Combining evolution and machine learning for functional annotation and classification of remote homologous proteins.

Juliana Silva Bernardes

Universidade Federal do Rio de Janeiro
PESC COPPE

Université Pierre et Marie Curie
UMR 7238

École doctorale informatique, télécommunications et électronique de Paris

Thèse de doctorat

Discipline : Informatique

présentée par

Juliana Silva Bernardes

Evolution et apprentissage automatique pour l'annotation fonctionnelle et la classification des homologies lointains en protéines.

---

dirigée par Alessandra Carbone,
Gerson Zaverucha

Soutenue le 28 mars 2012 devant le jury composé de :

M. Gerson Zaverucha	Universidade Federal do Rio de Janeiro	président
M^{me}. Alessandra Carbone	Université Pierre et Marie Curie	directrice
M^{me}. Catherine Vaquero	Université Pierre et Marie Curie	examinatrice
M. Valmir C. Barbosa	Universidade Federal do Rio de Janeiro	examinateur
M. Alberto Davila	Fundação Orvaldo Cruz	rapporteur
M. Yann Guermeur	l'Université de Lorraine	rapporteur

L'UFR díngénierie
Université Pierre et Marie Curie
4, Place Jussieu, 75252
Paris Cedex 05, France

Programa de Engenharia de
Sistemas e Computação, COPPE
Universidade Federal do
Rio de Janeiro
Caixa Postal 68504, 21941-972
Rio de Janeiro - RJ, Brazil

Génomique des Microorganismes
UMR 7238 CNRS-Université
Pierre et Marie Curie
Site des Cordeliers
Bât. A - 4éme étage,
15, Rue de l'Ecole de Médecine
75006 Paris, France

Abstract

Evolution et apprentissage automatique pour l'annotation fonctionnelle et la classification des homologies lointains en protéines.

Résumé

La détection d'homologues lointains est essentielle pour le classement fonctionnel et structural des séquences protéiques et pour l'amélioration de l'annotation des génomes très divergents. Pour le classement des séquences, nous présentons la méthode «ILP-SVM homology», combinant la programmation logique inductive (PLI) et les modèles propositionnels. Elle propose une nouvelle représentation logique des propriétés physico-chimiques des résidus et des positions conservées au sein de l'alignement de séquences. Ainsi, PLI trouve les règles les plus fréquentes et les utilise pour la phase d'apprentissage utilisant des modèles d'arbre de décision ou de machine à vecteurs de support. La méthode présente au moins les mêmes performances que les autres méthodes trouvées dans la littérature. Puis, nous proposons la méthode CASH pour annoter les génomes très divergents. CASH a été appliqué à *Plasmodium falciparum*, mais reste applicable à toutes les espèces. CASH utilise aussi bien l'information issue de génomes proches ou éloignés de *P. falciparum*. Chaque domaine connu est ainsi représenté par un ensemble de modèles évolutifs, et les sorties sont combinées par un méta-classificateur qui assigne un score de confiance à chaque prédiction. Basé sur ce score et sur des propriétés de co-ocurrences de domaines, CASH trouve l'architecture la plus probable de chaque séquence en appliquant une approche d'optimisation multi-objectif. CASH est capable d'annoter 70% des domaines protéiques de *P. falciparum*, contre une moyenne de 57% pour ses concurrents. De nouveaux domaines protéiques ont pu être caractérisés au sein de protéines de fonction inconnue ou déjà annotées.

Mots-clefs

Biologie computationelle, approche discriminative, programmation logique inductive, machine à vecteurs de support, ensemble de modèles, optimisation multi-objectif.

Combining evolution and machine learning for functional annotation and classification of remote homologous proteins.

Abstract

Detection of remote homologous proteins is essential for functional and structural classification of protein sequences and for the completion of the annotation for highly divergent genomes. Here, we present two new methods to address these problems. For the first problem, we introduce ILP-SVM Homology that combines inductive logic programming (ILP) and propositional models. It proposes a novel logical representation of physico-chemical properties, conserved amino acid positions and conserved physico-chemical positions in sequence alignments. Based on these signals, ILP finds the most frequent patterns and uses them to train models, such as decision trees and support vector machines. ILP-SVM Homology achieves at least equal performance when compared with other methods. To address the second problem, we propose CASH, a large-scale pipeline to annotate highly divergent genomes. CASH was applied to the *Plasmodium falciparum*, but it is applicable to any species. In CASH we explore different evolutionary pathways including those that are phylogenetically distant from *P. falciparum*. As a result, each known domain is represented by an ensemble of heterogeneous models, and the outputs are combined through a meta-classifier that assigns a confidence score to each prediction. Based on this score and on properties as domain co-occurrence, CASH finds the most probable architecture for each query sequence by resolving a multi-objective optimization problem. CASH provides domain annotation for 70% of proteins in *P. falciparum*, while its competitors achieve at most 57%. We find additional domains into already annotated proteins, and predict domains for proteins with unknown function.

Keywords

Computational Biology, Discriminative approaches, Inductive Logic Programming, Support Vector Machine, Ensemble of models, Multi-objective optimization.

Contents

Chapter 1

Introduction

1.1 Background

In the last decade, the large-scale sequencing and genome projects have completed the genome of many distinct organisms. With the sequence in hand, the challenge is to analyze and to interpret this massive amount of biological data (genes and proteins). The biochemical role of each protein is crucial to the understanding of the complex cellular machinery of these different organisms. Proteins are macromolecules formed by a chain of amino acids. This chain is typically folded by producing a particular three-dimensional structure that is related to biological function of the protein. When proteins have their function characterized they are deposited into public databases [1, 2, 3, 4], where they can be scanned to infer important properties for new proteins, such as function [5], functional sites [6], structure [7], and evolutionary relationships [8].

Here, we focalize our attention on the functional characterization of new proteins. This is an important problem in Computational Biology, playing a crucial role in sequence annotation and protein family classification. A solution to the problem will have a practical strong impact to guiding laboratory experiments. In order to infer the function of a target protein, computational methods search into public databases for a known protein that be *homologous* to the target. Homologs are proteins derived from a common ancestor and that probably share the same function. Thus, one could try to assign to the target protein the function of known homologous protein. This process is called functional annotation transfer. Traditional approaches, such as BLAST [9] and FASTA [10], deal with the homology detection problem by searching for regions of local similarity among pairs of sequences. They can successfully detect protein homology when the identity (percentage

of identical amino acids in a pairwise alignment), is greater than 30%. However, proteins can be homologs even in the case of low sequence identity. They are known as remote homologous proteins, that is, they have a common ancestor but they have diverged significantly in their primary sequence during their evolutionary history. This is an important and hard problem: the development of methods addressing the problem of identifying remote homology in proteins is essential for functional and comparative proteomic.

Generative methods are an alternative to the traditional methods and they are often more effective in detecting remote homologous proteins. First, they train a model to represent a group of homologous sequences (a protein family), and then match a query sequence against the model to evaluate the similarity of the query sequence to the group/family. Profile Hidden Markov Models (pHMMs) [11] and PSI-BLAST [12] are examples of such approaches, also known as family-based or sequence-profile based approaches. Although, sequence-profile approaches achieve better performance than methods based on pairwise comparison only [13], they still largely fail to detect distant homologous proteins. A significant improvement over those methods were made possible by comparing profile-profile instead of sequence-profile. Methods such as PROF-SIM [14], COMPASS [15] and HHsearch [16] build a profile from the query protein and then compare it against a profile database constructed from the target proteins.

Limitations in the performance of generative methods has motivated researchers to follow mainly two directions, i) to combine extra information to previous approaches, such as phylogenetic [17], protein structure information [18, 19], and domain co-occurrence [20, 21, 22, 23, 24, 25] and ii) to search for new accurate methods. Among the new approaches, a family of methods called "discriminative", have been able to attain additional accuracy to remote homology detection by modeling the differences between positive and negative examples. The most popular discriminative method applied to the remote homology detection problem is Support Vector Machine (SVM) [26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42]. Basically, SVMs learn a classification function, from positive and negative training examples, that optimally separates the unseen data into two categories, for instance, homologous and non-homologous proteins. The kernel function that measures the similarity between a pair of examples plays a key role in the SVM performance. Typically, each protein sequence is represented by a fixed-length vector, where

each vector's item is a protein's property, and the kernel function is defined to be the inner product between different vectors. An alternative to this standard kernel function is pre-compute a kernel matrix where each element is the measure of similarity between a pair of examples [32, 36].

The main difference among SVM approaches is the protein property representation. Since, remote homology detection is a hard problem, the protein property representation plays a key role in the performance of the method. Sophisticated and effective representations can be obtained if relational learning algorithms are employed, such as Inductive logic programming (ILP) [43]. This idea has been exploited by several methods [44, 45, 46, 47] as an attempt to improve the performance of remote homology detection.

1.2 Challenges of detecting remote protein homology

The functional characterization of proteins still represents a challenge for computational biology. A number of protein sequences of different organisms have been identified but their specific functions remain unknown. In particular, the genome of some of these organisms is so divergent that almost half of the proteins into the genome remain without any putative annotation. This is the case of *Plasmodium falciparum* genome, the organism that causes Malaria in human. The development of accurate methods for remote homology detection based on sequences is crucial to improve functional annotation. In fact, these methods can help to improve the classification of remote homologous proteins that are later used to annotate highly divergent genomes. It is known that structural properties can improve the performance of remote homology detection methods [19, 18], but approaches that use this kind of information cannot be used into large-scale systems, since structures are not available for most existing proteins. Thus, an intelligent use of information coming from sequence is necessary to hope for a large scale annotation. Such systems should exhibit a feasible computational time in order to treat the large amount of biological data.

1.3 Significance and contribution of the thesis

To tackle these challenges, we introduced two approaches for remote homology detection. The first method called ILP-SVM Homology [48] is a hybrid approach that combines inductive logic programming and propositional models for classifying remote homologous

proteins, and the second method, called CASH - Combination of Annotations by Species and pHMMs [49] - combines several computational approaches and explores protein pathways of evolution to provide annotation of highly divergent genomes.

For the development of ILP-SVM Homology method, our motivation was to explore the first-order logic language to represent, in a more sophisticated way, properties of homologous protein sequences. We have used sequential properties, as done in previous approaches [47, 45, 44, 46], and we introduced a novel representation based on conserved amino acid positions and conserved physico-chemical positions in a Multiple Sequence Alignment (MSA). From these representations, we induced through the Inductive Logic Programming (ILP) a set of logical rules that summarize essential features of protein functions. These rules were transformed in features that were used to train propositional models, such as Decision Trees (DT) and SVM. We used the SCOP [8] database to perform our experiments by evaluating protein recognition within the same superfamily, where a common evolutionary origin is not easily deduced from sequence identity. Our results show that our methodology, when using SVM, performs significantly better than some of the state of the art methods, and that it is comparable otherwise. The contributions of this hybrid methodology can be summarized as follows:

- We have proposed for the first time a logical representation based on conserved amino acid positions in a MSA. We have related this new representation with a sequential representation [47, 45, 44, 46] creating a new one that takes into account conserved physico-chemical positions in a MSA. Our results showed that these two new representations perform better than sequential ones.

- We proposed the use of SVM in place of DT and this strategy achieved better performance.

- We showed that exploring only conserved positions in a MSA is more suitable to the remote homology detection problem than exploring all amino acids positions. We demonstrated that remote homologous proteins seem to share a number of restricted properties in order to keep their function, and we succefully represented these properties by first-order logic predicates. Based on this set of properties, an ILP system

was able to filter logical rules that summarize essential features of homologous proteins.

- By validating our approach, we showed that the performance of our model is at least comparable, but in some cases better than some of its competitors. This includes the cases where sequence identity is low (that is, below 30%). The output of our method can be interpreted biologically to provide insights into conserved features of homologous protein families.

For the development of CASH method, our motivation was to explore alternative evolutionary protein pathways and to provide a large-scale pipeline for highly divergent genomes annotation. In fact, CASH is a hybrid methodology that employs both generative and discriminative methods. First, CASH proposes the use of additional generative models to represent each known Pfam domain [50]. For this, it constructs PSI-Blast profiles starting from a large and differentiated panel of homologous sequences associated to a protein domain. Second, CASH combines these models with existing pHMMs by providing an ensemble of generative models to represent each known domain. Third, the outcomes of generative models are processed and transformed into features used to train a discriminative method (SVM) that assigns a confidence score to each domain prediction. Finally, based on this score and on other properties, as domain co-occurrence [20, 21, 22, 23, 24, 25], CASH finds the most probable architecture for each query protein sequence by resolving a multi-objective optimization problem. Applied to the *P. falciparum* genome, CASH achieved superior performance when compared to any state-of-art methods. It finds additional domains into already annotated proteins, and predicts domains for proteins with unknown functions. The method has been compared to Pfam, CODD [24] and dPUC [25]. In the following, we highlighted CASH contributions:

- The use of alternative profiles created from orthologous sequences was proposed in [51]. It explored a very small set of sequences, made no use of prediction combination strategies, and it just considered the profile that best matches a target sequence in order to provide its annotation. Here, we increased the number of orthologous sequences what allowed us to explore different evolutionary protein pathways within

the life phylogenetic tree, and possibly those of species that are phylogenetically distant from *P. falciparum*. Moreover, we proposed to combine predictions coming from different profiles to produce a more reliable domain annotation.

- CASH proposes, for the first time, to represent each known Pfam domain through an ensemble of heterogenous models that compete for the best detection signals. Also, it employs a meta-learning [52, 53] combination strategy that uses a second classifier (SVM) trained from features that were obtained by pre-processing outputs of heterogenous models. Our meta-features have, as aim, to highlight individual model results that can help the prediction, when a consensus among base models is not observed, and to provide an indication of the performance of all models.

- In order to find the most likely domain arrangement/architecture for a given query sequence, we introduced a novel algorithm that treats this problem as a multi-objective optimization problem. For that, a set of objective functions is defined for each candidate solution, objective functions are arranged in order of importance, and constraints are formulated on these functions that are optimized one by one. To the best of our knowledge, it is the first time that the domain architecture problem is addressed as a multi-objective optimization problem.

We demonstrated that when homologous sequences are too divergent, signals of homology must be explored in a more intelligent way. We did it by exploring only the most important patterns characterizing homologous proteins in the ILP-SVM-Homology method, and by exploring alternative protein evolutionary pathways through the construction of additional profiles in the CASH method.

1.4 Publications

The following publications arose from works conducted during the course of this thesis research:

- Bernardes, J. S., Carbone, A. and Zaverucha, G. "A discriminative method for family-based protein remote homology detection that combines inductive logic programming and propositional models", published in *BMC Bioinformatics* (2011) [48].

It is a following research of previous works reported in [19, 6]. The main ideas and algorithms of this paper are presented in chapter 4.

- Bernardes, J. S., Zaverucha, G., Vaquero C., and Carbone, A. "Combining evolution and machine learning for functional annotation in *Plasmodium falciparum* annotation", to be submitted to the *Genome Research Journal*. The main ideas, methodology and results of this work are presented in chapter 5.

1.5 Organization of the thesis

In chapter 2, we present some basic ideas about proteins, their functions, and their structures. We present some concepts of protein evolution and explain the problem treated in this thesis. In Chapter 3, we review the main state-of-art methods for remote homology detection. In Chapter 4, we present the ILP-SVM-Homology method, we show and discussed the results, and we draw some conclusion about this work. In Chapter 5, CASH pipeline is presented in details, results are discussed, and conclusions are given. In Chapter 6, we conclude this thesis and discuss some direction for future work.

List Of Abbreviation

3D	-	Three-Dimensional
Acc	-	Accuracy
AUC	-	Area Under Curve
BLAST	-	Basic Local Alignment Search Tool
CASH	-	Combination of Annotations by Species and PHMMs
CODD	-	Co-Occurrence Domain Discovery
D	-	Delete state in a pHMM
DNA	-	Deoxyribonucleic Acid
dPUC	-	domain Prediction using context
DT	-	Decision Tree
HI	-	Homology Induction
I	-	Insert state in a pHMM
ILP	-	Inductive Logic Programming
LSA	-	Latent Semantic Analysis
M	-	Match state in a pHMM
MDCO	-	Multi-Domain Co-Occurrences
MIS	-	Maximal Independent Set
mRNA	-	message RiboNucleic Acid
MSA	-	Multiple Sequence Alignment
PDB	-	Protein DataBank
PHM	-	phylogenetic models
pHMM	-	Profile Hidden Markov Models
PPV	-	Positive Predictive Value
PR	-	Precision Recall
PSI-BLAST	-	Position-Specific Iterative BLAST
PSSM	-	Position Specific Score Matrix
ROC	-	Receiver Operating Characteristic
SCOP	-	Structural Classification of Proteins
Sen	-	Sensitivity
Spe	-	Specificity
SSS	-	Sequence Similarity Searching
SVM	-	Support Vector Machine
WHO	-	World Health Organization

List of Figures

List of Tables

Chapter 2

Proteins

This chapter introduces some basic concepts about proteins that are objects of this study. In section 2.1, we discuss the importance of proteins for living organisms, and present some concepts about their structures and functions. Proteins in different organisms can share a common ancestor, and they are called homologous proteins, we discuss their evolutionary relationships in section 2.2, also we describe in section 2.2.1, the remote homology detection problem, that is the central problem addressed in this thesis. We finalize this chapter presenting in section 2.3 some key concepts needed for understanding the rest of the thesis.

2.1 Structure and function

The deoxyriboNucleic Acid (DNA) carries the genetic information of the living organism cells, this information is encoded within thousands of genes. Each gene serves as a recipe on how to build a protein molecule. Two cellular processes are involved in the synthesis of a protein: transcription [54] and translation [55]. During the process of transcription the genetic information stored in a gene is transferred to a molecule called message RiboNucleic Acid (mRNA), and during the process of translation this information is decoded by the ribosome (a cellular component) to produce a specific amino acid chain that will fold into an active protein. The information encoded in the mRNA is "read" according to the genetic code, which relates the DNA sequence to the amino acid sequence in the protein. Each group of three nucleotides in mRNA constitutes a codon, and each codon specifies a particular amino acid. Amino acids are considered as building blocks of proteins. An amino acid is a molecule containing an amine group (NH_2), a carboxylic acid group (CO_2H),

and a side-chain that is specific to each amino acid. Basically, there are 20 different amino acids, and they can be divided according to their physico-chemical properties. The most important properties are charge, hydrophobicity, hydrophilicity, size, and side-chain specificity, see Figure 2.1, that shows for each amino acid its chemical formula (side chains are highlighted in red) and its physico-chemical properties. These 20 amino acids can be arranged in several different ways to create a number of different proteins. Proteins are typically folded into a particular three-dimensional structure that is related to their biological function. Physico-chemical properties of amino acid proteins play a major role in folding protein structure. For instance, the water-soluble proteins tend to have their hydrophobic residues buried in the middle of the protein, whereas hydrophilic side-chains are exposed to the aqueous solvent.

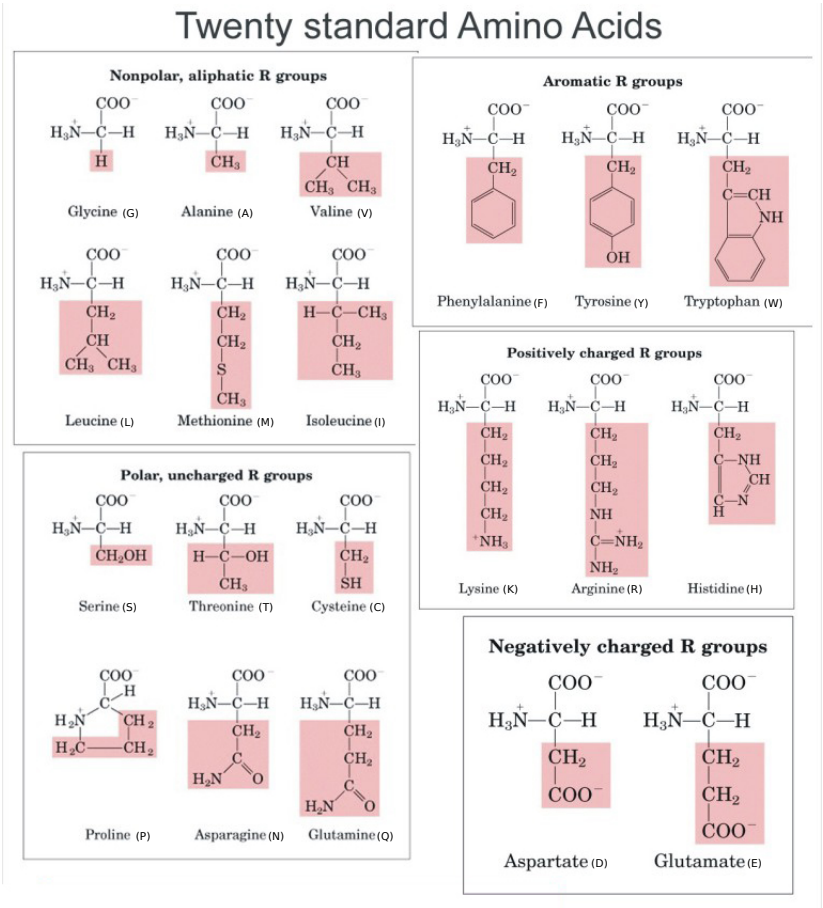


Figure 2.1: **Amino acids and their physico-chemical properties.**

Proteins are involved in a huge number of activities within a cell, such as: the gene regulation, the RNA transcription, the protein translation, the transport of materials, the

catalysis of biochemical reactions (enzymes), they act as receptors for hormones, etc. Frequently, the function of a protein is determined by its structure. During the *synthesis of a protein*, that is, the cell process where proteins are produced, the protein structure is in its *primary form*. This form, also called primary structure, refers to amino acid sequence of the protein. Amino acids are held together by chemical bonds, which are made during the process of protein synthesis. As a next step regular local sub-structures, known as *secondary structure*, are formed. There are two main types of secondary structures: the alpha helix and the beta strand [56]. These regular structures are connected by a “loop" [56]. Loops are uncoiled regions of variable sizes. Next, the alpha-helices, beta-strands and loops are folded into a compact globule by forming the three-dimensional structure. Many proteins are formed by a larger assembly of several protein molecules, usually called subunits. These subunits form complexes called quaternary structure. Figure 2.2 illustrates the four structural descriptions of the protein. Note that, they describe different structural subunits of the protein, and they do not illustrate intermediate steps of the folding process. No precise knowledge of the folding process is yet available.

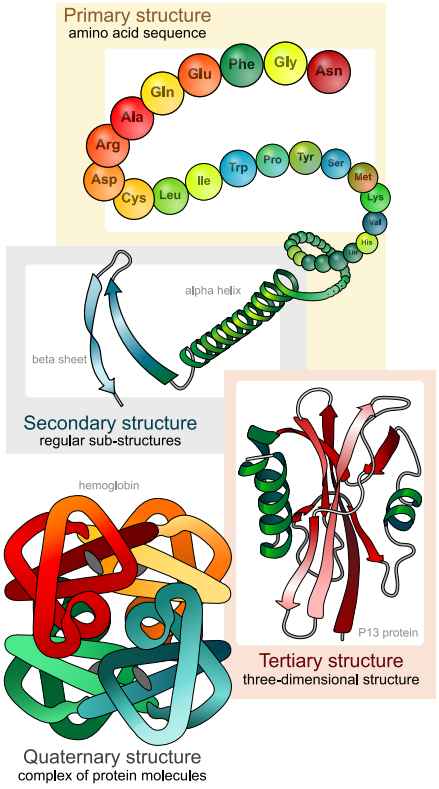


Figure 2.2: **Protein structure phases.**

2.2 Homologous proteins

Similarities among species suggest that all living organisms have origin in the same ancestor. Evolutionary processes, such as gene duplications [57] and mutations [58] give rise to diversity at every level of biological organization, including species and molecules such as proteins. For instance, when a gene is essential for a given species it can be duplicated, and two copies of this gene are produced. For example, in Figure 2.3-A the gene A was duplicated, and two identical copies A_1 and A_2 were created, these homologous genes are called *paralogs*. After duplication, genes A_1 and A_2 evolve independently, and they can suffer mutation events by producing paralogous genes A_3 and A_4, respectively, see Figure 2.3-B. Through speciation [59] new species arise, as shown in Figure 2.3-C, where species II and III were created from species I. Homologous genes in different species are called orthologs, see Figure 2.3-D. Since proteins are produced from genes, homologous genes produce homologous proteins.

