 2001. ACM.

[42] F. Noy and A. Musen. Evaluating ontology-mapping tools: Requirements and experience, 2002.

[43] F. Noy and N. Musen. An algorithm for merging and aligning ontologies: Automation and tool support, 1999.

[44] N. Noy and M. Musen. Smart: Automated support for ontology merging and alignment, 1999.

[45] N. Noy and M. Musen. The prompt suite: Interactive tools for ontology merging and mapping, 2002.

[46] Natalya Fridman Noy and Michel C. A. Klein. Ontology evolution: Not the same as schema evolution. *Knowl. Inf. Syst.*, 6(4):428–440, 2004.

[47] Debnath Pal. On gene ontology and function annotation. *Bioinformation*, 1(3):97–98, 2006.

[48] Lukasz Salwinski, Christopher S. Miller, Adam J. Smith, Frank K. Pettit, James U. Bowie, and David Eisenberg. The database of interacting proteins: 2004 update. *Nucleic Acids Research*, 32(suppl_1):D449–451, 2004.

[49] Vineet Sangar, Daniel J Blankenberg, Naomi Altman, and Arthur M Lesk. Quantitative sequence-function relationships in proteins based on gene ontology. *BMC Bioinformatics*, 8:294, 2007.

[50] Roded Sharan and Trey Ideker. Modeling cellular machinery through biological network comparison. *Nat Biotechnol*, 24(4):427–433, 2006.

[51] Michael Siegel and Stuart E. Madnick. Context interchange: sharing the meaning of data. *SIGMOD Rec.*, 20(4):77–78, 1991.

[52] Thomas Strang, Claudia Linnhoff-Popien, and Korbinian Frank. Applications of a Context Ontology Language. In Dinko Begusic and Nikola Rozic, editors, *Proceedings of International Conference on Software, Telecommunications and Computer Networks (SoftCom2003)*, pages 14–18, Split/Croatia, Venice/Italy, Ancona/Italy, Dubrovnik/Croatia, October 2003. Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, University of Split, Croatia.

[53] Philip Tan, Stuart E. Madnick, and Kian-Lee Tan. Context mediation in the semantic web: Handling owl ontology and data disparity through context interchange. In *SWDB*, pages 140–154, 2004.

[54] Zuojian Tang, Sieu Phan, Youlian Pan, and A. F. Famili. Prediction of co-regulated gene groups through gene ontology. pages 178–184, 2007.

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

[56] M. Theodorakis, Anastasia Analyti, Panos Constantopoulos, and Nicolas Spyratos. A theory of contexts in information bases. *Information Systems*, 27(3):151–191, 2002.

[57] Imre Vastrik, Peter D'Eustachio, Esther Schmidt, Geeta Joshi-Tope, Gopal Gopinath, David Croft, Bernard de Bono, Marc Gillespie, Bijay Jassal, Suzanna Lewis, Lisa Matthews, Guanming Wu, Ewan Birney, and Lincoln Stein. Reactome: a knowledge base of biologic pathways and processes. *Genome Biol*, 8(3):R39, 2007.