A common way for studying the evolutionary relationships in homologous proteins is to perform a Multiple Sequence Alignment (MSA), as shown in Figure 4.1. The alignment of homologous proteins consist of trying to place amino acids in positions that derive from a common ancestral amino acid. To do so, we need to introduce *gaps*, which represent insertion or deletion into sequences. Thus, an alignment is a hypothetical model of mutations (substitutions, insertions, and deletions) that occurred during sequence evolution.

2.2.1 Remote homologous proteins

Homology can be detected easily if a strong sequence similarity is observed among proteins. A possible way to measure this similarity is to determine sequence identity among homologous proteins, that is, the percentage of identical amino acids in a protein alignment. If sequence identity is greater than 30%, homology can be asserted [60] with confidence. Otherwise, we say that these proteins are in the "Twilight zone", where homology signals get blurred, and more evidences are needed to confirm the homology. However, proteins can be homologs even in the case of low sequence identity. They are known as remote homologous proteins, that is, they have a common ancestor but they have diverged significantly in their primary sequence during their evolutionary history. To illustrate the

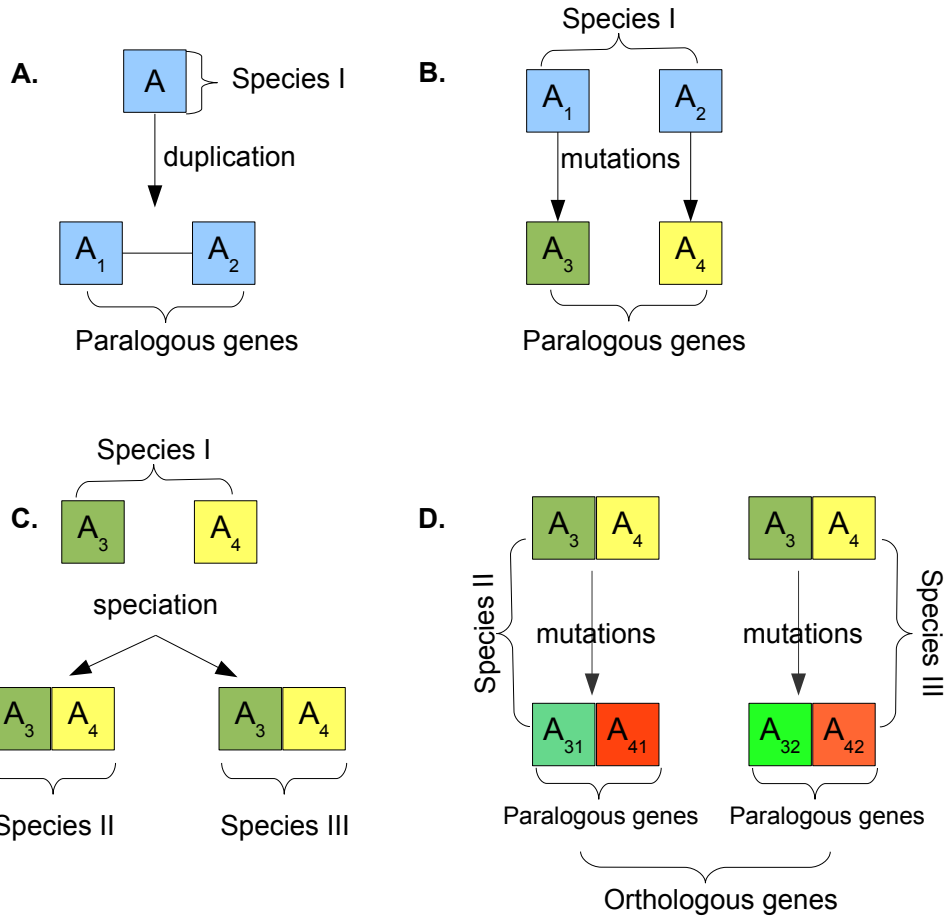


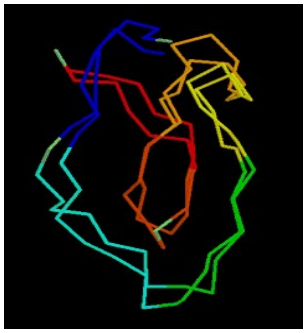
Figure 2.3: **Representative schema to ilustrate paralogous, orthologous and homologous genes.**

concept, observe Figure 2.4-A that shows two homologous proteins: on the top, see their sequence alignment, where identical amino acids are indicated by the symbol $*$, and on the bottom, see their structural alignment. Note that, sequence identity in the sequence alignment is low and that the hydrophobic blocks (highlighted in grey), known to play an important role in protein structural stability [61] are not aligned. Based only on these observations, we cannot assert that these proteins are homologs and only an analysis of their structural similarities can conclude it. On the other hand, non-homologous proteins such as those show in figure 2.4-B can present physico-chemical similarities like the conservation of hydrophobic blocks, but their structures show clearly that they are not homologs. This shows that remote homology detection is hard when using only sequence properties. To make it possible, we should mine valuable properties from homologous

protein sequences that allow us to identify homologous proteins, and in the same time, avoid false predictions. For this, we propose two methods presented in chapters 4 and 5.

A.

```
1fleI  AQEPVKGPVSTKPGSCPIILIRCAMLNPPN----RCLKDTDCPGIKKCCEGSCG-MACFVP-Q
1udkA  ----------NEKSGSCPDMSM---PIPPLGICKTLCNSDSGCPNVQKCCKNGCGFMTCTTPVP
                  ****          *       *  *  **   ***   ** * *  *
```



B.

```
1fleI -AQEPVKGPVSTKPGSCPIILIRCAMLNPPNRCLKD---TDCPG-IKKCCEGSCGMACFVPQ----
1a0aA MKRESHKHAEQARRNRLAVALHELASLIPAEWKQQNNVSSAAPSKAATTVEAACRYIRHLQQNGST
         *  *             *   * * *             *       *  *       *
```

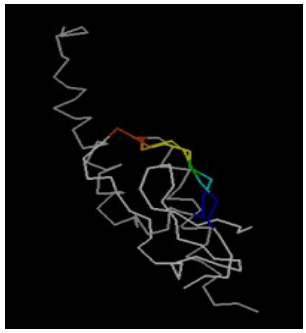


Figure 2.4: **Pairs of proteins aligned by amino acid sequence and structures.**

Remote homology detection is a challenge for the computation biology. There is still a number of proteins with unknown function, and although structural properties can be useful to decrease this number, this information is not available for most proteins, and consequently, it cannot be used in large-scale approaches. In this scenario, it is necessary to develop intelligent strategies to address the remote homology detection problem, as those presented in chapters 4 and 5.

2.3 Protein families, domains and motifs

Homologous proteins can be organized into protein families. Proteins in a family typically have similar three-dimensional structures, functions, and some times significant sequence similarity. To organize homologous proteins in families can serve to extract important rules and to provide rich automatic functional annotation. For example, sequences within a protein family can be aligned to identify regions of similarity that may be a consequence of functional, structural, or evolutionary relationships between the sequences. Also, the evolution of these proteins can be studied by reconstructing a phylogenetic tree [62] that shows how proteins in this family have evolved.

At the functional level, proteins can be organized in *domains* or *motifs*. A domain is a part of the protein sequence which can fold into a stable structure independently on the rest of the sequence. Proteins considered as related often share the same domain(s). Domains are considered as building blocks and they may be recombined in different arrangements to create proteins with different functions. Many proteins consist of several structural domains, and they are called multi-domain proteins. Motif is a short stretch of amino acid sequence that potentially encode the function of proteins. Frequently, they are located inside protein domains.

There is a number of different classification systems to organize protein. They are based on different classification categories: (1) hierarchical protein families, such as: PIR-PSD [63] and ProtoMap [64], (2) families of protein domains such as Pfam [50], TIGRFAMs [65] and ProDom [66], (3) sequence motifs such as PROSITE [67] and PRINTS [68], (4) structural classes, such as SCOP [8] and CATH [69], and (5) integrations of various family classifications, such as iProClass [70] and InterPro [71]. These tools can be interrogated to provide the probable function for a query sequence (that is, a protein with unknown function). For this, computational approaches discussed in the next chapter are employed.

Chapter 3

Methods for remote homology detection

In this chapter, we review the main methods developed for addressing the problem of remote homology detection. Basically, these methods can be divided into three categories: methods that search for sequence similarity, described in section 3.1, generative methods that build a probabilistic model to represent each protein family and then evaluating each query sequence to see how well it fits the model, explained in section 3.2, and discriminative methods that model the difference between positive (homologous) and negative (non homologous) sequences to discriminate between members and non-members of a given protein family, reviewed in section 3.3. Finally, we compare these methods and discuss their performance and computational time (section 3.4).

3.1 Sequence Similarity Searching

Sequence Similarity Searching (SSS) is the most commonly employed computational tool for the protein annotation task. BLAST [9] and FASTA [10] are examples of such method. Although, both tools use different approaches, they adopt the same basic principle by locating short matches between two sequences. First, words of fixed length are extracted from the query sequence (that is, the sequence that one wishes to annotate). Second, exact matches (usually very short words) between the query sequence and sequences of a database with known proteins are identified. Third, best hits are extended to look for longer stretches of similarity, see Figure 3.1. In order to select the best hits, SSS uses a score system based on substitution score matrix, such as BLOSUM62 [72], and gap

penalty.

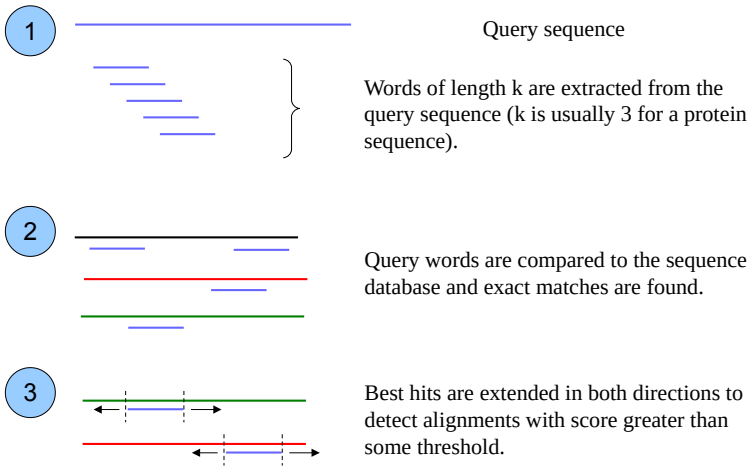


Figure 3.1: **Basic principle of the Sequence Similarity Searching.**

SSS optimizes speed over sensitivity. This property is essential and renders BLAST and FASTA the algorithms applicable to huge genome databases. Before these heuristic methods were introduced, query protein searches against huge databases were done with exhaustive search, through algorithms like the Smith-Waterman [73] one that revealed to be too time consuming. Contrary to the Smith-Waterman algorithm, SSS methods cannot guarantee optimal alignments. However, it provides a statistical significance score for pairwise alignments establishing a confidence level for the alignments. For this, it assigns to each alignment an expected score E (E-value) [74] that is computed with respect to the database. The E-value for an alignment x having a score s on the database D denotes the number of times that an unrelated sequence in D would obtain a score s_* higher than s by chance.

The performance of SSS methods is directly related to sequence identity of the query sequence and its homologs. These methods can be used to detect homology when sequence identity is greater than 30%, but they fail to give satisfiable answers when sequences are in “Twigthlight zone” (that is, when sequence identity is smaller than 30%). Thus, SSS methods are not expected to be suitable to remote homology detection.

3.2 Generative methods

An alternative to SSS are generative methods. First, they train a probabilistic model to represent a group of homologous sequences (a protein family), and then they match a query sequence against the model to evaluate the similarity of the query sequence to the group/family. Position-Specific Iterative BLAST (PSI-BLAST) [12], explained in section 3.2.1, and profile Hidden Markov Models (pHMMs) [11], described in section 3.2.2 are examples of such approaches, also known as family-based or sequence-profile based approaches. Finally, in the section 3.2.3 we present how the performance of generative methods can be improved by using domain co-occurrence information [20, 21, 22, 23, 24, 25].

3.2.1 Position-Specific Iterative BLAST

Position-Specific Iterative BLAST (PSI-BLAST) [12] is an iterative method that builds a probabilistic model based on Position Specific Score Matrix (PSSM) [75], and uses it to search for new matches in a subsequent iteration. In each iteration PSSM is updated by using sequences obtained in the previous iteration. The first PSSM is built from the multiple alignment of sequences extracted from BLAST results, while subsequent PSSMs are refined by using new matches. Figure 3.2 summarizes the PSI-BLAST flowchart. Note that, PSI-BLAST can be executed for a fixed number of iterations N or until convergence (when no new matches are found).

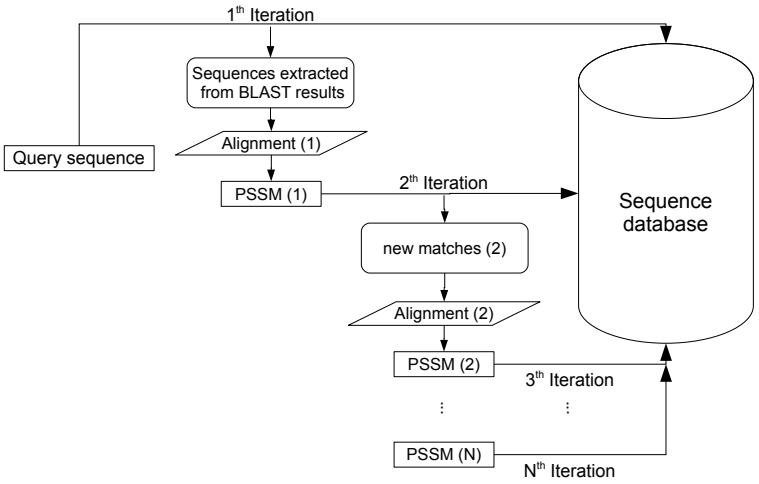


Figure 3.2: **PSI-BLAST flowchart.**

A PSSM is a probabilistic model built for a specific query sequence, and it can

be seen as an evolutionary model, since it describes the amino acid distribution for each query sequence position. PSSM assumes independence between positions in the query sequence, as it calculates scores at each position independently from the amino acids at other positions. In PSI-BLAST, a PSSM is used in place of a standard substitution matrix, such as BLOSUM62 [72]. It achieves better results because it defines for each amino acid in the query sequence a different rate of mutation (substitution, deletion and insertion) [76] that depends on its position in the query sequence.

3.2.2 Profile Hidden Markov Models

A Profile Hidden Markov Models (pHMM) [11] is a probabilistic model built from a multiple alignment of homologous sequences. A pHMM represents an alignment by using a sequence of nodes, usually one node per alignment position. Each node is composed of three states: match (M), insert (I) and delete (D) that represent the rate of mutation (substitution, deletion and insertion) respectively. Match states model conserved regions in the alignment, while insert and delete states model "indel" regions. Figure 3.3 shows a general architecture for a pHMMs, where squares (in blue), diamonds (in red) and circles (in yellow) represents match, insert and delete states respectively, and edges represent allowed state transitions. Note that, Begin and End states are included.

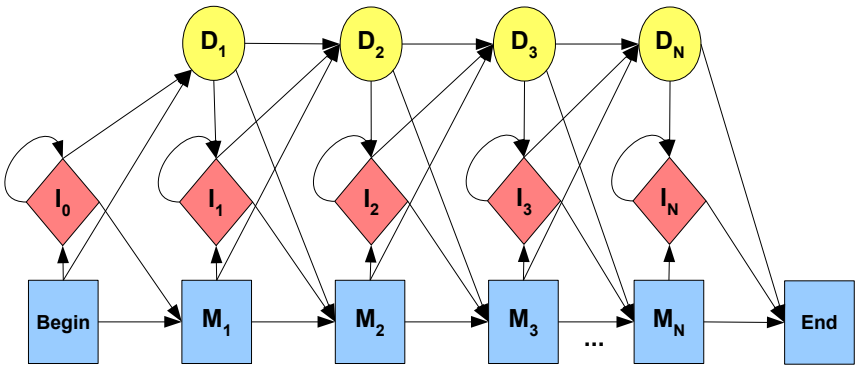


Figure 3.3: **General architecture for profile hidden Markov models.**

Profile HMMs have probabilities on two events: a transition from a state to another state (represented by edges in Figure 3.3), and the probability that a specific state will emit a specific character (say, a specific amino-acid when comparing proteins). Only match and insert states generate characters. Delete states are quiet. Therefore, each match and insert state has an emission probability distribution, for proteins this distribution will have 20

entries, one per amino acid. Transitions can define the architecture of the pHMM. Systems such as SAM [77] allow transitions between all types of states, totaling 3 transitions per state, hence 9 per node, as showed in Figure 3.3. This is not always the case, and the HMMER system [78], for instance, relies on the Plan7 model [78], which does not allows I $\rightarrow$ D and D $\rightarrow$ I transitions.

Performance of a pHMM critically depends on the quality of the estimated emission and transition probabilities. Emission probabilities are obtained by counting amino-acid frequencies at each alignment position. Unfortunately, the global alignment will usually have too few sequences to estimate all the parameters with sufficient confidence. Priors, such as mixtures of Dirichlets components [79] (a method for estimating probabilities of amino acids given small samples), is used to compensate for the small sample size and avoid over-fitting. A second major issue when estimating parameters is the relationship between the sequences themselves. Clearly, the information that a residue is better conserved across a number of very different sequences should carry more weight than the information the residue is conserved across a large number of very similar sequences. Most pHMMs thus include a sequence weighting step, which may be based on sequence trees [80], as in HMMER, or in entropy [81], as in SAM. In all cases, closer sequences carry less weight than more divergent sequences. Last, notice that the total weight of the sequences governs how much we trust the sequences versus the prior. Increasing the total weight of the sequence counts over the priors reinforces our trust in the sequence data, but may lead to over-fitting.

After the probability estimations, a pHMM can be used for aligning and scoring a query sequence with the model. For this, dynamic programming algorithms, such as Forward [82] and Viterbi [82] are employed. These algorithms compute the probability that a given query sequence S_n be matched by the pHMM $M(w)$, where n is the length of the query sequence, and w is the set of estimated parameters for the pHMM. In other words, they compute the maximum likelihood given by,

$$P(S_n|M(w)) = \sum_{\pi} P(S_n, \pi|M(w)), \tag{3.1}$$

where π is a sequence of consecutive states in the pHMM $M(w)$ (also called pathway) that starts in the Begin state, ends in the End state, and S_n is observed. The Forward al-

gorithm finds the maximum likelihood by considering partial probabilities $f_j(i)$ associated to sub-sequences $S_i = (x_1, \ldots, x_i)$ $(i \leq n)$. The probability $f_j(i)$ is obtained by summing probabilities of all possible state sequences that end at the j-th state, since S_i is observed. Thus, the maximum likelihood for S_n is computed recursivelly by the algorithm 1.

Algorithm 1 Forward

Require: $f_0(0) = 1, f_k(0) = 0$ $(k \neq 0)$
  for $i = 1 \rightarrow n$ **do**
    $f_j(i) = e_j(x_i) \sum_k f_k(i-1)a_{kj}$
  end for
Ensure: $P(S_n) = \sum_k f_k(n)a_{k0}$

where $e_j(x_i)$ is the emission probability for the character x_i provided by the state j, and a_{kj} is the transition probability from k to j states.

The Viterbi algorithm also provides pHMM's state sequence that better has recognized the query sequence, that is, it maps each amino acid in the query sequence to a match (M), insert (I) or delete (D) states. Thus, it computes both the maximum likelihood for a sequence S_n and the most probable pathway $\pi^* = argmax_\pi(x, \pi|M(w))$. Similar to the Forward algorithm, Viterbi uses the partial probability of observing S_i through the pathway π^* starting in the Begin state and ending in j-th state. The Viterbi algorithm is presented in 2. In order to produce the most likely pathway for S_n, Viterbi storages in the variable $ptr_i(j)$ a pointer for the previous state that achieved the best partial probability.

Algorithm 2 Viterbi

Require: $v_0(0) = 1, v_k(0) = 0$ $(k \neq 0)$
  for $i = 1 \rightarrow n$ **do**
    $v_j(i) = e_j(x_i)max_k(v_k(i-1)a_{kj})$
    $ptr_i(j) = argmax_k(v_k(i-1)a_{kj})$
  end for
Ensure: $P(S_n, \pi) = max_k(v_k(n)a_{k0})$
Ensure: $\pi_n^* = argmax_k(v_k(n)a_{k0})$

3.2.3 Methods based on domain co-occurrence

The majority of proteins are multidomains, domains do not form random combinations, and we observe fewer combinations than the statistically expected ones. This suggests functional cooperation, that is, two or more domains can interact to determine the protein function. Recently, this cooperation, also called domain co-occurrence, has been used

to improve the performance of domain recognition methods [20, 21, 22, 23, 24, 25]. In this section, we reviewed two of these methods that were applied to *P. falciparum* annotation: CODD (Co-Occurrence Domain Discovery) [24] and dPUC (domain Prediction using context) [25]. The performance of these methods is compared to the CASH method in chapter 5.

Both methods proposed to improve pHMM performance, and they were implemented on Pfam database [50], a collection of protein domains largely used for annotation task. They detect a set of potential Pfam domains $\mathcal{P}_t$, for a given protein sequence s_t, by setting a permissive Pfam threshold and by allowing overlaps. They pre-compute a list $\mathcal{L}$, containing domain pairs that present strong co-occurrence, directly from the list of domain architectures in Uniprot database [83]. CODD ranks $\mathcal{P}_t$ by ordering Pfam scores. It starts by assigning to s_t a set of domains $\mathcal{Q}_t$, that were obtained by Pfam. Then, it iteratively tries to add to $\mathcal{Q}_t$ a domain $d_i \in \mathcal{P}_t$ if d_i does not overlap with domains in $\mathcal{Q}_t$, and if it co-occurs with some domain in $\mathcal{Q}_t$ according to $\mathcal{L}$. On the other hand, dPUC represents $\mathcal{P}_t$ as nodes in a graph, where edges connect non-overlapping domains in s_t. It weights each node with normalized Pfam scores, and it weights edges with a special context score that captures the propensity of pairwise domains combinations in $\mathcal{L}$. Then, dPUC finds a set of domains for s_t by looking for the maximum-weighted clique in the graph.

3.3 Discriminative methods

Generative methods still largely fail to detect distant homologous proteins. In this scenario, discriminative approaches have emerged as an alternative to these methods, and have been able to attain additional accuracy to remote homology detection. The most popular discriminative method applied to the remote homology detection problem is Support Vector Machine (SVM) [84] revised in section 3.3.1. Some drawbacks of SVM, such as its incompatibility with relational data, have motivated researchers to explore other discriminative approaches, such as Inductive logic programming (ILP) [43]. In section 3.3.2, we revised some first-order logic concepts that are essential for the understanding of methods based on ILP. We shall describe two methods based on ILP that have been applied to remote homology detection problem [44, 45, 46, 47] .

3.3.1 Support vector machine

SVMs have been widely applied to remote homology detection [26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42]. In order to discriminate between members and non-members of a given protein family, SVMs learn the following classification function

$$f(x) = \sum_{i:x_i \in \mathcal{X}_+} \lambda_i K(x, x_i) - \sum_{i:x_i \in \mathcal{X}_-} \lambda_i K(x, x_i), \tag{3.2}$$

where $\mathcal{X}_+$ and $\mathcal{X}_-$ are positive and negative training examples, $K(x, x_i)$ is a function that measures the similarity between a pair of examples, and λ_i are non-negative weights computed during training by maximizing the function K also called kernel function. After the SVM training phase, a given query sequence x is predicted to be member (non member) of a protein family if the function $f(x)$ is positive (negative).

The classification function plays a key role in SVM performance. Typical approaches represent x as a fixed-length vector, where each vector's item is a protein property, and design a kernel function taking the inner product between these vectors. Several feature protein vector representations have been proposed. SVM-Fisher [26] represents each protein x as a vector of Fisher scores. These scores are obtained comparing x to the pHMM built from the positive training sequences (a protein family). SVM-pairwise [30] also uses scores to compose its feature vector, those are extracted from pairwise alignments of x and each sequence in the training set. Some methods use representations based on primary-sequence motifs, where a sequence x is represented in a vector space indexed by a set of pre-computed motifs [27]. GPkernel [39] is another method based on motifs, but instead of using pre-computed motifs coming from an existing database, it generates motifs from training data. Other methods have used structural motifs in place of primary-sequence ones for feature extraction task. The SVM-I-sites method [28] constructs the feature vector of a protein x by comparing the x profile (built by using PSI-BLAST) to the pre-computed I-sites library of local structural motifs. Later, this work was improved taking into account the order and relationship of the I-sites motifs [31]. A series of works have explored the use of k-mers (short subsequences of size k). Mismatch kernel [29] represents a sequence x as a vector of k-mers occurrence, that is, each vector position has a non-zero weight if the k-mer is present in x and zero weight otherwise. A k-mer is said to be present in x if x contains a substring that has at most n mismatches to the k-mer. Profile kernel [35] vector

representation is similar to the Mismatch kernel representation. However, it considers a k-mer to be present if x contains a substring whose PSSM-based ungapped alignment scores with the k-mer is above a user defined threshold. A feature vector representation based on distances between k-mers was introduced in [37]. Statistical and relevant features have been extracted from all possible k-mers (coming from training protein sequences) by using latent semantic analysis (an efficient feature extraction technique from natural language processing) in [38]. Later, this work was improved by using Top-n-grams that extracts from protein sequence frequency profiles [40].

Some approaches have followed a way that is alternative to the feature protein vector representation and pre-compute a kernel matrix where each element is the measure of similarity between a pair of examples. This matrix can be used directly as a kernel function. Some new tools have followed this way, such as SVM-LA [32], which measures the similarity between a pair of sequences by summing up scores obtained from its local alignment, and SW-PSSM [36], which uses profiles scoring schemes for measuring the similarity between pairs.

Most of SVMs are family-based, that is, a protein family is required to train them, and the aim is to classify unseen proteins as member or non-member of this family. Certainly, these methods are limited to the number of known families. In order to overcome this drawback, a new SVM category has been proposed, that is, pairwise SVM [41, 42]. Here, the aim is to rank proteins that are homologs to a given query protein. These methods are an alternative to the most commonly used methods in the biology community, that is: BLAST and PSI-BLAST.

In fact, in the SVM-HUSTLE [41] a training strategy was presented that could convert a family-based SVM into a pairwise SVM. Like PSI-BLAST it iteratively searches for homologs against a database by using BLAST in the first iteration. Next, it trains concurrent SVMs from positive sequences selected from BLAST output, and negative sequences selected randomly from the remaining database. Then, trained SVMs scan the database searching for new homologs that are added to the positive set. The algorithm stops when no new sample is classified as positive or when a maximum number of iterations is achieved.

To improve the performance some methods have applied the semi-supervised training

strategy, that is, they combine information from labeled (known proteins) and unlabeled (unknown proteins) databases in order to recruit more training sequences. This strategy is generally applied when unlabeled data is abundant while labeled data is limited, and this is the actual scenario of protein classification, since there is a large group of still unannotated proteins. However, semi-supervised approaches can become computationally hard when unlabeled large databases are used, such as the non-redundant protein database (NR). Among methods that employ this strategy are RANKPROP [85], SVM-HUSTLE [41], Top-N-Gram [40] and SW-PSSM [36].

3.3.2 Inductive logic programming

ILP is a relational data-mining method that uses first-order logic predicates to represent background knowledge, theories and examples (positives and negatives). From those an induction system can learn a hypothetical logic program which entails all the positive and none of the negative examples. This logic program is a comprehensible set of logical rules that can be used to classify unseen examples. We start this section by reviewing some basic concepts of first-order logic predicate that is essential to understand ILP systems, and then we present two methods that employ ILP to address the remote homology detection problem.

First-order logic concepts

First-order logic, also called predicate logic, represents logic sentences in a more sophisticated way than propositional logic. For example, consider the following sentences: "the amino acid i is hydrophobic" and "the amino acid m is hydrophobic". In propositional logic these sentences are treated as two unrelated propositions. On the other hand, the first-order logic can related them creating the predicate hydrophobic(X), which asserts that the amino acid represented by the variable X is hydrophobic. First-order logic allows us to define relations about properties that are shared among objects. For example, we can observed from Table 4.3 that a tiny amino acid is also a small amino acid. Then, we can denote this relation by using the logical rule R_a: $\forall(X)(\text{tiny}(X) \rightarrow \text{small}(X))$, where the symbol $\rightarrow$ is a logic connective used to denote a conditional (if/then) statement, and the symbol $\forall$ ("for all") is the universal quantifier symbol. The other quantifier is the $\exists$

("there exists") called existential quantifier. The part of R_a before connective $\rightarrow$ is called antecedent and the part after is called consequent. The standard logic connectives are $\wedge$ for conjunction, $\vee$ for disjunction, $\rightarrow$ for implication, $\leftrightarrow$ for bi-conditional and $\neg$ for negation. Next, we define some syntax rules for the first-order logic language. A variable (X,Y and W in Table 4.3) is a term. If $t_1, \ldots, t_n$ are terms and f is a function symbol then $f(t_1, \ldots, t_n)$ is a term, where n is arity (number of arguments) of the function, and $n \geq 0$. A function of zero arity ($n = 0$) is a constant ($c, k, cg, 24, 27, \ldots$ in Table 4.7). If $t_1, \ldots, t_n$ are terms and p is a ($n \geq 0$) predicate symbol then $p(t_1, \ldots, t_n)$ is an atomic formula, also called here predicate. More complex formulae can be built using the logical connectives and quantifiers. A substitution $s = X/i$ is an assignment of term i to variable X. For example, when s is applied to the predicate hydrophobic(X) an instantiation of the predicate hydrophobic(i) is created. A ground predicate is a predicate without any variables.

ILP approaches applied to remote homology detection

To the best of our knowledge, researchers have developed two approaches for applying ILP to remote homology detection. The first method is known as *Homology Induction* (HI) [44, 45] and uses ILP to improve on conventional sequence-based homology method. First, a standard method for homology detection such as PSI-BLAST is run to find homologous sequences for a given query sequence. Second, HI learns rules which are true for sequences of high similarity to the query sequence and false for general sequences (assumed to be non homologous to the query sequence). Next, the rules learnt by HI are used to discriminate sequences in the twilight zone between the homologous and non-homologous. HI employs a logical representation based on protein sequence properties such as physico-chemical conservation, relative molecular weight, amino acid frequency (taken alone or in pairs), and others. HI was compared with PSI-BLAST achieving best performance. However, the method is bound to the performance of standard method chosen in the first step.

The second method uses a hybrid ILP-propositional machine learning method to predict protein functional classes directly from sequences [46, 47]. First, it represents each sequence in a homologous group through first-order logic predicates. It creates predicates based on properties extracted directly from sequences, such as frequency distribution of

single residues, and on properties predicted from sequences such as secondary structure elements. Second, it uses WARMR [86], an ILP data-mining program, to identify the most frequent patterns for the homologous group. Third, it converts these frequent patterns (logical rules) into binary attributes to be used in propositional learning. For each query sequence, to be represented, a binary attribute takes the value 1 if the corresponding logical rule succeeds, and it takes the value 0 otherwise. Finally, these attributes are used to train decision trees (DTs) [87] that are later used to discriminate between members and non-members of the homologous groups.

3.4 Comparing different methods

Methods based on only sequence similarity searching are still the most used to detect homology and to transfer annotation in proteins. Although, they do not achieve good performance on remote homologous proteins, their heuristics provide a feasible computational time. Generative methods, that previously compute a probabilistic model for each protein family, are seen as an alternative to the traditional methods, since their computational time remains acceptable while better performance is achieved. However, they still fail in detecting remote homologous proteins. On the other hand, discriminative methods that model the difference between members and non-members of given protein family are more effective for remote homology detection, but their computational time make them not applicable to the large-scale protein annotation and classification. In conclusion, remote homology detection problem is still a challenge for computational biology and there is not a method that works well in all cases.

Chapter 4

ILP-SVM Homology

In this chapter we describe the ILP-SVM Homology method, a computational framework for remote homologous protein classification that combines inductive logic programming and propositional models. In section 4.1, we present our motivation for the combination of different models and introduce the framework. In section 4.2, we describe the framework and in section 4.3 we present and discuss the results. The conclusion of this work is presented in section 4.4.

4.1 Background

SVM methods (reviewed in section 3.3.1) are among the most effective and accurate methods for solving the remote homology detection problem. They classify query sequences as member or non-member of homologous proteins, but they do not provide any insight to the user concerning the reasons for the separation. Moreover, SVM is not able to work directly over relational data. However, biological data is naturally relational. For example, a specific amino acid in a protein could belong to an α-helix and at the same time belong to the active site of that protein. Therefore, methodologies that explore relational data are expected to be more suitable to deal with biological data. In this vein, we focus our attention on Inductive Logic Programming (ILP) [43], see section 3.3.2. Through ILP we can relate properties of the homologous protein, and represent background knowledge for a protein family. From these an ILP system can learn a hypothetical logic program that summarizes essential rules that can help to understand what determines a protein function. Remote homologous proteins seem to share only the essential properties in order to keep their function, and these properties can be represented by first-order logic predi-

cates. For instance, Figure 4.1 shows the partial alignment of "Glucocorticoid receptor-like (DNA-binding domain)" superfamily sequences. The sequence identity of this alignment is smaller than 30%. We can observe that some positions are completely conserved (marked by $*$). Also, there are positions which are partially conserved (marked by $\bullet$). Methods that have the ability of exploring only these positions most likely will outperform the methods that consider the whole alignment, since non-conserved positions could add noise to the model. For these reasons, we believe that the combination of these two approaches can improve the performance of remote homology detection.

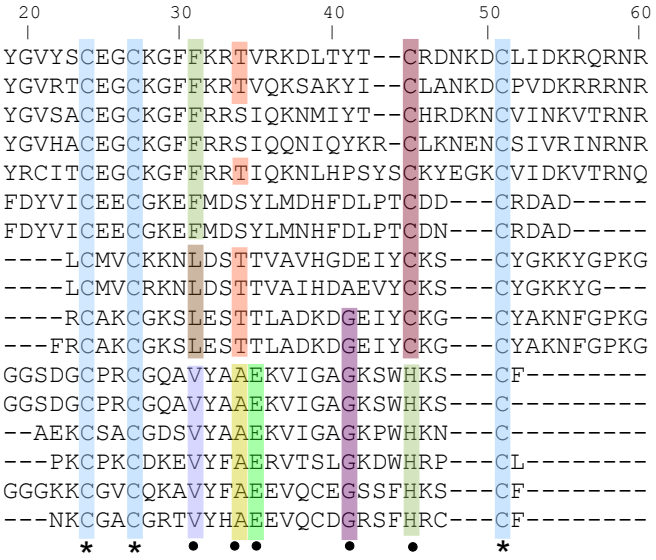


Figure 4.1: **Partial alignment of "Glucocorticoid receptor-like (DNA-binding domain)" superfamily sequences.**

As described before, ILP has already been applied to the protein classification task [45, 47, 44, 46], and we shall review these methods in section 3.3.2. Methods presented in [46, 47] have combined ILP and propositional models (Decision Trees) to predict protein functions. Our work is based on the same basic approach. However, there are significant differences. First, we have proposed a novel first-order logical representation based on conserved amino acid positions in a multiple sequence alignment (MSA). Second, we have related the first-order logical representation based on sequence properties, proposed in [46, 47, 88], with our novel representation based on conserved positions for creating a hybrid representation that takes into account conserved physico-chemical positions in a MSA. Third, we have joined features created by these representations to train propositional models. In a general way, this combination of features has improved the performance of

models. Fourth, we have proposed to use SVM as propositional machine learning method rather than DTs.

An overview of our methodology is showed in Figure 4.2. It is divided into two parts: a training phase (Figure 4.2-A) and a test phase (Figure 4.2-B). In the training phase, each sequence in the positive training set is represented through first-order logic predicates. WARMR [89], an ILP system, learns logical rules on the set. These rules are converted into binary attributes in order to train propositional models; this step is called propositionalization. Next, each sequence in the positive and negative training set is represented through binary attributes, and finally propositional models, such as DTs or SVM, are trained. In the test phase, each sequence in the positive and negative test set is represented through binary attributes that correspond to the logical rules learned during the training phase. Next, the propositional model is tested and its output is divided into sequences classified as positives and sequences classified as negatives.

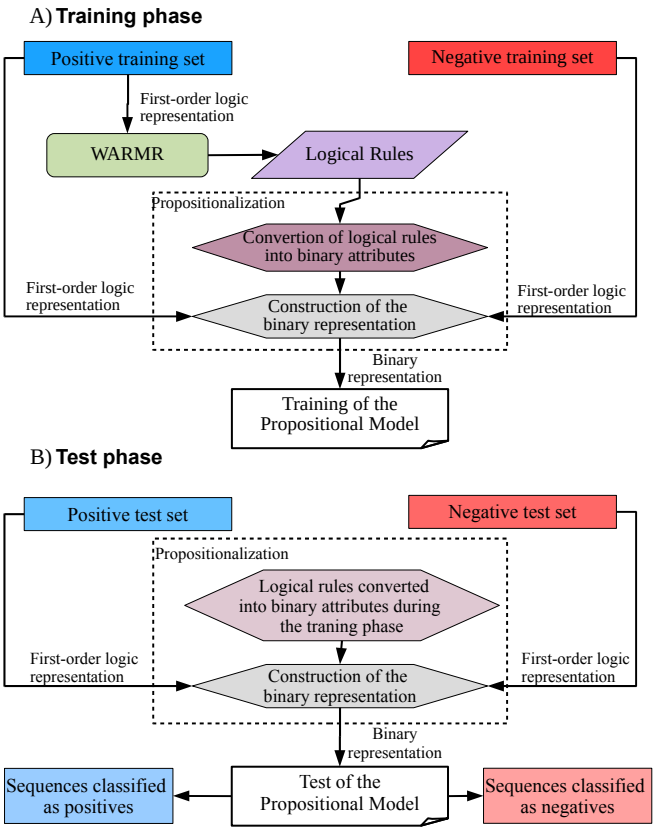


Figure 4.2: **ILP-SVM-Homology flowchart.**

4.2 Methods

Here, we present our approach in detail. First, we describe the benchmark used for performing our experiments (section 4.2.1). Second, we present the first-order logical representations for protein sequences (section 4.2.2), a step that is essential for using ILP systems. We present three kinds of logical representations. The first, named sequential, is based on properties coming directly from sequences. This representation being already proposed in previous works [46, 47, 88]. The second, named alignment, is based on conserved amino acid positions in a MSA. In the third representation, we related the first two representations creating a new one that takes into account conserved physico-chemical positions in a MSA as well. To the best of our knowledge the second and the third logical representations have been proposed here for the first time. Third, we present WARMR, the ILP system used here to learn logical rules, and explain how these rules are converted into binary attributes to train propositional models. Fourth, we describe the methodology used to assess and compare different methods (section 4.2.4). Finally, we discuss parameter settings and tools used in this work (section 4.2.5).

4.2.1 Dataset description

In order to evaluate our methodology we used a common superfamily benchmark, that is the SCOP database [8]. SCOP is a reference dataset for evaluating the performance of remote homology detection methods [38, 40, 19, 32, 37]. SCOP classifies all protein domains of known structure into a hierarchy with four levels: class, fold, superfamily, and family. In our study, we work at the superfamily level: it groups families for which a common evolutionary origin is not easily deduced from sequence identity, but rather from an analysis of structural and functional features. To provide a good comparability with previous approaches, we used the same database version used in [40, 37, 38, 32, 30, 28]. It contains 54 families and 4352 proteins selected from SCOP version 1.53. All protein sequences were extracted from the Astral database [90] and all pairwise alignments have E-value no greater than 10^{-25}.

We adopted the leave-one-family-out experimental methodology, as used in previous works. Thus, the sequences of each SCOP family are taken as positive test samples, and

the proteins outside the family but within the same superfamily are taken as positive training samples. Negative samples are selected from outside of the superfamily and are separated into training and test sets. Previous works have considered random samples by splitting the *remaining SCOP database* (that is, SCOP minus the positive dataset) into training and test respecting the same ratio as the positive samples. This strategy produces unbalanced datasets: negative instances far outnumber the positive instances. For example, for the family test set "Nuclear receptor" in the "Glucocorticoid receptor-like (DNA-binding domain)" superfamily, there are 20 positive training sequences and 3204 negative training sequences. We show the complete list with the distribution of positive and negative samples in Table 4.1. If an unbalanced dataset is used to train a classifier, this latter will tend to predict that most of the incoming data belong to the majority class, that is the negative class. As a result, it would present poor predictive accuracy over the minority class, that is the positive one. To the best of our knowledge, previous works do not use a performance measure that evaluates the predictive accuracy over the minority class. They have used the area under the ROC curve (AUC-ROC) as performance measure, and AUC-ROC can present an excessively optimistic view on the algorithm performance when there is a large difference between positive and negative sample distributions [91]. We showed this behavior in Table 4.2, where the AUC-ROC presents higher values than the area under the Precision-Recall curve (AUC-PR). Moreover, methods as "ILP-SVM-Seq" and "ILP-SVM-Aln_{cons}", appear to be comparable in ROC Space, while in PR space, "ILP-SVM-Aln_{cons}" has a clear advantage over "ILP-SVM-Seq".

Table 4.1: **Distribution of positive and negative samples for the original unbalanced database.**

SCOP Family	Tr^+	Tr^-	Te^+	Tr^-
7.3.5.2	12	2330	9	1746
2.56.1.2	11	2509	8	1824
3.1.8.1	19	3002	8	1263
3.1.8.3	17	2686	10	1579
1.27.1.1	12	2890	6	1444
1.27.1.2	10	2408	8	1926
3.42.1.1	29	3208	10	1105
1.45.1.2	33	3650	6	663
1.4.1.1	26	2256	23	1994
2.9.1.2	17	2370	14	1951
1.4.1.2	41	3557	8	693
2.9.1.3	26	3625	5	696
1.4.1.3	40	3470	9	780
2.44.1.2	11	307	140	3894
2.9.1.4	21	2928	10	1393
3.42.1.5	26	2876	13	1437
3.2.1.2	37	3002	16	1297
3.42.1.8	34	3761	5	552
3.2.1.3	44	3569	9	730
3.2.1.4	46	3732	7	567
3.2.1.5	46	3732	7	567
3.2.1.6	48	3894	5	405
2.28.1.1	18	1246	44	3044
3.3.1.2	22	3280	7	1043
3.2.1.7	48	3894	5	405
2.28.1.3	56	3875	6	415
3.3.1.5	13	1938	16	2385
7.3.10.1	11	423	95	3653
3.32.1.11	46	3880	5	421
3.32.1.13	43	3627	8	674
7.3.6.1	33	3203	9	873
7.3.6.2	16	1553	26	2523
7.3.6.4	37	3591	5	485
2.38.4.1	30	3682	5	613
2.1.1.1	90	3102	31	1068
2.1.1.2	99	3412	22	758
3.32.1.1	42	3542	9	759
2.38.4.3	24	2946	11	1349
2.1.1.3	113	3895	8	275
2.1.1.4	88	3033	33	1137
2.38.4.5	26	3191	9	1104
2.1.1.5	94	3240	27	930
7.39.1.2	20	3204	7	1121
2.52.1.2	12	3060	5	1275

Continued on next page

SCOP Family	$\mathbf{Tr}^+$	$\mathbf{Tr}^-$	$\mathbf{Te}^+$	$\mathbf{Tr}^-$
7.39.1.3	13	2083	14	2242
1.36.1.2	29	3477	7	839
3.32.1.8	40	3374	11	927
1.36.1.5	10	1199	26	3117
7.41.5.1	10	2241	9	2016
7.41.5.2	10	2241	9	2016
1.41.1.2	36	3692	6	615
2.5.1.1	13	2345	11	1983
2.5.1.3	14	2525	10	1803
1.41.1.5	17	1744	25	2563

Table 4.2: **AUC-ROC and AUC-PR for ILP-SVM models trained from the original unbalanced database.**

	AUC-ROC	AUC-PR
ILP-SVM-Seq	0.81	0.10
ILP-SVM-Aln_{cons}	0.83	0.21
ILP-SVM-Seq-Aln_{cons}	0.85	0.22

Our analysis of protein sequence-identity in this unbalanced database, see Figure 4.3-A (right), shows that around 46% of protein pairs have at least 30% of sequence-identity. Also, around 25% have sequence-identity between 90 and 100%, Figure 4.3-A (left). Moreover, we observed a bias in the composition of negative and positive classes: pairs of sequences in the negative set have, on average, higher sequence-identity than pairs of sequences in the positive set. This average is 22% for positive sequences, 4.3-B (right), against 57% for negative sequences, 4.3-B (left). We argue that this unbalanced database is not appropriate to evaluate the performance of remote homology detection methods, mainly because negative sequences are not into the *Twilight Zone*. Thus, we adopted a new experimental methodology to train and to test discriminative methods applied to the remote homology detection problem. The positive samples were taken as before, that is, within a SCOP superfamily. However, several negative samples were constructed by randomly selecting sets of sequences from the remaining SCOP database of size that is comparable to the size of the positive set. We constructed as many negative samples as it is needed to statistically cover the remaining SCOP database. For this, let

Tr^+ and Te^+ be sizes of positive training and positive test sets, respectively. Also, let $D^* = D - (Tr^+ + Te^+)$ be the size of the remaining SCOP database, where D is the total number of sequences in the SCOP database. Thus, we repeated the random selection of negative samples T times, where T is given by equation 4.1.

$$T = \lfloor D^*/min(Tr^+, Te^+) \rfloor, \tag{4.1}$$

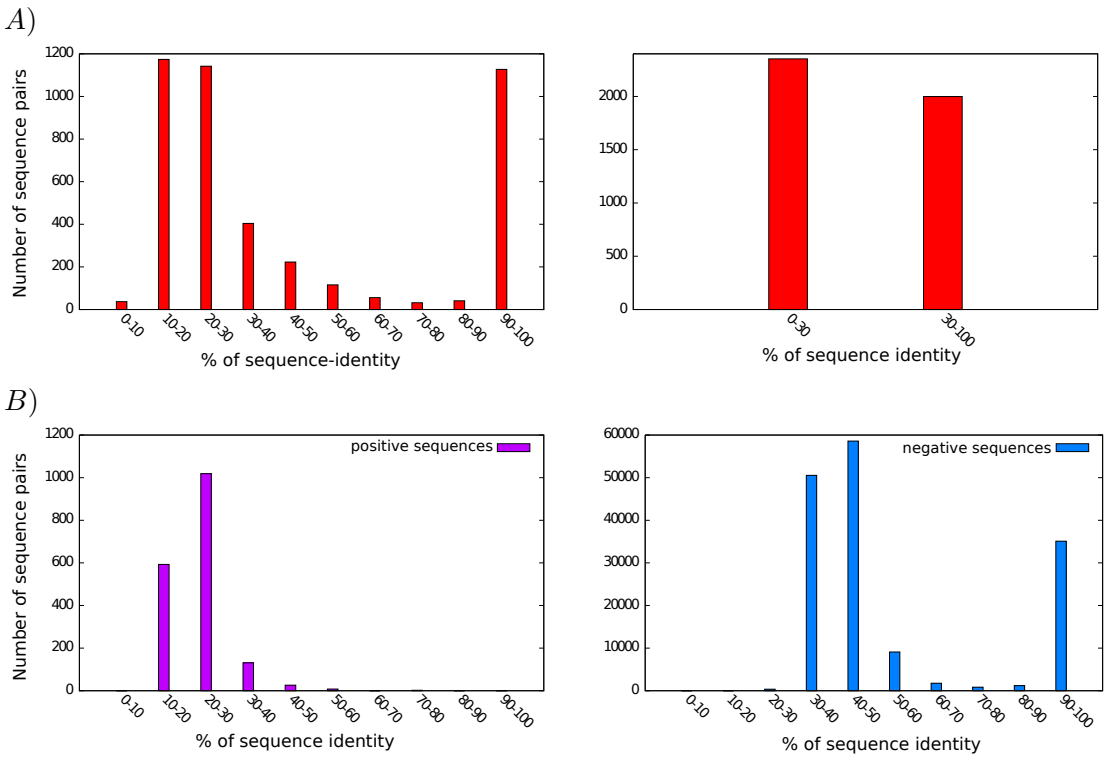


Figure 4.3: **Analysis of sequence-identity for the original unbalanced database.**

In order to examine more systematically the performance of remote homology detection methods, we produced a database of sequences from the original one getting only sequences with identity smaller than 30%. It is named S_{30} and the original database S_{full}. S_{30} contains 25 families and 2362 sequences.

4.2.2 Logical representations

In order to use ILP systems, such as WARMR, first we have to represent each training and test examples as relational data. Good ILP overviews, including first-order logic concepts, can be found in [43, 92, 93]; we describe them briefly in the section 3.3.2. We created

three kinds of predicates, the first, called sequential predicates, represent each protein in terms of its physico-chemical properties and the frequency of their amino acids (taken alone or in pairs). The second, called alignment predicates, are based on conserved amino acid positions within a protein MSA. Additionally, we related both predicates to represent conserved physico-chemical positions within a protein MSA. The next sections explain them in detail.

Sequential predicates

The sequential predicates are based on properties that can be calculated directly from sequences. These include groups of amino acids that share some physico-chemical properties as done in [88], see Table 4.3 from property 1 to 16. Additionally, we created predicates to represent the distribution for singles and pairs of residues as done in [44, 45, 46, 47], showed in Table 4.3 properties 17 and 18. All predicates used in this study are listed in Table 4.3, where X is the sequence identifier, Y is the percentage of amino acids with some physico-chemical property. For the predicate *aminoacidRatio*(X, W, Y), X is the sequence identifier, W is an amino acid, and Y is the percentage of amino acid W within sequence X. For the predicate *aminoacidPairRatio*(X, W, Y), X and Y are defined as before, and W is a pair of amino acids. The variable Y can assume only numerical values, however ILP systems such as WARMR are not very suitable for handing with numerical values. To overcome this limitation we map each percentage value Y to $\lfloor Y/10 \rfloor + 1$, as done in [46, 47].

Alignment predicates

Additionally, we created a predicate based on conserved positions in a protein MSA. The predicate that represents each alignment position is $col(X, W, Z)$, where X is the sequence identifier, Z is the alignment position where the amino acid W belongs. To illustrate how these predicates are created, see Figure 4.4. The sequences in the positive training set (S_+) are aligned and a ground predicate is created for each amino acid in each alignment position. For example, the ground predicate $col(s_1, v, 1)$ means that the sequence s_1 has the amino acid v in the first alignment position.

The creation of these ground predicates for the sequences of the positive training

Table 4.3: **Sequential Predicates.**

Property {amino acid set}	**Predicate**
1- small {A,G,S,T}	small(X,Y)
2- polar {D,E,H,K,N,Q,R,S,T,W,Y}	polar(X,Y)
3- polar uncharged {N,Q}	polarUncharged(X,Y)
4- aromatic {F,H,W,Y}	aromatic(X,Y)
5- charged {D,E,H,I,K,L,R,V}	charged(X,Y)
6- positively charged {H,K,R}	positivelyCharged(X,Y)
7- negatively charged {D,E}	negativelyCharged(X,Y)
8- tiny {A,G}	tiny(X,Y)
9- bulky {F,H,R,W,Y}	bulky(X,Y)
10- aliphatic {I,L,V}	aliphatic(X,Y)
11- hydrophobic {I,L,M,V}	hydrophobic(X,Y)
12- hydrophilic basic {K,R,H}	hydrophilicBasic(X,Y)
13- hydrophilic acidic {E,D,N,Q}	hydrophilicAcidic(X,Y)
14- neutral weakly hydrophobic {A,G,P,S,T}	neutralWeakHydrophobic(X,Y)
15- hydrophobic aromatic {F,W,Y}	hydrophobicAromatic(X,Y)
16- acidic {E,D}	acidic(X,Y)
17- amino acid ratio	aminoacidRatio(X,W,Y)
18- amino acid pair ratio	aminoacidPairRatio(X,W,Y)

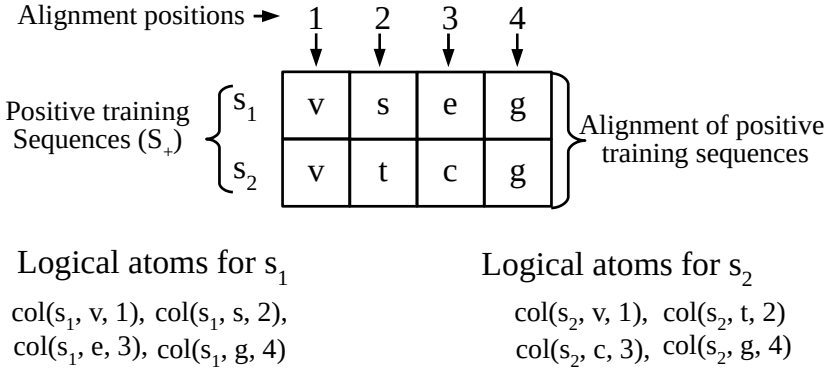


Figure 4.4: **Creating ground predicates from alignment positions.**

set is a trivial task, since they can be extracted directly from the MSA. However, how can we create ground predicates for the query sequences? We aim to find out how a query sequence is aligned in respect to the positive training alignment (homologous proteins). If the query sequence is closely matched to the positive training alignment, probably this suggests much higher conservation than for a query sequence weakly matched. Therefore, we must fix the positive training alignment and align a query sequence against it, that is, we aligned each query sequence with the consensus sequence of the positive training

alignment. To do this, we built a pHMM (see section 3.2.2) from the positive training alignment, and used it for matching the query sequences.

To illustrate the logical representation, observe Figure 4.5. First, the positive training sequences are aligned (Figure 4.5-A). Second, a pHMM is built from this alignment. Note that, each alignment position corresponds to a pHMM state: match (M), insert (I) and delete (D), see Figure 4.5-B. Third, Viterbi algorithm (see 3.2.2) is used to align a query sequence q_1 with the pHMM. Viterbi provides the pHMM's state sequence that better has recognized the query sequence (Figure 4.5-C). Since, we know the mapping between alignment positions and pHMM states we can create ground predicates for q_1 in a similar way to Figure 4.4, see Figure 4.5-D.

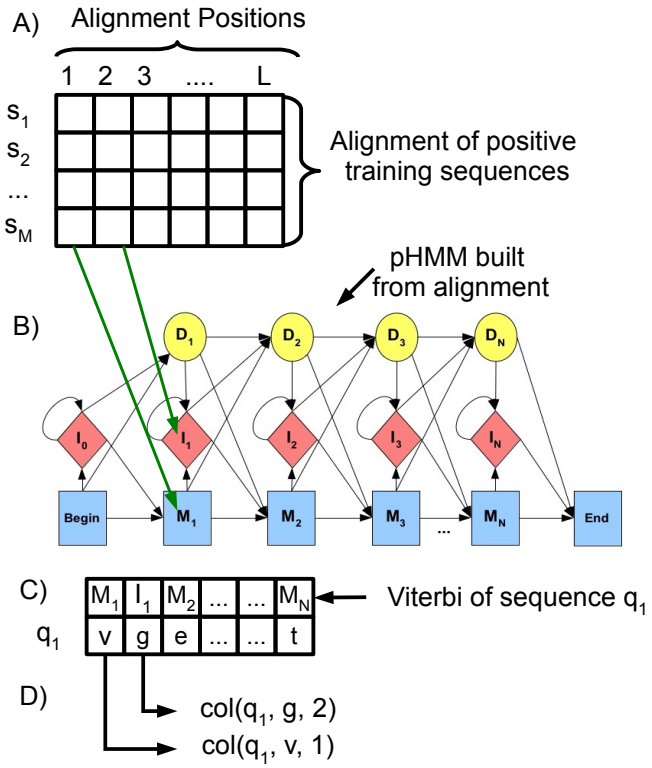


Figure 4.5: **Using pHMMs to create ground predicates for alignment positions.**

Relating sequential and alignment predicates

Through first-order logic, new knowledge statements can be extracted from data relations. For example, observe position 34 in the alignment shown in Figure 4.1. All amino acids in this position are small (see Table 4.3) thus, we can learn a logical rule that relates position

34 to small amino acids (see R_6 in Table 4.7). This rule allows us to introduce the new concept of "conserved physico-chemical position" in a MSA.

4.2.3 Construction of propositional classifiers

In our approach we aim to build models that will be able to explore the most frequent patterns in the homologous protein datasets. As a first step we run WARMR program to learn these most frequent patterns on the positive training set. The WARMR algorithm discovers frequent patterns on databases applying an extended version of APRIORI algorithm [94]. WARMR learns association rules over multiple relations in relational datasets. Basically, WARMR algorithm works as a filter on all possible rules selecting, for example, those rules with *confidence* above a threshold. The confidence of an association rule is a percentage value that shows how frequently the rule occurs. In other words, the confidence value indicates how reliable this rule is. As a second step (propositionalization step), we converted each rule learned by WARMR into a binary attribute (feature) for the training of propositional learning methods. An attribute a_i has value *1* for a specific protein sequence if the corresponding query r_i succeeds, and *0* if the query fails. Finally, we trained two propositional models from these attributes, DTs and SVMs.

4.2.4 Comparison between different methods

To statistically analyze remote homology detection methods, we run them several times over the same positive set, but over several negative sets. In each run the number of positive samples is equal to the number of negative samples. Since datasets are now balanced curves in the AUC-PR space are similar to curves in the AUC-ROC space [91]. Therefore, we just show AUC-ROC as the classification accuracy measure. For each protein family, the AUC-ROC score was averaged over T runs (see equation 4.1), and the overall performance of each method was averaged over all families.

4.2.5 Parameter settings and tools used

We used CLUSTALW [95] version 2.0.10 in order to provide the positive training alignments. The CLUSTALW parameters were kept at default. We used HMMer [78] version 2.3.2 (default parameters) to build the pHMMs from positive training alignments. These pHMMs were used to score query sequences and their output was used to construct logi-

cal representations based on alignment positions. In order to learn logical rules we used WARMR. The confidence parameter (c%) of the WARMR filters the most frequent patterns, that is, only those with frequency above c% are considered. We tested several threshold values for c% and the best results were obtained with 25% for logical representations based on sequential properties and based on conserved amino acid positions, and 50% for the representation based on conserved physico-chemical positions.

Next, the rules generated by these representations were converted into binary attributes for training propositional models. We have created two kinds of propositional models: DT and SVM. For SVM we have used the publicly available Gist SVM package version 2.1.1 `http://svm.sdsc.edu`. We used radius basis function as kernel function and other parameters by default, and DT models were built using the WEKA software (default parameters) available in `http://www.cs.waikato.ac.nz/~ml/weka/index_downloading.html`. In order to compare our approach with state of the art methods, we consider SVM-LA [32], SVM-Ngram-LSA [38], PSI-BLAST and HMMer-3.0. SVM-LA is a complex method kernel that defines a similarity measure between protein pairs by summing up scores obtained from their local alignment. The SVM-LA parameters were kept as default. SVM-Ngram-LSA extracts N-gram from protein sequences and uses them to train a SVM model. To consider only the most significant N-grams it applies Latent Semantic Analysis (LSA), which is a feature extraction technique from natural language processing. We downloaded SVM-Ngram-LSA from `http://www.insun.hit.edu.cn/news/view.asp?id=413` and used it with parameters described in [38]. HMMer-3.0 was trained from MSAs produced by CLUSTALW, and all parameters were kept as default. PSI-BLAST was ran on two configurations: in the first, we used the same dataset used to train the other methods and 4 iterations; in the second we used nrdb90 and 20 iterations. We also considered to compare to Top-N-gram [40], a recent work that applies SVM to the remote homology detection problem. However, the program was unavailable. We used chi-square as a feature selection approach, and we set the parameter δ to 0.05 and 0.25 values, as done in [89], δ specifies the confidence level for the chi-square test selection. We carried out rank-sum test [96] to compare the curves showed in Figure 4.6.

Table 4.4: **Average AUC for S_{full} and S_{30} databases.**

Methods	$\mathbf{S_{full}}$	$\mathbf{S_{30}}$
ILP-SVM-Seq	0.79	0.77
ILP-SVM-$\text{Aln}_{\mathbf{cons}}$	0.81	0.81
ILP-SVM-$\text{Aln}_{\mathbf{ps}}$	0.80	0.81
ILP-SVM-Seq-$\text{Aln}_{\mathbf{cons}}$	0.85	0.80
ILP-SVM-$\text{Aln}_{\mathbf{cons}}$-$\text{Aln}_{\mathbf{ps}}$	0.82	0.82
ILP-SVM-Seq-$\text{Aln}_{\mathbf{cons}}$-$\text{Aln}_{\mathbf{ps}}$	**0.87**	**0.82**
ILP-DT-Seq	0.67	0.65
ILP-DT-$\text{Aln}_{\mathbf{cons}}$	0.70	0.69
ILP-DT-$\text{Aln}_{\mathbf{ps}}$	0.68	0.67
ILP-DT-Seq-$\text{Aln}_{\mathbf{cons}}$	0.72	0.69
ILP-DT-$\text{Aln}_{\mathbf{cons}}$-$\text{Aln}_{\mathbf{ps}}$	0.71	0.71
ILP-DT-Seq-$\text{Aln}_{\mathbf{cons}}$-$\text{Aln}_{\mathbf{ps}}$	0.74	0.71
SVM-Ngram-LSA	0.79	0.77
SVM-LA	**0.87**	0.80
PSI-BLAST	0.75	0.69
HMMer-3.0	0.63	0.60

4.3 Results and Discussion

In order to assess our methodology, we have trained DTs and SVMs using representations described in Methods (section 4.2.2). We called *Seq* those models that are trained from sequential properties only, and we named Aln_{cons} those models that are trained from conserved amino acid positions in a MSA. We have created a hybrid representation that takes into account conserved physico-chemical positions in a MSA (see R_6 in Table 1), the resulting models were called Aln_{pc}, where *pc* is an abbreviation for physico-chemical properties. Additionally, we created models by joining *Seq*, Aln_{cons} and Aln_{pc} features. We named ILP-SVM and ILP-DT models trained from our methodology. Table 4.4 summarizes results (see also Figure 4.6 that shows only ILP models with best performance). ILP-DT models did not reach good performance on S_{full} and S_{30} databases (see Methods). ILP-SVM-*Seq*-Aln_{cons}-Aln_{pc} models outperformed all other ILP methods for both databases.

We highlighted that the novel logical representation, based on conserved amino acid positions (Aln_{cons}) and based on conserved physico-chemical positions (Aln_{pc}) in MSA, that we propose here, is able to achieve better prediction accuracy than the sequential

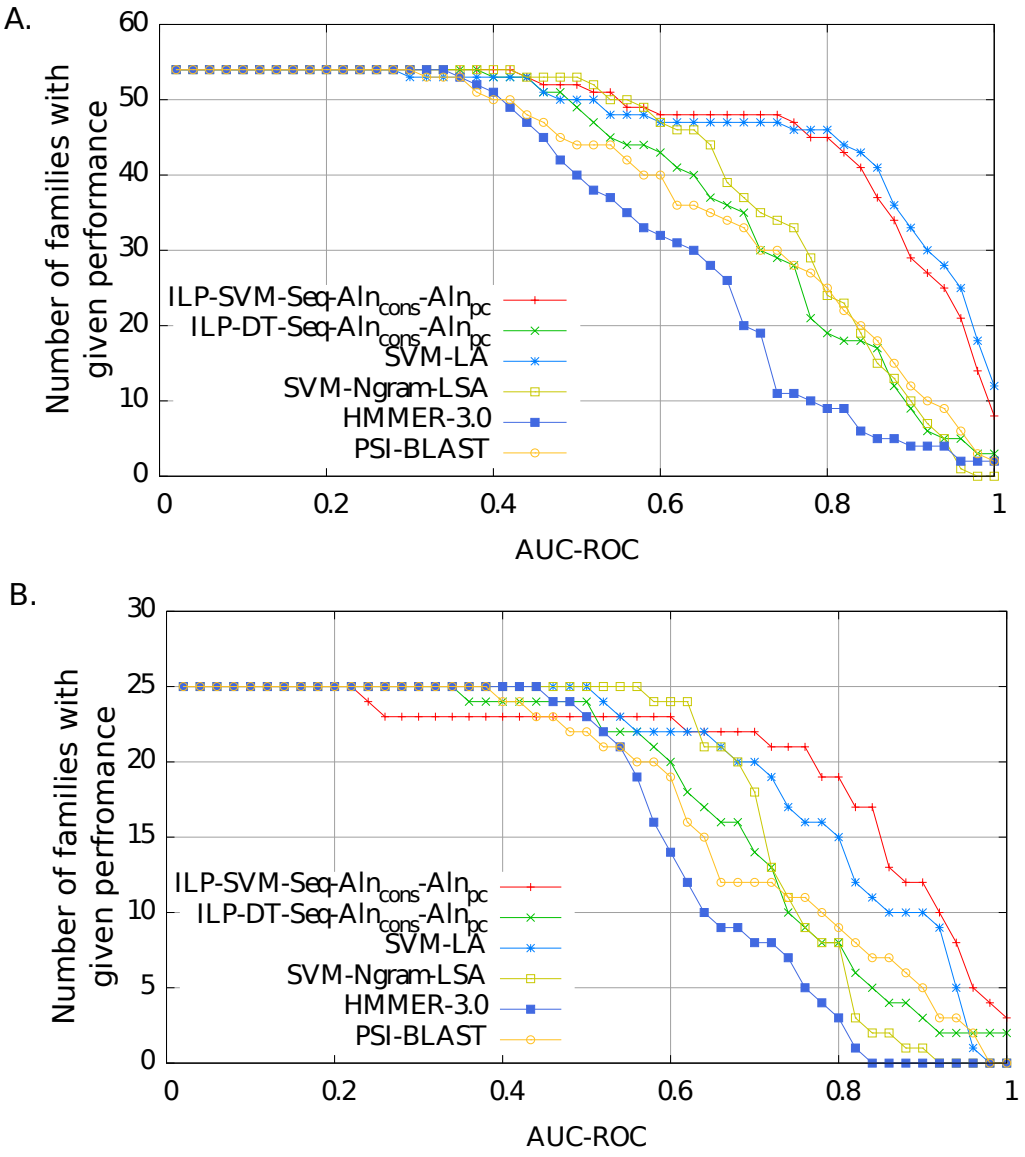


Figure 4.6: **Performance as measured by AUC-ROC.**

logical representation commonly used by related works. This result is expected since it is known that MSAs contain more functional and structural information than properties extracted from unaligned sequences. On the one hand, the combined models (*Seq*-Aln_{cons}, Aln_{cons}-Aln_{pc} and *Seq*-Aln_{cons}-Aln_{pc}) were able to attain better accuracy than single models (*Seq*, Aln_{cons} and Aln_{pc}) for the S_{full} database. On the other hand, for the S_{30} database, the sequential logical representation does not contribute to improve the model performances. Based on the observation that MSA information is richer than sequential properties, we tested the hypothesis that below 30% of sequence-identity, MSA information is still rich enough to build accurate models and that sequential properties might add noise within combined models. In fact, we observed that ILP-SVM-Aln_{cons} outperformed ILP-SVM-*Seq*-Aln_{cons}, while ILP-SVM-*Seq*-Aln_{cons}-Aln_{pc} and ILP-SVM-Aln_{cons}-Aln_{pc} achieved a similar performance.

When we compare ILP-SVM models with ILP-DT models, all ILP-SVM models outperformed ILP-DTs. In fact, SVMs are often more accurate than DTs. We observed that ILP-DTs have produced fewer and simpler rules for both databases and they presented a poorer classification on test examples. In order to provide a comparison with [46, 47], we considered for comparison the ILP-DT-Seq model, since it uses all properties handled in [46, 47], except those predicted from sequences, such as secondary structure. Our results show that all models proposed here outperformed ILP-DT-Seq for both databases.

When we combine the representations the number of features can increase creating sparse data. However, this can be avoided by using a feature selection technique. Here, we applied chi-square statistical test to remove class uncorrelated rules. We set δ for 0.05 and 0.25, see Methods. Table 4.5 shows how AUC values vary according to δ and without the chi-square test. We can observe that for most methods the performance was kept with δ=0.05 with a significant reduction in the number of features. On the other hand, δ=0.25 worsened the performance for all methods. In fact, for some families δ=0.25 removed all logical rules.

We compared our best models, that is, those trained from *Seq*-Aln_{cons}-Aln_{pc} features, with two models based on SVM (SVM-Ngram-LSA [38] and SVM-LA [32]), and also with two other widely used methods: HMMer-3.0 [78] and PSI-BLAST [12]. We carried out rank-sum tests [96] to compare models listed in Figure 4.6, and we show in Table

Table 4.5: **Average AUC and number of logical rules according to chi-square test for S_{full} and S_{30} databases.**

Methods	**AUC (#logical rules) for S_{full}**			**AUC (#logical rules) for S_{30}**		
	no chi-square	δ =0.05	δ =0.25	no chi-square	δ =0.5	δ =0.25
ILP-SVM-Seq	0.79 (228.59)	0.79 (89.15)	0.75 (59.30)	0.77 (311.09)	0.77 (91.04)	0.70 (26.4)
ILP-SVM-Aln_{cons}	0.81 (44.91)	0.81 (34.98)	0.77 (12.76)	0.81 (56.72)	0.80 (36.44)	0.72 (13.16)
ILP-SVM-Aln_{ps}	0.80 (191.65)	0.79 (139.61)	0.75 (66.15)	0.81 (241.96)	0.81 (178.72)	0.73 (71)
ILP-SVM-Seq-Aln_{cons}	0.85 (311.09)	0.83 (144.07)	0.79 (35.8)	0.80 (381.12)	0.80 (178.56)	0.74 (49.96)
ILP-SVM-Aln_{cons}-Aln_{ps}	0.82 (236.56)	0.82 (174.59)	0.79 (46.04)	0.82 (283.12)	0.82 (209.76)	0.80 (57.28)
ILP-SVM-Seq-Aln_{cons}-Aln_{ps}	0.87 (502.74)	0.85 (283.69)	0.81 (74.3)	0.82 (623.56)	0.82 (357.28)	0.79 (90.96)
ILP-DT-Seq	0.67 (228.59)	0.67 (89.15)	0.62 (59.30)	0.65 (311.09)	0.65 (91.04)	0.61 (26.4)
ILP-DT-Aln_{cons}	0.70 (44.91)	0.70 (34.98)	0.72 (12.76)	0.69 (56.72)	0.69 (36.44)	0.65 (13.16)
ILP-DT-Aln_{ps}	0.68 (191.65)	0.68 (139.61)	0.64 (66.15)	0.67 (241.96)	0.67 (178.72)	0.62 (71)
ILP-DT-Seq-Aln_{cons}	0.72 (311.09)	0.71 (144.07)	0.67 (35.8)	0.69 (381.12)	0.68 (178.56)	0.63 (49.96)
ILP-DT-Aln_{cons}-Aln_{ps}	0.71 (236.56)	0.71 (174.59)	0.73 (46.04)	0.71 (283.12)	0.70 (209.76)	0.62 (57.28)
ILP-DT-Seq-Aln_{cons}-Aln_{ps}	0.74 (502.74)	0.74 (283.69)	0.69 (74.3)	0.71 (623.56)	0.71 (357.28)	0.63 (90.96)

Table 4.6: **Rank-sum test p-values for curves of Figure 4.6.**

	$\mathbf{S_{full}}$			
Methods	**SVM-LA**	**SVM-Ngram-LSA**	**PSI-BLAST**	**HMMer-3.0**
ILP-SVM-Seq-Aln$_{\mathbf{cons}}$-Aln$_{\mathbf{ps}}$	0.93	4.93e−05	2.55e−07	5.63e−07
ILP-DT-Seq-Aln$_{\mathbf{cons}}$-Aln$_{\mathbf{ps}}$	7.27e−06	2.23e−04	0.17	5.44e−07
	$\mathbf{S_{30}}$			
ILP-SVM-Seq-Aln$_{\mathbf{cons}}$-Aln$_{\mathbf{ps}}$	0.07	0.05	3.85e−05	1.33e−06
ILP-DT-Seq-Aln$_{\mathbf{cons}}$-Aln$_{\mathbf{ps}}$	1.44e−05	0.015	0.41	1.2e−05

4.6 statistical measures for this comparison, we consider a result with p $\leq$ 0.05 to be significant.. For both databases, ILP-SVM-*Seq-Aln*$_{cons}$-*Aln*$_{pc}$ outperformed SVM-Ngram-LSA, PSI-BLAST and HMMer-3.0, and achieved comparable performance to SVM-LA. ILP-DT-*Seq-Aln*$_{cons}$-*Aln*$_{pc}$ model achieved better results than HMMer-3.0, and achieved similar performance to PSI-BLAST. Based on ILP-SVM-*Seq-Aln*$_{cons}$-*Aln*$_{pc}$ performance, we can conclude that the combination of alignment information and sequence properties, and the strategy of selecting only the most important features can yield a more accurate model than those that explore all alignment positions, as HMMer-3.0 and PSI-BLAST, and those that extract Ngram from unaligned sequences, such as SVM-Ngram-LSA.

Although the results show that ILP-SVM-*Seq-Aln*$_{cons}$-*Aln*$_{pc}$ outperformed some state-of-art methods, we emphasize that the performance of PSI-BLAST depends on the number of iterations and on the size of the database used to build the profiles. Thus, to extract the maximum performance of PSI-BLAST, we adopt the semi-supervised training strategy and we used nrdb90 as unlabeled database and set the number of iterations to 20, as done in [42]. Unsurprisingly, it performed better than our ILP-SVM models. For example, PSI-BLAST$_{nr20}$ achieved a AUC of 0.88 for the database S_{full} and 0.83 for the database S_{30}. Certainly, methods trained from the nrdb90 database are expected to build more accurate models and be more effective in annotating remote homologous proteins. However, the computation time of methods that adopt semi-supervised training depends on size of the unlabeled database. Therefore, PSI-BLAST run on this configuration is computational time consuming. In conclusion, when supervised training strategy is employed ILP-SVM methods obtain better or comparable performance than its competitors.

In order to provide an example of the biological interpretation of the logical rules constructed by WARMR, we show in Table 4.7 some rules which have been learned on members of the "Glucocorticoid receptor-like (DNA-binding domain)" superfamily. Note that, we show in Table 4.7 (top) original rules learned by WARMR, and 4.7 (bottom) their interpretation. Rules R_1, R_2 and R_3 were learned from conserved amino acid positions (*Aln*$_{cons}$) and R_6 from conserved physico-chemical positions (*Aln*$_{pc}$), see Figure 4.1, while rules R_4 and R_5 were learned from sequential properties. These rules represent only the conserved properties of "Glucocorticoid receptor-like (DNA-binding domain)" superfamily members, that is, these rules catched essential features identifying the superfamily mem-

Table 4.7: **Some logical rules learned by WARMR on "Glucocorticoid receptor-like (DNA-binding domain)" sequences (see Figure 4.1).**

WARMR output
R_1 : Homologous(A):- col(A,c,24), col(A,c,27), col(A,c,51) **1.0** R_2 : Homologous(A):- col(A,c,45) **0.7** R_3 : Homologous(A):- col(A,k,29) **0.45** R_4 : Homologous(A):- hydrophobic(A,2) **0.7** R_5 : Homologous(A):- aminoacidPairRatio(A,cg,1) **0.77** R_6 : Homologous(A):- col(A,B,34), small(B) **1.0**
Interpretation
R_1 : 100% of homologous proteins have the C amino acid in positions 24, 27 and 51. R_2 : 70% of homologous proteins have the C amino acid in position 45. R_3 : 45% of homologous proteins have the K amino acid in position 29. R_4 : 70% of homologous proteins have between 10 and 20% of hydrophobic amino acids. R_5 : 77% of homologous proteins have at least 1 pair of CG. R_6 : 100% of homologous proteins have a small amino acid in positions 34.

bers. This was possible by using first-order logic predicates to represent the properties of each superfamily member, and by applying ILP in order to filter the essential features.

4.4 Conclusion

We have combined ILP and propositional models for improving the accuracy of remote homology detection methods. Our approach can be segmented into three parts. First, training sequences are represented through first-order logic predicates. Similar to [46, 47, 88], we have used a representation based on sequence properties. Additionally, we introduced a novel representation based on conserved amino acid positions in protein alignments. Also, we related the logical representation based on sequential properties with our logical representation based on conserved positions creating a new representation for conserved physico-chemical positions in a MSA. Second, we executed WARMR, an ILP system, in order to find only the most frequent patterns in our training set. Third, the logical rules learned in the previous stage were converted in binary attributes for training propositional models. Here, we applied decision trees and the widely used support vector

machine as propositional methods.

Our methodology is partly similar to the study developed in [46, 47]. However, we proposed a novel logical sequence representation based on conserved positions in MSA; we combined this representation with the logical representation based on sequence properties only, proposed in [46, 47, 88]; we applied SVMs rather than DTs. We showed that the prediction performance of our method, that uses logical representation of alignment information, is better than the prediction performance of our models trained only on the sequential representation. Also, the combined representations improved the performance of ILP-DT models in any sequence identity range and the performance of ILP-SVM for the original database. We carried out comparisons among the models proposed here with models based on SVM (SVM-Ngram-LSA and SVM-LA), a model closer to the model proposed in [46, 47], that is, ILP-SVM-Seq, HMMer-3.0 and PSI-BLAST. Our experiments showed that for the same data set, ILP-SVM models achieves a superior or a comparable performance for any sequence identity range. In particular, our method produces a human-understandable output that can provide insights about conserved features of protein families.

We can conclude that the first-order logic language is suitable to represent conserved protein properties, and that from this representation, an ILP system can learn the essential rules that discriminate between homologous and non-homologous proteins. Our methodology supports the intuition that proteins with remote evolutionary relationship have suffered several mutational events, and that only essential amino acids and their physico-chemical properties are kept in evolved sequences. Thus, computational methods that explore only the conserved positions are more suitable to the remote homology detection problem than the methods that explore all amino acids within sequences.

We have confirmed through this study that conserved alignment positions play an important role in recognizing remote homologous proteins contrary to sequential properties extracted from unaligned sequences. We highlight that sequential properties can be useful for helping to identify remote homologous proteins, however, when the sequence identity is smaller than 30%, this information might become noise and worsens the performance

of methods, as we observed for ILP-SVM-Seq-Aln_{cons}.

Another advantage of our methodology is the simplicity to include additional sequence properties. For this we can create a new predicate that represents the property and no modification of the algorithm is necessary. In this study, we used only properties that can be extracted directly from sequences or from conserved alignment positions. We considered a limited number of amino-acid physico-chemical properties (only 16), since our logic sequential representation is based on previous ones [46, 47]. However, the Amino Acid Index Database [**?**] has defined amino acid numerical indices for more than 500 different kinds of physico-chemical properties. Some methods used these indices to train SVM and achieved a good performance [42]. Thus, we intend to create a logic sequential representation that takes into account properties of the Acid Index Database. Other points that we would like to explore are: the presence of short hydrophobic blocks in homologous proteins, as well as, structurally conserved amino acids [19], and functional amino acids, that is, active and binding sites. Moreover, we would like to replace WARMR with MineSeqLog [97]. MineSeqLog is an extension of WARMR that works on sequences where each sequence is an ordered list of ground predicates. This approach seems to be more suitable to deal with protein sequences, since the amino acid order is taken into account. PSI-BLAST performs better when run on nrdb90 with 20 iterations, and the use of PSI-BLAST output, as done in [40], to train our models provides another path to be explored.

Chapter 5

CASH - Combination of Annotations by Species and pHMMs

In this chapter we present CASH, a large-scale pipeline for remote homologous protein annotation. CASH combines several computational approaches and explores in a more sophisticated way protein properties in order to provide a system for annotation of divergent protein sequences. In section 5.1, we discuss the problem of protein annotation in highly divergent genomes, like *P. falciparum*, and we present our motivation for the developing of the method. In section 5.2, we detail the framework is detailed and in section 5.3 we present and discuss CASH' results. The conclusion of this work is presented in section 5.4.

5.1 Background

Malaria is one of the most debilitating pathogenic infections responsible for the death of around 800.000 people per year, primarily children and pregnant women in sub-Saharan Africa (WHO 2010). Fatal malaria is almost exclusively caused by *P. falciparum*, an organism eukaryote unicellular. The completion of its genome has opened up a multitude of avenues for providing enhanced knowledge in the complex mechanisms that contribute to the development and dissemination of this pathogen along with exploration and identification of novel drugs and vaccine targets. However, around 48% of the open reading frames predicted in this genome, remarkably rich in A and T (on average 85%), remain without any putative annotation [98, 99, 100, 101, 102]: PlasmoDB [98] (version 8.1) identifies

2638 proteins with unknown or hypothetical function over 5491 genes.

This poor annotation is due to weak similarity of *P. falciparum* proteins with the sequences of known eukaryotic. This can also be consequence of the inefficiency of large-scale annotation tools that still are based on simple and fast similarity search method [9], see section (3.1). More effective tools (generatives methods, see section 3.2) have been proposed to decrease the number of unknown proteins. These methods create protein family signatures, for all known family, and try to recognize these patterns into proteins with unknown function. The most used generative methods construct a profile Hidden Markov Models (pHMM) to represent a consensus of signals that characterize a given protein family [50, 103, 65, 63, 13, 104, 105]. However, these tools still largely fail to detect distant homologous proteins. An alternative is to use discriminative methods, see section 3.3. Although, these new methods have achieved good performance on benchmark databases, they have a significant computational cost to be applied in large-scale annotation task. Thus, researchers have tried to improve the performance of generative methods by adding to them extra information, such as phylogenetic [17] and protein structure information [18, 19]. The coexistence of domains within a protein has revealed to be another very powerful tool to annotate divergent protein sequences [106], especially for the *P. falciparum* genome [24, 107, 25]. However, the number of proteins with unknown functions in *P. falciparum* remains high (around 43%) even after domain co-occurrence analysis. The main reason for this is that relevant signals in homologous sequences might become too weak to be identified by generative methods that build a single probabilistic model by representing the protein family consensus. In other words, if sequences within the protein family are very divergents or if the pool of homologous is biased (too small or overrepresented by sequences of certain species) this gereral profile can fail in detecting remote homologous members.

The use of alternative profiles created from ortholog sequences has helped to identify new general transcription factors in *P. falciparum* [51]. Motivated by the positive results of Callebaut et al's, we proposed CASH [49], a pipeline that construct additional profiles to represent a single Pfam domain [50]. Contrary to Callebaut et al's [51] that explored a very small set of orthologous sequences, we increased the number of sequences what allow us to explore different evolutionary pathways within the life phylogenetic tree, and possibly

those of species that are phylogenetically distant from *P. falciparum*. In our approach, these additional profiles are combined with existing pHMMs to search for homologous sequences in *P. falciparum* genome. The outcomes of those profiles are processed and transformed into features used to train a meta-classifier (Support Vector Machine (SVM)) that assigns a confidence score to the domain predictions. Based on this score and on other properties, as domain co-occurrences, CASH finds the most probable architecture (domain arrangements) for each *P. falciparum* protein sequence by resolving a multi-objective optimization problem. CASH have been applied to the *P. falciparum* genome, but the framework is highly generic and can be applied to any other genome.

5.2 Methods

Here, we present our approach in detail. First, we describe databases used in this work (section 5.2.1). Second, we explain our criteria for selection of representative species or orthologous sequences (section 5.2.2) used to build additional profiles, here called phylogenetic models. Third, we explain how Pfam methodology works (section 5.2.3) and how we modify it including phylogenetic models (section 5.2.4). Fourth, we describe how to combine those models to produce a more reliable prediction (section 5.2.5). Fifth, we present a novel algorithm that finds the most likely domain arrangement, for a given protein sequence to be annotated, taking into account domain co-occurrences (section 5.2.6). Sixth, we describe how our predictions were obtained (section 5.2.7), how we carried out comparisons with early results (section 5.2.8), and how our results can be visualized and interpreted. Finally, we discuss parameter settings and tools used in this work (section 5.2.10)

5.2.1 Databases

Our method extends Pfam, an important collection of protein domains, that has been widely used for annotating proteins with unknown function. We used version 24 downloaded from (`http://pfam.sanger.ac.uk`) that contains 11912 protein domains. In order to assess the performance of our method, we applied it to the *P. falciparum* genome. For this, we have used PlasmoDB (`http://PlasmoDB.org`), that is the official repository of the *P. falciparum* genome, used as a reference database by malaria researchers. Version

8.1 contains 5491 proteins. Note that by “unknown functions" we shall refer to several key words used in PlasmoDB: unknown function, product unspecified, hypothetical protein, pseudogene and conserved *Plasmodium falciparum* protein family.

5.2.2 Selection of representative species from the eukaryotic tree of life

Phylogenetic models were built from 46 species selected from the eukaryotic phylogenetic tree: (i) 10 species are close to the *P. falciparum*, and they belong to the *alveolata* clade, and (ii) 36 species are spread out the entire eukaryotic tree and represent distant clades. See complete list in Table A.1 annexe A.

5.2.3 Pfam methodology

Let D^i be an arbitrary protein domain in the Pfam database. D^i identifies a set of protein sequences s^i_j that share evolutionary and structural properties, $S^i = \{s^i_1, \ldots, s^i_n\}$. Each s^i_j is associated to a protein in the Uniprot database [83], that provides a taxonomy for it. Pfam defines a subset S^{i*} that contains only *seed sequences*, that is, only representative members of S^i. It aligns sequences in S^{i*} to build a profile hidden Markov model, pHMM^{i*}, that represents the consensus of the seed alignment, and it describes the common features of the sequences in S^{i*}. Pfam provides a library of pHMMs, one for each domain. This library can be used to scan databases of proteins with unknown function localizing regions of the sequence that belong to known Pfam domains. Figure 5.1 (only solid lines) shows the Pfam flowchart.

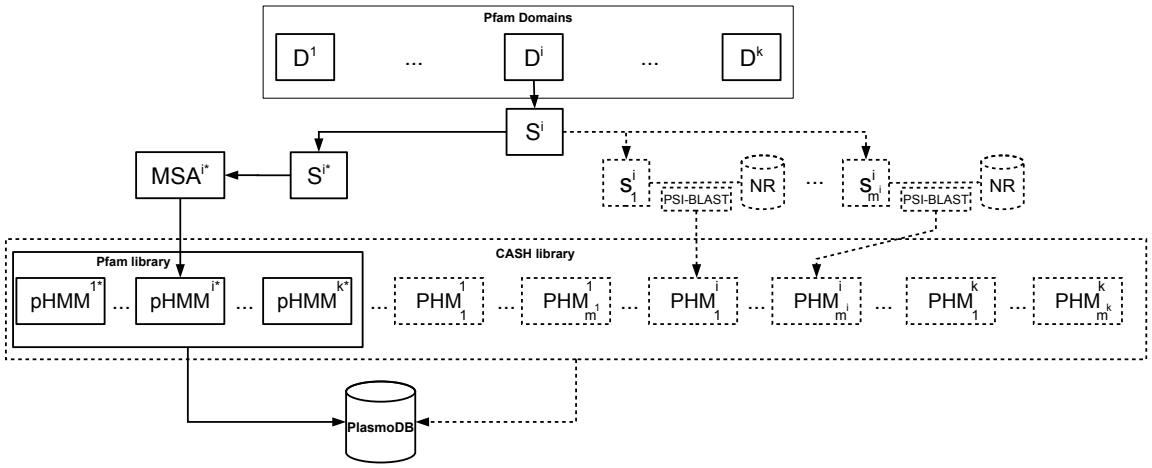


Figure 5.1: **Pfam and CASH flowchart.** Pfam methodology is showed in solid lines, while modifications proposed by CASH are shown in dotted lines.

5.2.4 Phylogenetic Models

The pHMM^{i*} is a probabilistic model explaining how sequences in the seed set S^{i*} have evolved. In contrast, we shall build a number of new models by exploiting information coming from evolutionary paths associated to specific species. We shall take all sequences from our 46 selected species (see previous section). If a selected species is not represented within S^i, we shall use a close species following the map established in Table A.1 annexe A. If this is not possible, we randomly choose a new species in S^i. Each selected sequence will be used as a query to search, with PsiBlast, for similar sequences within the non-redundant protein database (NR). As a result, several probabilistic models are created, here named, PHM$^i_1, \ldots,$ PHM$^i_{m_i}$, with $m_i \leq 46$, and PHM is the abbreviation for "Phylogenetic Models", see Figure 5.1 (dotted lines).

5.2.5 Combining Models Predictions

We modify Pfam original library, in such a way that, a single domain D^i is now represented by an *ensemble model* $\mathcal{E}$ ={PHM$^i_1, \ldots,$ PHM$^i_{m_i}$, pHMM^{i*}} [108]. Ensembles are usually build by applying: (i) a single learning algorithm to subsets of the training data, like Bagging [109] and Boosting [110], or (ii) different learning algorithms to a single dataset, like Stacking [111, 112]. We combined both approaches to construct $\mathcal{E}$. For this, we trained two algorithms (PSI-BLAST and HMMer) from different datasets (individual sequences used as query to train PSI-BLAST, and seed alignments to train HMMer), because this hybrid approach produces more heterogeneous models.

After the ensemble training phase, we need to combine its output models to produce a final decision. For this, one can employ plurality voting [113] or meta-learning techniques [52, 53]. We implemented a meta-learning decision strategy that uses a second classifier (SVM) trained from features that were obtained by pre-processing outputs of base models, see schema of Figure 5.2. In general, base model outputs (frequently confidence scores) are used directly in the training of the second classifier, also called meta-classifier [111]. We have applied this general approach, but it did not achieve good performance, possibly for two reasons. First, the high divergence of the *Plasmodium Falciparum* genome implies that the distribution of confidence scores in the training set is different from the distribution in the testing data. Second, our models are very heterogeneous, since they were trained from

different source data, thus, it is not expected that there is necessarily an agreement among their predictions. Because of these two observations, any decision strategy that tries to find a consensus into base model answers is expected to fail. To avoid it, we designed meta-features which aim: (i) to highlight individual model results, when a consensus among base models is not observed, and (ii) to provide an indication of the performance of all models.

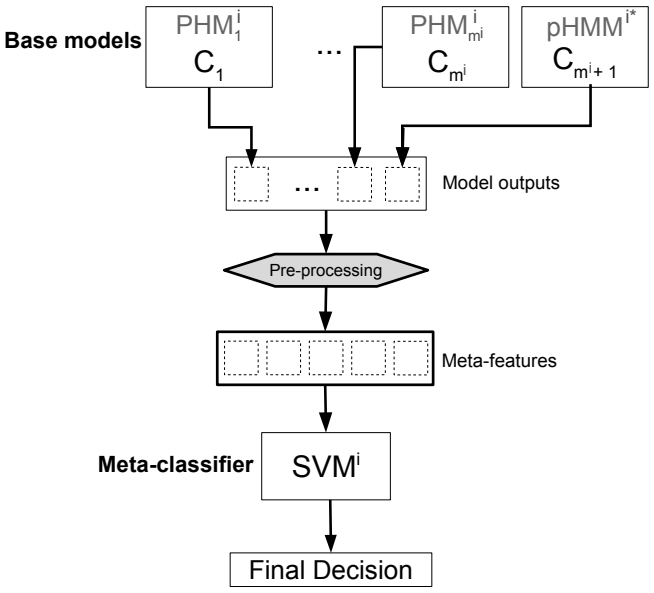


Figure 5.2: **Meta-classifier for model prediction combination.**

We defined five meta-features as follows. Let $\mathcal{C} = \{C_1, \ldots, C_{m_i}, C_{m_i+1}\}$ be the set of base models for an arbitrary domain D^i, where $C_1, \ldots, C_{m_i}$ are phylogenetic models, and C_{m_i+1} is the pHMM^{i*}. We let s_t be a query-sequence that we wish to score against all models in $\mathcal{C}$, and $\mathcal{C}^*$ a subset of $\mathcal{C}$, where each $C_j \in \mathcal{C}^*$ is a base model that best matched an exclusive segment of s_t, that is, no two members of $\mathcal{C}^*$ matched overlapped segments. Thus, for a single s_t we create x meta-examples, where x is the size of $\mathcal{C}^*$. This is done because several copies of the domain D^i can be found in s_t and we want to represent each of them.

To achieve goal (i), we extract from the C_j output three features: the E-value, the hit length, and a binary feature that indicates if the E-value of C_j is smaller than a threshold T'. For goal (ii), we define two features concerning the percentage of models in $\mathcal{C}$ that *supports* the prediction of C_j. For this, we say that a model C_i supports the prediction of a model C_j if their matches on s_t overlap with each other, and the overlapping size is

greater than 50%. Thus, the fourth meta-feature is defined as the percentage of models that supports C_j having E-values smaller than a threshold T''. The fifth feature represents the percentage of models that support C_j and that are built from species that belong to the clade of s_t. Our motivation is based on the assumption that species of the same clade tend to share more domains than species of different ones. Note that, we do not penalize predictions that are obtained from models built from species of distant clades, we just wish to use this evolutionary information to reinforce the presence of a domain when it is observed in species close to s_t. We discuss T' and T'' thresholds in the section "Parameter settings and tools used".

From these five meta-features, we trained a meta-classifier (SVM) to distinguish between real domains and false predictions. SVM is a linear classifier that discriminates two classes by finding a large-margin separation among their training samples. We built an one-vs-rest SVM [114], to conduct binary classifications for each domain D^i, independently, as seen in Figure 5.2. For this, we used as positive training set all sequences in the S^i, except those used in the training of base-models, and we randomly selected negative training sequences that are not in S^i. For both sets, we selected only one sequence per species to increase the training set diversity. Positive and negative datasets have the same size, in order to avoid unbalanced sets.

After the training phase, each query sequence s_t in the *P. falciparum* genome is scored by each one of the base models, and the five features are extracted from their outputs, as indicated in Figure 5.2. Then, the SVM^i (trained to recognize the domain D^i) is asked to determine if the domain D^i is found in the protein sequence s_t, and to provide a confidence score. However, from a biologist's perspective, it is more valuable to identify the most likely domains that occur in s_t and that are not overlapping. This is known as a multiclass classification problem. To enable a set of one-vs-rest SVMs (one for each domain) to work with this problem, it is essential to calibrate the output of each classifier into a confidence measure, like the posterior probability. Since standard SVMs do not provide such probabilities, we employed Platt's method [115] to map SVM outputs into posterior probabilities. As a result, SVM's probabilities are comparable and one can assign to s_t the domain that achieves the highest predictive value, as done in [116]. However, the tendency of the domains to occur preferentially with a small set of other domains in a

protein sequence can favor lower confidence domains. Thus, we present in the next section a novel algorithm to determine the most probable domain arragenment for s_t that takes into account domain combinations.

5.2.6 Resolving protein domain architectures

Recently, domain co-occurrence information has been used to improve the performance of domain recognition methods [20, 21, 22, 23, 24, 25]. Here, we consider two of these methods that were applied to *P. falciparum* annotation: CODD (Co-Occurrence Domain Discovery) [24] and dPUC (domain Prediction using context) [25], see section 3.2.3 for a revision of these methods.

dPUC presents two advantages over CODD: it takes into account co-occurrence of repeated domains, and penalizes higher confidence domains without co-occurrence. However, we believe that there are two points into dPUC's approach that could be improved. First, it did not consider multi-domain co-occurrence to compute protein architectures (its importance is illustrated in Figure 5.3, where two architecture are proposed for a hypothetical protein p. Suppose that both individual domain scores and pairwise domain co-occurrence probabilities are slightly better in the first architecture. However, suppose that the three-domain combination *abc* has never been observed before, while *def* is known to be frequent. Naturally, methods based on only pairwise domain combination will select the first architecture as the most probable, likely making a wrong decision). Second, dPUC has combined individual domain scores and co-occurrence information into a very simplified function, that is then optimized. However, we argue that this combination is non trivial, and that the function could be more complex: domain and co-occurrence scores could be weighted, and extra information, like multi-domain combination scores, could be included. To address these two points, we propose a novel algorithm treating the protein domain architecture problem as a multi-objective optimization problem.

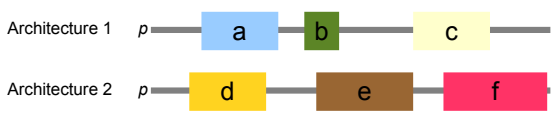


Figure 5.3: **Importance of using multi-domain combinations.**

First, our algorithm enumerates all possible architectures, subjected to the domain

co-occurrence constraints, and then it elects the architecture that maximizes a set of objective functions. To do so, let s_t be a query sequence, $\mathcal{P}_t$ be its set of potential Pfam domains (Figure 5.4-A), $\mathcal{L}$ be a list having all domain architectures found in Uniprot [83] (Figure 5.4-C), G be an interval graph where nodes represent domains in $\mathcal{P}_t$ and edges connect overlapping domains (Figure 5.4-B). Note that, domain overlap is allowed if it is less than 30 amino acids, and it comprises at most 50% of the match, as done in [106], or if it was already observed in Uniprot proteins, as done in [25]. We also let $MIS(n, G)$ be the *Maximal Independent Set*(MIS) of the graph G containing the node n. We recall that an independent set is a set of nodes in G, such that no edge connects two vertices in the set, and that a maximal independent set is a set that is not a proper subset of any independent set. Our algorithm enumerates all maximal independent sets taking each domain $d_i \in \mathcal{P}_t$ that satisfies the following constraint: a node d_j is in $MIS(d_i, G)$ iff d_i and d_j co-occur, that is, if both are present in the some architecture in $\mathcal{L}$ (Figure 5.4-D). The set of all feasible solutions is ordered by putting on the top the architecture containing domains with highest scores. We call this set $\mathcal{L}'$, and from it we wish to find the optimal architecture. For this, we associate a set of functions to each candidate solution, and we treat this problem as a multi-objective optimization problem. There exist many methods to find a solution for this problem [117] and we used a variation of the lexicographic approach proposed in [118], where objective functions are arranged in order of importance, constraints are formulated on these functions, and the following optimization problems are solved one at a time:

$$\begin{aligned} &\underset{x \in X}{Maximize}\ F_i(x) \\ &subject\ to\ F_j(x) \leq F_j(x_j^*) + \delta_j \quad j = 1, \ldots, i-1, i > 1, \\ &i = 1, 2, \ldots, I \end{aligned} \tag{5.1}$$

where $i = 1, 2, \ldots, I$ indexes the preferred order of the functions, $F_j(x) \leq F_j(x_j^*) + \delta_j$ is a constraint on the jth function, $F_j(x_j^*)$ represents the optimum for F_j, and δ_j is a positive constant that defines a value tolerance for each objective function (values for it are discussed in the next section). For each function F_i, we find the maximum value x_i^*, such that $F_j(x_i^*) \leq F_j(x_j^*) + \delta_j$, for all $j < i$. Note that, if we set $\delta_j = 0$, the final solution is dictated by the initial objective-function ranking process. On the other hand, if we set $\delta_j > 0$, we expand the decision region and allow other functions be optimized.

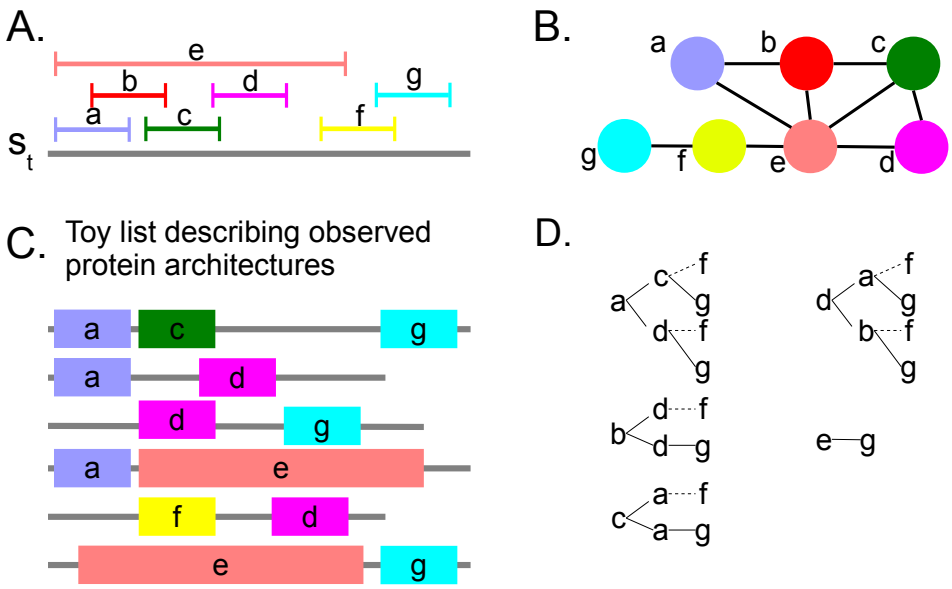


Figure 5.4: **Illustration of CASH computing framework**

The lexicographic approach makes sense only iff x_i^* can be computed for all F_i's, and we ensure this by applying our objective-functions to the list of feasible architectures $\mathcal{L}'$. We designed four functions according to several objectives: a. To ensure that higher confidence domains are in the final architecture, we define

$$F_1(x) = arg\,max\{SVM_{prob}(x_i)\} \tag{5.2}$$

where x_i is a domain contained in some feasible architecture $x \in \mathcal{L}'$, and $SVM_{prob}(x_i)$ is the SVM probability for x_i. b. To maximize the number of Multi-Domain Co-Occurrences (MDCO) , we define

$$F_2(x) = MDCO(x), \tag{5.3}$$

where $MDCO(x)$ is the multi-domain co-occurrence factor, that is the number of domains in x co-occuring in $\mathcal{L}$. For instance, observe Figure 5.4-D, $F_2(acg) = 3$, because acg is found in $\mathcal{L}$ (Figure 5.4-C). On the other hand, $F_2(adg) = 2$, because only ad and dg are found in $\mathcal{L}$. c. Pairwise combinations are useful to decide between two architectures presenting the same MDCO, and we define

$$F_3(x) = pairDCO(x), \tag{5.4}$$

where $pairDCO(x)$ counts the number of domain pairs of x that co-occurring in $\mathcal{L}$. d. To select the architecture with highest score domains, we define

$$F_4(x) = \frac{1}{N}\sum_{1}^{N} SVM_{prob}(x_i) \tag{5.5}$$

where N is the number of domains in x.

5.2.7 Prediction analysis

In order to filter out false predictions, firstly, we cut off all predictions with domain coverage smaller than 40%. Secondly, we consider a threshold for SVM probabilities, so that, a prediction is validated if its $SVM_{prob} > 0.5$. Moreover, we set a permissive threshold ($SVM_{prob} > 0.05$) to validate co-occurrence domain predictions.

We assessed the physico-chemical conservation of our predictions (Table 5.6) by comparing each sequence predicted to the distribution of amino acids of the multiple sequence alignment used to build the profiles. For this, we first score each query sequence s_t by using CASH library, and then we analyze the hit-match produced by pHMM and by the phylogenetic profile that better match s_t. From these hit-matches, we compute true positive, true negative, false positive and false negative rates for each aligned amino acid in the s_t. Then, we calculate performance measures like: accuracy (Acc), sensitivity (Sen), specificity (Spe) and positive predictive value (PPV).

5.2.8 Comparison with earlier results

We carried out direct comparisons with standard Pfam predictions. With CODD and dPUC approaches direct comparisons was not possible because of unavailability of the CODD's source code, and the incompatibility of dPUC's code with the version 24 of Pfam. To provide some comparisons, we compute the percentage of agreement between their predictions and CASH ones, and the percentage of improvement of CASH on the Standard Pfam, that was then compared to performance measures provided by CODD and dPUC.

5.2.9 Visualizing our results

All results are available in a web site at `http://www.lgm.upmc.fr/PFAM_annotation/`. The site has a user-friendly interface that allow user search for Pfam domains by using Pfam accession number or key words, and for *P. falciparum* proteins by using PlasmoDB accession number or annotation keywords. The results are shown as a list of proteins that

match with search criteria. Previous annotations provided by PlasmoDB and previous domain predictions suggested by Pfam, CODD, dPuc and CASH are shown for each query protein. Also, we show graphically protein domain organizations proposed by CASH and its competitors. If CASH architecture weas already observed in the annotated proteins, we compare them by highlighting their similarities.

5.2.10 Parameter settings and tools used

We built an ensemble model for each Pfam domain by considering Pfam pHMMs, and by building additional phylogenetic models. Profiles HMMs were downloaded directly from the Pfam web site, while phylogenetic profiles were built by taking specific species from the eukaryotic tree of life as queries to train PsiBlast (version 2.2.23 for 5 iterations) on the NR database (downloaded in February 2011). In order to detect more potential Pfam domains, we set a permissive search E-value (100) in both tools, and we turned off the bias filter in the HMM tool. We combined the predictions of those profiles by training a SVM from features coming from profile outputs. For SVM, we used the LIBSVM tool [119] (version .3.0) with default parameters, and we turned on the option "-b" to provide probability estimates. The software is available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

We designed five features to highlight the best prediction, and to provide a measure of the performance of all profiles. For a given query sequence s_t, we set the E-value, the hit length and we defined a binary feature by setting a restrictive E-value threshold ($T' = min(1^{-30}, Ev_{D^i})$, where Ev_{D^i} is the greatest E-value observed among proteins in domain D^i). From all predictions, we extracted features that concern the percentage of models that agree with the best prediction. For the fourth feature, we computed the number of profiles with E-value smaller than $T'' = 100$. For the fifth feature, we just counted the number of phylogenetic models that were built from species sharing the same clade with s_t. If we do not observe predictions coming from close species, we ignore the feature with no penalization.

In order to find the best domain architecture for a given query sequence, we introduced a novel algorithm that generates a set of feasible architectures based on domain co-occurrence constraints, and finds the most likely by optimizing the four objective-functions above. We experimentally set the tolerance values δ_j for each function F_j ($j \leq 3$) Eq. 5.1.

To favor higher confidence domains we set $\delta_1 = 0.1$ if $F_1(x_1^*) > 0.5$, otherwise we consider domain co-occurrence by setting $\delta_1 = 0.25$. Since, we wish to maximize the number of domain co-occurrences, we set $\delta_2 = \delta_3 = 0$.

The phylogenetic tree showed in Fig 5.5A was obtained by combining sub trees (constructed by joining sequences of two distinct clades) into a super tree [120]. We used Neighbor Joining [121](contained in the Phylip package version 3.67) to build the su-trees. The super tree was built by using the program *supertree* with the algorithm proposed in [122].

5.3 Results and Discussion

A large number (2638) of *P. falciparum* proteins has no functional domain annotation in PlasmoDB and they remain with no putative annotation even after the analysis of known predictive methods such as (2050) [24, 25]. By comparing our predictions to those obtained with Pfam version 24 (Pfam_{24}), we observe that for the same range of E-values we perform better, since the number of proteins remaining with no predictions is significantly smaller after the CASH application. There are 2312 proteins (Table 5.1, for E-values ≤ 1, where bold values highlight best performance) in Pfam_{24} with no identified putative domains, and with CASH this number has been drastically reduced to 1664, providing the 28.03% improvement over Pfam. These values describe the impact of CASH on the full genome, but indeed CASH realizes a domain analysis. Its predictions contribute new information to single domain proteins as well as multi-domain proteins having a yet unraveled domain architecture. Hence, besides finding domains for never annotated proteins, CASH attempts to complete domain architectures whenever needed.

Table 5.1: **Number of proteins with no domain annotation in Pfam and CASH.**

E-value	**Pfam_{24}**	**CASH**	**Improvement(%)**
1e-60	4755	**4426**	6.92
1e-30	3938	**3399**	13.69
1e-15	2962	**2436**	17.76
1e-05	2318	**1795**	22.56
1	2312	**1664**	28.03

To predict domains and domains architectures in *P. falciparum* sequences we construct several models by using two computational approaches that underlie different evolutionary assumptions. The known profile Hidden Markov Model (pHMM), exploited also by other annotation approaches like Pfam [50], CODD [24] and dPUC [25], is constructed to provide a general profile capturing the consensus of homologous sequences. The idea behind this model is that homologous proteins should share common physico-chemical and structural features that could be described by a sequence profile based on the entire set of homologs [11]. Here, we introduce another class of models, called *phylogenetic models*, built by taking individual homologous sequences as reference sequences, and by constructing different profiles for each one of these sequences. The idea behind the construction of these models is that protein evolution pathways are limited due to the numerous structural and functional constraints that a protein undergoes. This means that the evolutionary constraints that drive a protein evolution in a specific species and the corresponding signals identifiable in a sequence, might be more easily detectable by looking closely at the way some other species found its own evolutionary solution. The hope in doing this is that single species, possibly very distant from *P. falciparum*, will share their evolutionary solutions with *P. falciparum*. The construction of phylogenetic models constitutes a basic difference between our approach and those based on pHMM.

The reference sequences generating phylogenetic models were chosen to be representative of the whole tree of eukaryotic life, see section 5.2.2. We defined a reference tree of species (Figure 5.5-A) that represents well the Alveolata clade but that spans also across very distant eukaryotic clades, see list of species in (Figure 5.5-B). The tree was built from Pfam ribosomal protein families [123] (109 Pfam families coming from small and large ribosomal subunits, see list of families in B.1 annexe B). Sequences of these families were divided in 7 groups according to their clades (Alveolata, Amoebozoa, Cryptophyta, Diplomonadida, Fungi, Kinetoplastida, Metazoa, and Viridiplantae). We considered all 21 pairs of clades, and for each pair, we concatenated as many ribosomal families as possible to build a subtree. We removed from concatenated sequences those subsequences showing no similarity, such as N- and C-terminals, if any. The tree was obtained by building a supertree [122] from the subtrees.

For each Pfam domain and the set of homologous sequences associated to it, we

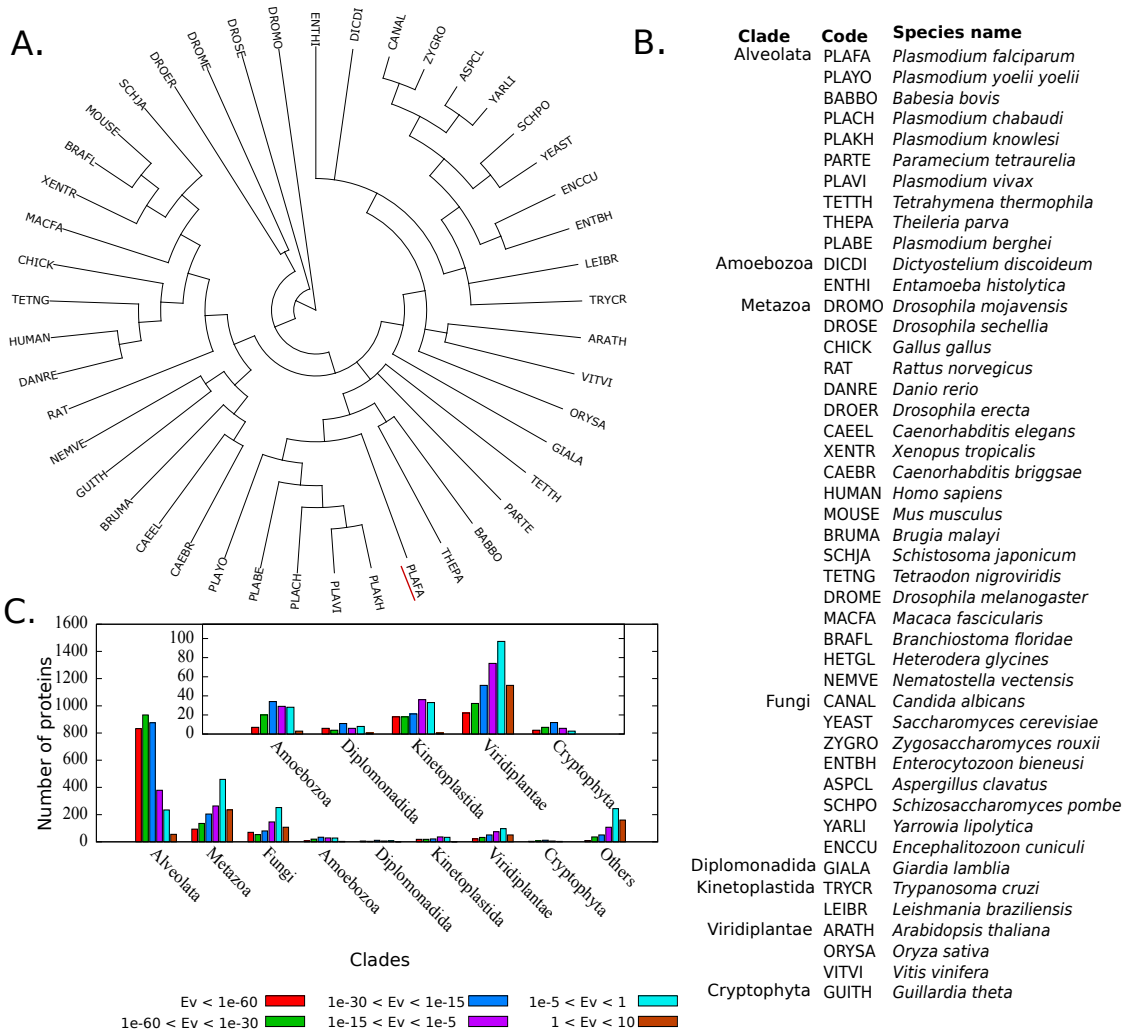


Figure 5.5: **Reference phylogenetic tree of 46 species**.

selected those homologous sequences that belong to the species in the tree, or eventually searched for homologous sequences that are close to the ones in the reference tree, see section 5.2.2. In Figure 5.5-C, we report the distribution of species used to predict domains, for all Pfam domains and all protein sequences in PlasmoDB. The 50.3% of the contribution is provided by homologs belonging to the Alveolata clade and the 49.7% of homologs is provided by other clades and among them Metazoa, Fungi, Viridiplantae and Kinetoplastida appear as the most represented (see the E-value distribution in Fig. 5.5C). A non negligeable contribution is also recorded from viruses, bacteria and archaea homologs, represented by the label other in Figure. 5.5C.

Table 5.2 compares domain predictions obtained by Pfam_{24} and CASH. The number of predictions for all *P. falciparum* proteins (Table 5.2-top) and for all proteins with

unknown function (Table 5.2-bottom) is reported with respect to a level of confidence estimated by an E-value (as in Table 5.1). We present in separated columns predictions with domain co-occurrence (Cooc) and the total number of predictions including those with no domain occurrence (T). Note that, we detailed CASH predictions by showing how many are obtained with profile hidden Markov models (pHMM) and with phylogenetic models (PHM). For phylogenetic models, we reported two values n/m, where n is the difference between CASH predictions and pHMM predictions and m is the number of predictions obtained exclusively from phylogenetic models.

Table 5.2: **Comparison of CASH and Pfam domain predictions.**

Domain predictions for all proteins in PlasmoDB								
			CASH					
	Pfam$_{24}$		**Total**		**pHMM**		**PHM**	
E-value	Cooc	T	Cooc	T	Cooc	T	Cooc	T
1e-60	297	810	566	1243	119	425	447/255	818/461
1e-30	908	1958	1433	2746	364	892	1069/704	1854/1193
1e-15	1687	3429	2365	4313	694	1486	1671/1135	2827/1876
1e-05	2827	5058	3579	5849	1466	2394	2113/1370	3455/2211
1	3178	**5414**	4832	**7196**	2262	3207	2570/1755	3989/2670
Domain predictions for proteins with unknown function in PlasmoDB								
1e-60	6	51	14	83	4	38	10/9	45/35
1e-30	23	177	104	378	16	113	88/76	265/220
1e-15	68	412	222	740	35	202	187/161	538/449
1e-05	246	803	490	1194	177	413	313/246	781/614
1	265	**824**	822	**1604**	327	576	495/409	1028/840

CASH predictions seem to exploit phylogenetic models in an exclusive manner. It agrees on the 96% of Pfam$_{24}$ predictions (Table 5.4), but it proposes a total of 1782 more predictions (this value was obtained by subtracting bold values in Table 5.2-top) at E-values ≤ 1 (this is the threshold used by dPUC for comparison). Notice that Pfam annotation is based on pHMMs, and that if we consider CASH pHMM predictions only, the number of CASH predicted domains is smaller than that of Pfam (3207 against 5414 proteins). This is because CASH predictions based on pHMM are often also obtained by phylogenetic models with a better E-value and counted as phylogenetic models predictions. In particular, about 780 (this value was obtained by subtracting bold values in Table

5.2-bottom) of the 1782 newly predicted domains belong to proteins that have unknown function in PlasmoDB, and the remaining 1002 domain predictions appear within known domain architectures.

We highlighted in Table 5.3 domain predictions found only by CASH. For this, we compared our prediction with those obtained by Pfam_{24}, CODD and dPuc, see section 5.2.8 for details of this comparison. In the top, we show new protein architectures proposed for the first time. In the Middle, enrichment of known protein architectures with additional domains. In the bottom, we present the number of functional domains unknown before to exist in the *P. falciparum* genome. Note that, the number of predictions is reported with respect to a level of confidence estimated by an E-value, as in Table 5.1, and the notation is the same presented in Table 5.2. By looking close at CASH predictions highlighted for the first time we notice that no contribution coming from pHMM seems to help, for all classes of predictions, most of the predictions are based on phylogenetic models. For instance, Table 5.3 (bottom) gives an account of 984 new domains (unknown to exist in *P. falciparum* before) of which 690 co-occur within known architectures. Of these domains, 558 are obtained exclusively by phylogenetic models, with no pHMM prediction (at E-value < 1).

To eliminate false positives from domain predictions, we used a score provided by a Support Vector Machine (SVM). The SVM discriminates potential annotations by evaluating which predicted domain is more probable among those displaying a small E-value, a sufficiently large domain length, the proximity to *P. falciparum* to the reference species generating the phylogenetic models leading to annotation, and a large consensus among models leading to the prediction. Scores issued by the SVM filter boosts weak domain predictions that positively satisfy several of the conditions and penalizes high confidence domains if the combination of conditions are not supporting the prediction. In Table 5.4, we show that SVM improves predictions (CASH) of about 8% over a score system based on best E-values (CASH_{BEv}). This means that CASH accumulates 33% of improvement on Pfam_{24} and 36.04% improvement on Pfam_{23}, where CODD and dPUC did less than 12%. Note that, these last two tools have been evaluated only on Pfam_{23}, see details in section 5.2.8.

Table 5.3: **Domain predictions found only by CASH**.

In proteins predicted for the first time						
	CASH					
	Total		**pHMM**		**PHM**	
E-value	Cooc	T	Cooc	T	Cooc	T
1e-60	1	2	0	0	1/1	2/2
1e-30	12	25	0	0	12/12	25/25
1e-15	25	60	0	0	25/24	60/59
1e-05	64	121	7	11	57/56	110/105
1	124	198	25	32	99/95	166/156
Enrichment of known protein architectures						
1e-60	4	6	0	0	4/4	6/6
1e-30	41	58	0	0	41/41	58/58
1e-15	103	134	0	0	103/102	134/132
1e-05	204	260	0	0	204/177	260/220
1	543	640	14	14	529/445	626/521
Predictions of new functional domains						
1e-60	6	14	1	2	5/4	12/11
1e-30	45	88	1	3	44/41	85/82
1e-15	136	251	1	3	135/130	248/243
1e-05	276	471	31	54	245/225	417/388
1	690	984	94	132	596/558	852/803

Table 5.4: **Improvements compared to Pfam (versions 23 and 24).**

	CASH	**CASH$_{\mathbf{BEv}}$**	**CODD**	**dPUC**
Pfam$_{23}$	36.04	28.02	11.7	10.03
Pfam$_{24}$	33.00	25.84	-	-

Also, we showed in Table 5.5 that CASH based on the usage of SVM performs better than CASH$_{\mathrm{BEv}}$ when it is compared with earlier results provided by Pfam$_{24}$, Pfam$_{23}$, CODD and dPUC. The percentage of agreement with other annotation tools is about 10% greater when we use CASH with scores provided by SVM.

More than half of the domains predicted at a given E-value are co-occurring domains (Table 5.2 and Table 5.3). The co-occurrence hypothesis greatly enhances the level of confidence on the prediction. This is because the majority of proteins are multidomains, domains do not form random combinations, and we observe fewer combinations than the statistically expected ones. This suggests functional cooperation, that is, two or more

Table 5.5: **Agreement between CASH and other annotation tools**.

	$Pfam_{24}$	**$Pfam_{23}$**	**CODD**	**dPUC**
CASH	96.22	84.51	81.33	82.95
$CASH_{BEv}$	85.04	73.53	70.13	72.49

domains can interact to determine the protein function. The use of this cooperation or domain co-occurrence to determine the most likely domain arrangement (architecture) of a given protein yields better performance, since weak domain predictions can be included into the protein architecture thank to the presence of another co-occurrent domain [20, 21, 22, 23].

Phylogenetic models used in domain prediction provide a profile of the protein that is usually more conserved than the one obtained by consensus with pHMM. We tested this observation with a global evaluation of physico-chemical properties conservation on CASH predictions realized over all proteins in PlasmoDB (Table 5.6-top), and realized over domain predictions obtained only by CASH (Table 5.6-bottom). Physico-chemical conservations in Table 5.6 were computed as described in section 5.2.7, and hold values indicate better performance. Different amino acids groups are analyzed for the two modeling approaches: profile hidden Markov models (pHMM) and phylogenetic models (PHM). It is not surprising that hydrophobic amino acids are the most conserved in divergent domain sequences and that they appear as the most conserved in our predictions. But the striking fact is that by comparing PHM and pHMM we obtain in a systematic way that Accuracy, Sensitivity, Specificity and PPV are better for phylogenetic profiles.

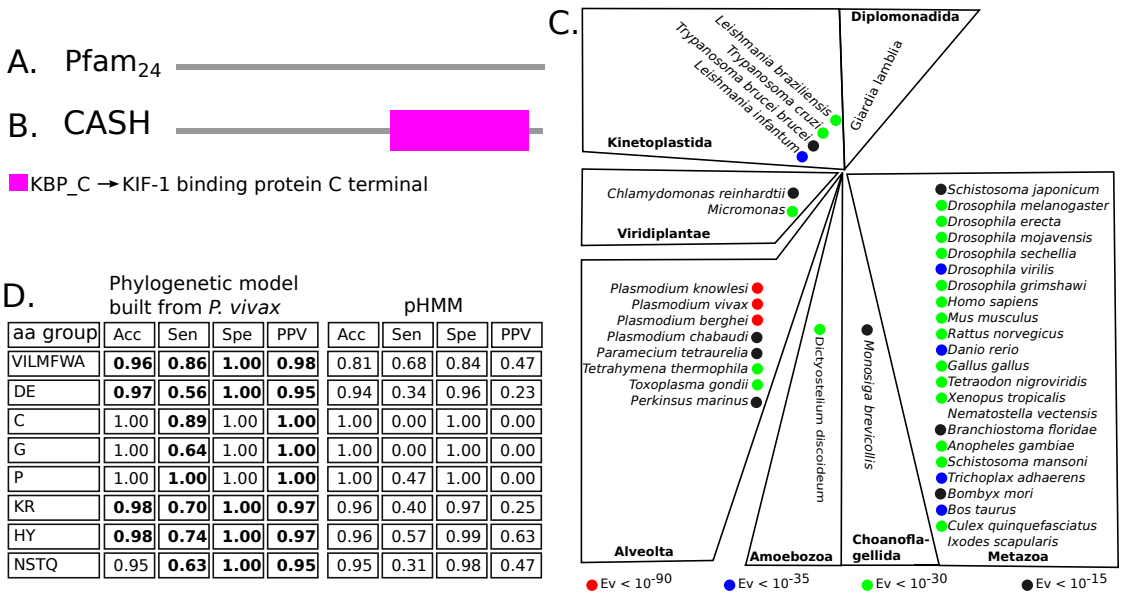
According to our results, CASH has proposed new architectures for protein with no annotation, and also proposed additional domains for annotated proteins. We analyze these kind of predictions by looking close two examples showed in Figures 5.6 and 5.7. Figure 5.6-B show that "KIF-1 binding protein C terminal" domain was predicted for the *P. falciparum* protein MAL13P1.370, while no prediction was obtained by Pfam, see Figure 5.6-A. CASH prediction was detected by a phylogenetic model built from *Plasmodium vivax* sequence with E-value 3*e*-96. No homologue of this protein is known in *P. falciparum*. We analyze predictions obtained by other phylogenetic models, and we observe that several models support this prediction, see Figure 5.6-C where color dots highlight

Table 5.6: **Comparison of Physico-chemical conservation**

Physico-chemical conservation analysis of all proteins in PlasmoDB								
	PHM				**pHMM**			
Amino acid group	Acc	Sen	Spe	PPV	Acc	Sen	Spe	PPV
VILMFWA	**0.92**	**0.84**	**0.93**	**0.82**	0.88	0.74	0.92	0.76
DE	0.97	**0.68**	0.98	0.69	0.97	0.41	**0.99**	**0.71**
C	1.00	**0.63**	1.00	**0.91**	1.00	0.45	1.00	0.90
G	0.99	**0.77**	0.99	**0.71**	0.99	0.61	0.99	0.69
P	0.99	**0.75**	0.99	0.70	0.99	0.52	**1.00**	**0.73**
KR	0.96	**0.66**	0.98	0.72	0.96	0.38	**0.99**	0.72
HY	0.99	**0.64**	0.99	0.77	0.99	0.37	1.00	**0.78**
NSTQ	0.94	**0.58**	0.97	0.71	0.94	0.31	**0.98**	0.71
Physico-chemical conservation analysis of proteins predicted for the first time by CASH								
VILMFWA	**0.84**	**0.74**	**0.85**	**0.57**	0.80	0.71	0.81	0.51
DE	0.95	**0.48**	0.96	**0.44**	0.95	0.44	0.96	0.41
C	1.00	0.69	1.00	**0.81**	1.00	**0.74**	1.00	0.79
G	0.98	0.67	0.98	**0.45**	0.98	**0.68**	0.98	0.38
P	0.99	0.71	0.99	**0.52**	0.99	0.71	0.99	0.47
KR	0.94	**0.48**	0.96	**0.44**	0.94	0.44	0.96	0.40
HY	0.98	0.47	**0.99**	**0.54**	0.98	**0.48**	0.98	0.50
NSTQ	0.90	**0.42**	0.94	**0.48**	0.90	0.38	0.94	0.45

predictions obtained with different e-value ranges. The localization of species showed in Figure 5.6-C was inspired by Figure 1 in [124]. Pfam did not predict the domain KBP_C in the protein MAL13P1.370 with an acceptable E-value, according to Pfam "gathering" thresholds. We show in Figure 5.6-D (left) that the physico-chemical conservation between the profile of *P. vivax* and the sequence of *P. falciparum* (MAL13P1.370) is much higher than the conservation obtained with pHMM showed in Figure 5.6-D (right), in both physico-chemical conservations were computed as in Table 5.6. Because this higher conservation the phylogenetic model built from *P. vivax* sequence was able to detect the domain KBP_C in the protein MAL13P1.370, while the pHMM has failed.

To illustrated how CASH can enrich architectures of known proteins we consider predictions for the *P. falciparum* protein PFE0100w showed in Figure 5.7. Pfam was able to detect only the domain VPS11_C (Figure 5.7-A), while CASH detected two additional domains: zf-C3HC4 and clathrin (Figure 5.7-B). In the clathrin domain prediction one observes that phylogenetic models with highest E-values are spread, explaining the difficulty



D.

	Phylogenetic model built from *P. vivax*				pHMM			
aa group	Acc	Sen	Spe	PPV	Acc	Sen	Spe	PPV
VILMFWA	**0.96**	**0.86**	**1.00**	**0.98**	0.81	0.68	0.84	0.47
DE	**0.97**	**0.56**	**1.00**	**0.95**	0.94	0.34	0.96	0.23
C	1.00	**0.89**	1.00	**1.00**	1.00	0.00	1.00	0.00
G	1.00	**0.64**	1.00	**1.00**	1.00	0.00	1.00	0.00
P	1.00	**1.00**	1.00	**1.00**	1.00	0.47	1.00	0.00
KR	**0.98**	**0.70**	**1.00**	**0.97**	0.96	0.40	0.97	0.25
HY	**0.98**	**0.74**	**1.00**	**0.97**	0.96	0.57	0.99	0.63
NSTQ	0.95	**0.63**	**1.00**	**0.95**	0.95	0.31	0.98	0.47

Figure 5.6: **CASH domain prediction for MAL13P1.370 a protein with no annotation.**

of annotating this protein with a pHMM approach (Figure 5.7-C). As showed in the first example, physico-chemical conservation is much higher for the phylogenetic model prediction than for the pHMM prediction, see Figure 5.7-D, where the profile generated by a *Dichtyostelium discoideum* sequence predicted homology in PFE0100w with E-value 9*e*-24 (Figure 5.7-D (bottom)), while the pHMM made a prediction with E-value 1.9*e*-08 (Figure 5.7-D (top)). Physico-chemical conservation is illustrated graphically for the prediction based on pHMM (Figure 5.7-D (top)) and the one based on *D. discoideum* profile (Figure 5.7-D (bottom)). Each prediction is represented by a display showing physico-chemical conservation for each position of the multiple sequence alignment (MSA) generating the profile. For each position, a color scale maps the physico-chemical class most represented at that position. The *P. falciparum* sequence is reported for both display together with the physico-chemical match, indicated by a '*', between the corresponding residue in the *P. falciparum* sequence and the alignment, if existing.

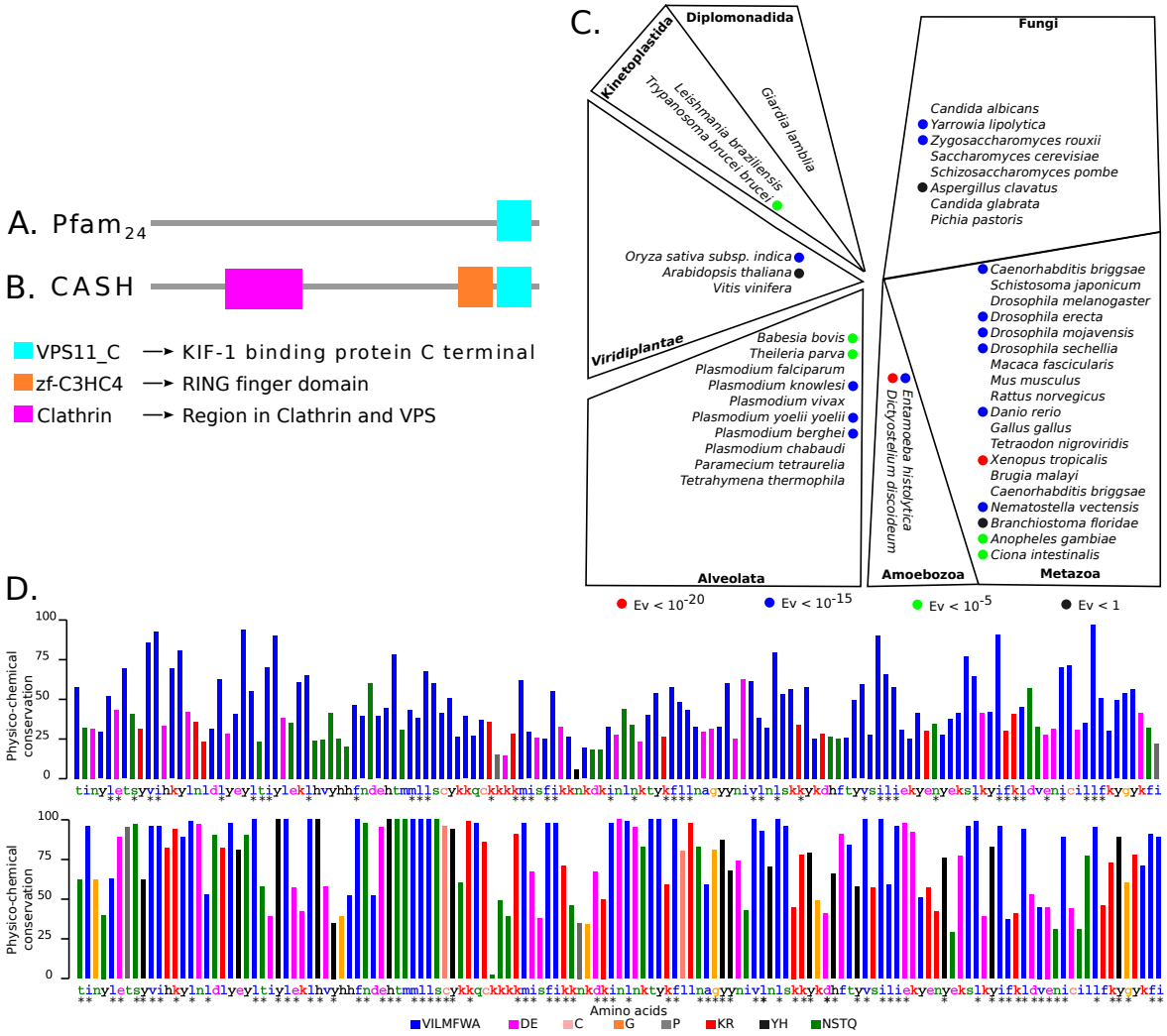


Figure 5.7: **PFE0100w architecture enrichment.**

5.4 Conclusion

We demonstrated that when sequences in a protein family are too divergent or too conserved, signals of homology are easier to trace when profile are constructed on single species rather than by consensus. We showed that this situation often occurs and that by combining in a unique tool pHMMs and phylogenetic models one can reinforce the predictive power. In CASH, the two kinds of models are run in parallel and compete against best signals detection.

Several observations can be drawn from our results. First, the high number of predictions that could not be identified by pHMM approaches underlies that a specific understanding of the evolutionary process extracted from specific species can be of essential

help in remote homology identification. This suggests that protein evolution follows a probably restricted number of pathways, that phylogenetic proximity is not a universal criteria to understand protein evolution and that global constraints do not always capture the essence of protein constraints.

At this regard, it has been observed that evolutionary pressures on networks of residues (that is, groups of residues which are physically connected in the structure) within a protein family play an important role in preserving or changing the protein function, structure, mechanical and folding properties [125, 126, 127, 128, 129]. In particular, the coexistence of several networks within a protein family and the sharing of residues between networks make the relations among residues an entangled evolutionary process, which is therefore highly non-random. The existence of possibly strong constraints supports the idea that the number of evolutionary solutions should be limited.

This observation, stated before for mutational events in bacterial and viral species, has, at our knowledge, never been reported nor exploited before in the computational search of highly divergent proteins. A reminiscent idea was present in the manual search strategy used in [130].

CASH achieved excellent results over *P. falciparum* genome. It predicts domains for 70% of proteins in PlasmoDB. However, our goal is to contribute for functional annotation of proteins in *P. falciparum*. For this, we are carrying out a carefully manually analysis on CASH domain predictions to suggest functional annotation for unannotated proteins, and possibly reannotate proteins with new CASH domain predictions. Moreover, We are analyzing *P. falciparum* domain predictions to provide insights about its biological processes. Among predicted domains, we observed that some are over-represented in the *P. falciparum* genome. For instance, there are 134 domains with more than 3 and up to 167 occurrences. Among them, there is a considerable number that has been never detected before by other methods, see Table C.1. Also, we observed that some domain combinations (some of them detected by the first time) seem to be important for the parasite, since they occur more than 3 times, see Table C.2. We believe that interesting information can be drawn from a careful analysis of our domain predictions.

CASH has been applied to the *P. falciparum* genome but its strategy can be applied to any other genome. In fact, the framework is highly general and independent on *P. falci-*

parum characteristics. As a future step we would like to create phylogenetic models for all species associated to the each Pfam domain. From this, the user could be choose between to run CASH by using all models or to select a species set of interest for his/her study.

Technically speaking, CASH is realized with the usage of an SVM. This choice can be revisited and other decision strategies could be employed instead, like multi-response linear regression [131] for instance. Investigation of new strategies will be realized in future work.

Chapter 6

General conclusions and future work

One of the fundamental challenges in computational biology is the identification of evolutionary related proteins for which primary sequences have significantly diverged (remote homologies). Remote homology detection is the problem of finding homology between sequences (proteins or genes), when the actual sequence identity is low (usually, lower than 30%). In a general way, homology detection methods are today quite important to aid for sequence annotation, protein classification and to guide laboratory experiments. Without the development of these methods the detection of homology from sequence would not be possible. However, remote protein homology detection is considered as a problem that has not been resolved yet in bioinformatics. There is still a large number of proteins with no identified function as well as a significant part of the genome of some organisms is still remains unannotated. This is the case of *Plasmodium Falciparum*, organism that causes malaria in human, whose the function of almost half of proteins is unknown. The identification of pathogenic proteins could help to derive the design of new drugs as medicine and the creation of specific cures for the disease.

The low similarities among remote homologous proteins makes this task a challenge. Some researchers have used extra information, such as structural protein features to develop more effective tools. However, this information is not available for most of existing proteins, and it cannot be used in large-scale annotation system. Another alternative is the use of discriminative methods that archive better performance than traditional methods. However, their high computational time made these methods unpopular among biologists.

Since sequence information is abundant contrary to the structural one, new approaches for remote homology detection should try to explore in a more sophisticated way properties coming from protein sequences to provide an effective annotation system and a useful protein classification tool. In this scenario, we proposed two new methods. The first, called ILP-SVM Homology [48] is a method for remote homologous protein classification. It proposes a novel representation for homologous properties based on the first-order logic language. The use of this language allow us to represent important sequential properties and relate them by exploring the relational power of the language. From this, WARMR (an ILP system) induces essential logic rules that aim to explain what homologous proteins have in common. These rules can be use to discriminate between homologous and non-homologous sequences. For this, we convert them into a feature vector that is used to train propositional models, such as DTs and SVMs. Assessed on benchmarch databases, our results show a superior/equal performance when compared to state-of-art methods. However, through our experiments we conclude that a suitable logical language of representation and an appropriated inductive system can mine important rules helping to detect more remote homologous proteins and, in the same time, providing insights into conserved features of homologous protein families.

As a second method, we propose CASH - Combination of Annotations by Species and pHMMs [49], that is specialized in annotation of highly divergent genomes. CASH is a pipeline that combines several computational approaches to provide a more effective annotation system. We propose to explore alternative evolutionary paths to represent each known protein domain in the Pfam database [50]. In fact, CASH creates an ensemble of models for each Pfam domain by combining additional profiles constructed out of similar sequences obtained from different species, and existing pHMMs. The output of these models are processed and transformed into features that are used to train a meta-classifier that assigns a confidence score to each domain prediction. Based on this score and other sequential properties, CASH proposes a domain architecture for a given query sequence by resolving a multi-objective optimization problem. The analysis of our results highlight that CASH is an effective large-scale annotation system and that it is able to decrease the number of unannotated proteins in *Plasmodium Falciparum*. The use of additional profiles created from a large and differentiated panel of homologous sequences was able to

enrich the representation of Pfam domains by improving the performance of the annotation system. Domain co-occurrences played a key role in the annotation process, since the presence of low confidence domains in a query sequence could be asserted thanks to the presence of other co-occurrent domains. Moreover, the strategy of resolving protein architectures by optimizing multi-objective functions seems to be a suitable approach, since several aspects of the problem can be considered.

In the future we wish to improve our methods and to study a way of connecting them. The performance of both methods could be improved applying other machine learning algorithms, such as multi-response linear regression [131] in CASH, and MineSeqLog [97] in ILP-SVM Homology. Another alternative is to explore sequential protein properties such as the presence of functional sites into homologous proteins, as done in [6], and available structural properties, as done in [132]. A possibility for combining our methods is to incorporate ILP-SVM Homology as a new feature for the CASH's meta-classifier training. For instance, for each Pfam domain we could learn its most frequent patterns, and use the outputs to discriminate query sequences through a score that computes the matches between the query sequence and the most frequent patterns of the domain. As these patterns are learnt from a protein alignment, we shall decide which alignments to take into account, that is, consensus alignments used to build pHMMs or specific alignments used to build phylogenetic models. Unfortunately, to consider all available alignments would make CASH time consuming when applied to a large-scale annotation (this was already observed in our computational experiments). However, the strategy could be applied to a reduced number of protein domains of interest, as those involved in the transcription process or the RNA silencing for instance.

Motivated by our excellent results over *P. falciparum* we would like to apply CASH to other highly divergent genomes, like Diatoms for instance, upon which around 40-50% of proteins are unannotated. Diatoms are organism found throughout marine and freshwater environments, and are believed to be responsible for around one-fifth of the primary productivity on Earth [133]. To the best of our knowledge, diatoms genomes have been annotated by using only sequence similarity searching methods and traditional generative models. We believe that CASH could decrease significantly the number of unannotated proteins contributing for understanding some important cellular mechanisms

in Diatom species.

Finally, we would like that CASH be used by the biologist community in place of traditional methods. For this, we will develop a user friendly web-server that will allow biologists to analyze their protein sequences.

Appendix A

List of representative species used in CASH system

Table A.1 shows 46 species that were selected from the eukaryotic life tree in order to build phylogenetic models (first column). If one species does not exists into a given Pfam domain, we try to replace it with a similar species (second column)

Table A.1: **Representative species**.

Reference species	**Alternative species**
Plasmodium falciparum	*Plasmodium falciparum (strain isolate Nig32/Nigeria, isolate ro-33/Ghana, isolate Dd2, isolate FCH-5, isolate kf1916, isolate 3D7, isolate 7G8, isolate tak 9, isolate 311, isolate FCBR/Columbia, isolate Camp/Malaysia, isolate CDC/Honduras, isolate FC27/Papua New Guinea, isolate FCR-3/Gambia, isolate thtn/Thailand, isolate imr143, isolate fid3/India, isolate K1/Thailand, isolate le5, isolate mad20/Papua New Guinea, isolate NF7/Ghana, isolate NF54, isolate Palo Alto/Uganda, isolate FcB1/Columbia, isolate fcm17/Senegal, isolate t4/Thailand, isolate v1, isolate Wellcome, isolate HB3, isolate mad71/Papua New Guinea)*
Plasmodium knowlesi	*Plasmodium knowlesi (strain H, nuri)*
Plasmodium vivax	*Plasmodium vivax (Brazil I, India VII, IQ07, Mauritania I, North Korean, strain Belem, strain Salvador I)*
Plasmodium yoelii yoelii	*Plasmodium yoelii (17, killicki, nigeriensis, YM)*
Plasmodium berghei	*Plasmodium berghei (strain Anka)*
Plasmodium chabaudi	*Plasmodium chabaudi adami*
Babesia bovis	*Babesia bigemina, Babesia canis, Babesia gibsoni, Babesia divergens, Babesia odocoilei, Babesia microti*
Theileria parva	*Theileria annulata, Theileria buffeli, Theileria cervi, Theileria lestoquardi, Theileria mutans, Theileria orientalis, Theileria sergenti (isolate Ikeda), Theileria sergenti, Theileria sergenti (isolate Chitose), Theileria taurotragi*
Paramecium caudatum	*Paramecium primaurelia, Paramecium tetraurelia, Paramecium bursaria Chlorella virus (1, AR158, NC1A, FR483, IL3A, MT325, NY2A, XZ-6E)*
Tetrahymena americanus	*Tetrahymena australis, Tetrahymena borealis, Tetrahymena capricornis, Tetrahymena canadensis, Tetrahymena cosmopolitanis, Tetrahymena caudata, Tetrahymena elliotti, Tetrahymena furgasoni, Tetrahymena hyperangularis, Tetrahymena leucophrys, Tetrahymena malaccensis, Tetrahymena mimbres, Tetrahymena nanneyi, Tetrahymena nippisingi, Tetrahymena patula, Tetrahymena pigmentosa, Tetrahymena paravorax, Tetrahymena pyriformis, Tetrahymena rostrata, Tetrahymena sonneborni, Tetrahymena thermophila, Tetrahymena tropicalis*

Candida albicans	*Candida apicola, Candida antarctica, Candida albicans (strain WO-1), Candida boidinii, Candida dubliniensis (strain CD36/CBS 7987/NCPF 3949/NRRL Y-17841), Candida glabrata, Candida glycerinogenes, Candida maltosa, Candida milleri, Candida norvegensis, Candida oleophila, Candida parapsilosis, Candida rugosa, Candida sp. (strain HA167), Candida shehatae, Candida stellata, Candida tenuis, Candida tropicalis, Candida tsukubaensis, Candida tropicalis (strain ATCC MYA-3404/T1), Candida zemplinina*
Yarrowia lipolytica	*Yarrowia lipolytica (strain 11p-1, 11P-2, 12p-1, 25L, 29L, 3m-1, 3w-1, 8c-1, 8p-1, 8s-3, 9w-1, D30L, DKl1, KS1, M15S06, SM-22, SSJO2005, TFM01)*
Zygosaccharomyces rouxii	*Zygosaccharomyces rouxii (strain ATCC 2623/CBS 732/IFO 1130/NBRC 1623/NCYC 568), Zygosaccharomyces bailii, Zygosaccharomyces bisporus, Zygosaccharomyces kombuchaensis, Zygosaccharomyces lentus, Zygosaccharomyces machadoi, Zygosaccharomyces mellis, Zygosaccharomyces microellipsoides, Zygosaccharomyces mrakii, Zygosaccharomyces pseudorouxii*
Entamoeba histolytica	*Entamoeba histolytica (2759071, DS4-868, HM-1:IMSS, HM-3:IMSS, KU27, KU48, KU50, MS96-3382), Entamoeba dispar, Entamoeba invadens*
Dictyostelium discoideum	*Dictyostelium citrinum, Dictyostelium mucoroides, Dictyostelium purpureum, Dictyostelium sp. (strain GA11)*
Giardia lamblia	*Giardia ardeae, Giardia intestinalis, Giardia microti, Giardia muris, Giardia psittaci, Giardia sp. AM1, Giardia sp. BR1, Giardia sp. MM1, Giardia sp. QE1, Giardia sp. quenda, Giardia sp. SM-2004, Giardia sp. Swemouse185*
Saccharomyces cerevisiae	*Saccharomyces cerevisiae (A364A, CAT-1, CBS 7960, CEN.PK113-7D, CLIB215, CLIB324, CLIB382, EC9-8, FL100, G600, I14, IL-01, KRY8, M22, MMY112, NC-02, PW5, Sigma1278b, SK1, (strain ATCC 204508/S288c), (strain AWRI1631), (strain AWRI796), (strain FostersB), (strain FostersO), (strain JAY291), (strain Kyokai no. 7/NBRC 101557), (strain Lalvin EC1118/Prise de mousse), (strain Lalvin QA23), (strain RM11-1a), (strain VIN 13), (strain YJM789), (strain Zymaflore VL3), T7, T73, UC5, W303, WE372, Y10, Y12, Y9, YJM269, YJM280, YJM320, YJM326, YJM421, YJM428, YJM451, YJM653, YJSH1, YPS1009, YPS163, Saccharomyces diastaticus*
Schizosaccharomyces pombe	*Schizosaccharomyces pombe (DM3650, DM3755, DM3757, NCYC132, OY26, SPK1820, (strain 972/ATCC 24843), strain SPY73 975 h+, Schizosaccharomyces japonicus, Schizosaccharomyces japonicus (strain yFS275/FY16936), Schizosaccharomyces kambucha, Schizosaccharomyces octosporus*
Guillardia theta	*Galdieria sulphuraria, Cyanophora paradoxa, Reticulomyxa filosa*
Enterocytozoon bieneusi	*Enterocytozoon bieneusi (strain H348), Enterocytozoon bieneus, Enterocytozoon salmonis, Enterocytozoon sp. (IS2005R, IS2005S, IS2005T, IS2005U, IS2005V, IS2005W, ST-2009a)*
Encephalitozoon cuniculi	*Encephalitozoon hellem, Encephalitozoon intestinalis*
Aspergillus aculeatus	*Aspergillus amstelodami, Aspergillus awamori, Aspergillus clavatus, Aspergillus fumigatus (strain CEA10/CBS 144.89/FGSC A1163), Aspergillus ficuum, Aspergillus flavus (strain ATCC 200026/FGSC A1120/NRRL 3357/JCM 12722/SRRC 167), Aspergillus fumigatus, Aspergillus giganteus, Aspergillus japonicus, Aspergillus kawachi, Aspergillus niger (strain CBS 513.88/FGSC A1513), Aspergillus nomius, Aspergillus oryzae, Aspergillus parasiticus, Aspergillus pallidus, Aspergillus pseudotamarii, Aspergillus restrictus, Aspergillus saitoi, Aspergillus shirousami, Aspergillus sojae, Aspergillus terreus (strain ATCC 20542/MF4845, NIH 2624/FGSC A1156), Aspergillus tubingensis, Aspergillus viridinutans, Aspergillus wentii*
Caenorhabditis brenneri	*Caenorhabditis briggsae, Caenorhabditis elegans, Caenorhabditis japonica, Caenorhabditis remanei*
Leishmania amazonensis	*Leishmania braziliensis, Leishmania chagasi, Leishmania donovani, Leishmania enriettii, Leishmania guyanensis, Leishmania infantum, Leishmania major, Leishmania mexicana, Leishmania peruviana, Leishmania pifanoi, Leishmania tarentolae, Leishmania tropica, Leishmania RNA virus 1 - 1 (isolate Leishmania guyanensis)*
Trypanosoma brucei brucei	*Trypanosoma brucei gambiense, Trypanosoma brucei rhodesiense, Trypanosoma congolense, Trypanosoma cruzi, Trypanosoma equiperdum, Trypanosoma evansi, Trypanosoma lewisi, Trypanosoma rangeli, Trypanosoma vivax, Trypanosomatidae*
Schistosoma japonicum	*Schistosoma bovis, Schistosoma haematobium, Schistosoma mansoni*

Drosophila melanogaster	*Drosophila adunca, Drosophila acanthoptera, Drosophila adiastola, Drosophila americana, Drosophila affinidisjuncta, Drosophila affinis, Drosophila algonquin, Drosophila ambigua, Drosophila ananassae, Drosophila atripex, Drosophila arizonae, Drosophila austrosaltans, Drosophila athabasca, Drosophila auraria, Drosophila aracataca, Drosophila azteca, Drosophila bakoue, Drosophila bocqueti, Drosophila bifasciata, Drosophila borealis, Drosophila bipectinata, Drosophila busckii, Drosophila buzzatii, Drosophila capricorni, Drosophila crassifemur, Drosophila cyrtoloma, Drosophila dasycnemia, Drosophila differens, Drosophila dossoui, Drosophila disjuncta, Drosophila ercepeae*
Drosophila elegans	*Drosophila emarginata, Drosophila equinoxialis, Drosophila erecta, Drosophila eugracilis, Drosophila ezoana, Drosophila ficusphila, Drosophila flavomontana, Drosophila fima, Drosophila funebris, Drosophila grimshawi, Drosophila guanche, Drosophila hawaiiensis, Drosophila heteroneura, Drosophila hydei, Drosophila iki, Drosophila immigrans, Drosophila insularis, Drosophila jambulina, Drosophila kanekoi, Drosophila kikkawai, Drosophila kitumensis, Drosophila kuntzei, Drosophila lacicola, Drosophila lebanonensis, Drosophila lineosetae, Drosophila limbata, Drosophila lini, Drosophila lowei, Drosophila littoralis, Drosophila lusaltans, Drosophila lutescens*
Drosophila mauritiana	*Drosophila microlabis, Drosophila madeirensis, Drosophila miranda, Drosophila milleri, Drosophila mimica, Drosophila montana, Drosophila mojavensis, Drosophila mercatorum, Drosophila mediostriata, Drosophila mettleri, Drosophila mulleri, Drosophila melanica, Drosophila mayaguana, Drosophila navojoa, Drosophila neocordata, Drosophila nebulosa, Drosophila nasuta F, Drosophila nigra, Drosophila narragansett, Drosophila nasuta, Drosophila obscura, Drosophila orena, Drosophila pseudoobscura bogotana, Drosophila pinicola, Drosophila persimilis, Drosophila picticornis, Drosophila planitibia, Drosophila punjabiensis, Drosophila petalopeza, Drosophila prosaltans, Drosophila pseudoobscura pseudoobscura, Drosophila paulistorum*
Drosophila pavlovskiana	*Drosophila robusta, Drosophila repleta, Drosophila saltans, Drosophila subsaltans, Drosophila sucinea, Drosophila sechellia, Drosophila simulans, Drosophila silvestris, Drosophila soonae, Drosophila sp., Drosophila serrata, Drosophila subsilvestris, Drosophila sturtevanti, Drosophila subobscura, Drosophila tanythrix, Drosophila teissieri, Drosophila takahashii, Drosophila tolteca, Drosophila tropicalis, Drosophila tristis, Drosophila tsacasi, Drosophila varians, Drosophila virilis, Drosophila vallismaia, Drosophila wheeleri, Drosophila willistoni, Drosophila yakuba*
Homo sapiens	*Homo sapiens neanderthalensis*
Macaca mulatta	*Macaca arctoides, Macaca assamensis, Macaca balantak, Macaca balantak x tonkeana, Macaca brunnescens, Macaca cyclopis, Macaca fascicularis, Macaca fuscata, Macaca hecki, Macaca hecki x tonkeana, Macaca leonina, Macaca maura, Macaca maura tonkeana, Macaca munzala, Macaca nemestrina, Macaca nigra, Macaca nigrescens, Macaca ochreata, Macaca pagensis, Macaca radiata, Macaca siberu, Macaca silenus, Macaca sinica, Macaca sp., Macaca speciosa, Macaca sylvanus, Macaca thibetana, Macaca tonkeana*
Mus musculus	*Mus musculus albula, Mus musculus bactrianus, Mus musculus brevirostris, Mus musculus castaneus, Mus musculus domesticus, Mus musculus gentilulus, Mus musculus homourus, Mus musculus molossinus, Mus musculus musculus castaneus, Mus musculus musculus domesticus, Mus musculus wagneri*
Rattus norvegicus	*Rattus norvegicus albus*
Danio rerio	*Danio aff. (albolineatus, dangila DP-2005, tweediei), Danio albolineatus (pulcher), Danio cf. dangila CTOL01570, Danio cf. rerio Assam, Danio choprai, Danio dangila, Danio feegradei, Danio kerri, Danio kyathit, Danio margaritatus, Danio nigrofasciatus, Danio roseus, Danio sp. (Bangladesh, CTOL02798, CTOL03307, Hikari, Ozelot, pantheri, SH-2001, snakeskin, SSH-2005, tinwini, tweediei)*
Gallus gallus	*Gallus gallus bankiva, Gallus gallus jabouillei, Gallus gallus murghi, Gallus gallus spadiceus*
Tetraodon nigroviridis	*Tetraodon biocellatus, Tetraodon cutcutia, Tetraodon fangi, Tetraodon fluviatilis, Tetraodon mbu, Tetraodon miurus, Tetraodon palembangensis*
Xenopus tropicalis	*Xenopus (Silurana) cf. tropicalis BJE-2004, epitropicalis, sp. BOLD:AAH0940, sp. LIN463-07, sp. LIN464-07, sp. new tetraploid 1, sp. new tetraploid 2*
Brugia malayi	*Brugia buckleyi, Brugia cf. malayi ex canine (Kadakkarappally 1/2), Brugia pahangi, Brugia patei, Brugia timori*
Heterodera glycines	*Heterodera arenaria, Heterodera aucklandica, Heterodera australis, Heterodera avenae, Heterodera betae, Heterodera bifenestra, Heterodera cajani, Heterodera cardiolata, Heterodera carotae, Heterodera cf. (graminophila TSH-2005, iri TSH-2005, medicaginis TSH-2005), Heterodera ciceri, Heterodera circeae, Heterodera cruciferae, Heterodera cynodontis, Heterodera cyperi, Heterodera elachista, Heterodera fici, Heterodera filipjevi*

Caenorhabditis briggsae	*Caenorhabditis angaria, Caenorhabditis brenneri, Caenorhabditis drosophilae, Caenorhabditis elegans, Caenorhabditis japonica, Caenorhabditis maupasi, Caenorhabditis plicata, Caenorhabditis remanei, Caenorhabditis sonorae, Caenorhabditis (11 MAF-2010, 12 KK-2010, 5 AC-2008, 5 DRD-2008, 7 MAF-2007, 9 AC-2009, 9 MAF-2010, DF5070, JLR-2009, JU727, SB341)*
Nematostella vectensis	*Nematostella sp. JVK-2006*
Branchiostoma belcheri	*Ž03a Branchiostoma californiense, Branchiostoma floridae, Branchiostoma japonicum, Branchiostoma lanceolatum, Branchiostoma malayanum*
Oryza sativa	*Oryza sativa subsp. indica, Oryza sativa subsp. japonica*
Vitis vinifera	*Vitis thunbergii, Vitis tiliifolia, Vitis treleasei, Vitis vulpina, Vitis wilsonae, Vitis champinii, Vitis doaniana, Vitis yeshanensis*
Arabidopsis thaliana	*Arabidopsis arenicola, Arabidopsis arenosa, Arabidopsis cebennensis, Arabidopsis croatica, Arabidopsis halleri, Arabidopsis lyrata, Arabidopsis neglecta, Arabidopsis pedemontana, Arabidopsis petrogena, Arabidopsis suecica*

Appendix B

List of Pfam Ribosomal families

Table B.1 shows 109 Pfam families used in the building of phylogenetic reference tree. The first column correspond to the Pfam accession number and the second is a small description about the family.)

Table B.1: **List of Pfam ribosomal proteins**.

Pfam Accession Number	**Description**
PF00163	Ribosomal protein S4/S9 Nterminal domain
PF03947	Ribosomal Proteins L2, Cterminal domain
PF00177	Ribosomal protein S7p/S5e
PF00164	Ribosomal protein S12
PF00252	Ribosomal protein L16p/L10e
PF00181	Ribosomal Proteins L2, RNA binding domain
PF00318	Ribosomal protein S2
PF00203	Ribosomal protein S19
PF00189	Ribosomal protein S3, Cterminal domain
PF00237	Ribosomal protein L22p/L17e
PF00238	Ribosomal protein L14p/L23e
PF00410	Ribosomal protein S8
PF00411	Ribosomal protein S11
PF00253	Ribosomal protein S14p/S29e
PF00276	Ribosomal protein L23
PF00312	Ribosomal protein S15
PF00416	Ribosomal protein S13/S18
PF00828	Ribosomal protein L18e/L15
PF00673	Ribosomal L5P family Cterminus
PF01084	Ribosomal protein S18
PF00281	Ribosomal protein L5
PF00338	Ribosomal protein S10p/S20e
PF00466	Ribosomal protein L10
PF00453	Ribosomal protein L20
PF00333	Ribosomal protein S5, Nterminal domain
PF00347	Ribosomal protein L6
PF00380	Ribosomal protein S9/S16
PF00687	Ribosomal protein L1p/L10e family
PF00861	Ribosomal L18p/L5e family
PF00573	Ribosomal protein L4/L1 family
PF03946	Ribosomal protein L11, Nterminal domain
PF03719	Ribosomal protein S5, Cterminal domain
PF00572	Ribosomal protein L13
PF00297	Ribosomal protein L3
PF00298	Ribosomal protein L11, RNA binding domain
PF00366	Ribosomal protein S17
PF00886	Ribosomal protein S16

Table B.1: **List of Pfam ribosomal proteins**.

Pfam Accession Number	**Description**
PF00831	Ribosomal L29 protein
PF00417	Ribosomal protein S3, Nterminal domain
PF01783	Ribosomal L32p protein family
PF00471	Ribosomal protein L33
PF00542	Ribosomal protein L7/L12 Cterminal domain
PF01245	Ribosomal protein L19
PF01196	Ribosomal protein L17
PF01016	Ribosomal L27 protein
PF00444	Ribosomal protein L36
PF01250	Ribosomal protein S6
PF00829	Ribosomal prokaryotic L21 protein
PF00327	Ribosomal protein L30p/L7e
PF01281	Ribosomal protein L9, Nterminal domain
PF00830	Ribosomal L28 family
PF01632	Ribosomal protein L35
PF01197	Ribosomal protein L31
PF03948	Ribosomal protein L9, Cterminal domain
PF01649	Ribosomal protein S20
PF00468	Ribosomal protein L34
PF01165	Ribosomal protein S21
PF02482	Sigma 54 modulation protein / S30EA Ribosomal protein
PF01386	Ribosomal L25p family
PF01248	Ribosomal protein L7Ae/L30e/S12e/Gadd45 family
PF00428	60s Acidic Ribosomal protein
PF00827	Ribosomal L15
PF01655	Ribosomal protein L32
PF01599	Ribosomal protein S27a
PF01015	Ribosomal S3Ae family
PF01201	Ribosomal protein S8e
PF00900	Ribosomal family S4e
PF01092	Ribosomal protein S6e
PF01280	Ribosomal protein L19e
PF01667	Ribosomal protein S27
PF01090	Ribosomal protein S19e
PF01246	Ribosomal protein L24e
PF01157	Ribosomal protein L21e
PF01198	Ribosomal protein L31e
PF08069	Ribosomal S13/S15 Nterminal domain
PF00935	Ribosomal protein L44
PF01020	Ribosomal L40e family
PF00833	Ribosomal S17
PF01282	Ribosomal protein S24e
PF01780	Ribosomal L37ae protein family
PF01775	Ribosomal L18ae/LX protein domain
PF01200	Ribosomal protein S28e
PF01199	Ribosomal protein L34e
PF01907	Ribosomal protein L37e
PF01294	Ribosomal protein L13e
PF01283	Ribosomal protein S26e
PF01251	Ribosomal protein S7e
PF00832	Ribosomal L39 protein
PF01929	Ribosomal protein L14
PF04758	Ribosomal protein S30
PF01777	Ribosomal L27e protein family
PF03297	S25 Ribosomal protein
PF01247	Ribosomal protein L35Ae
PF01158	Ribosomal protein L36e
PF08079	Ribosomal L30 Nterminal domain
PF01776	Ribosomal L22e protein family
PF01159	Ribosomal protein L6e
PF01781	Ribosomal L38e protein family
PF01778	Ribosomal L28e protein family
PF01249	Ribosomal protein S21e
PF01779	Ribosomal L29e protein family

Table B.1: **List of Pfam ribosomal proteins**.

Pfam Accession Number	**Description**
PF03939	Ribosomal protein L23, Nterminal domain
PF05635	S23 Ribosomal protein
PF08561	Ribosomal protein L37
PF10501	Ribosomal subunit 39S
PF05162	Ribosomal protein L41
PF03868	Ribosomal protein L6, Nterminal domain
PF11993	Ribosomal S4
PF08136	30S Ribosomal protein subunit S22 family

Appendix C

Domains and architectures over-represented in *P. falciparum*

Table C.1 shows domains over-represented, that is, predicted at least 3 times. We also show the number of predictions for CASH' competitors. Table C.2 show the most frequent architectures (domain arragenment) found by CASH and its competitors. we report only the architectures that were predicted for at least 3 proteins.

Table C.1: **List of domains frequently found in the *P. falciparum* genome**.

Pfam Domain Name	Domain Description	Pfam24	CODD	dPuc	CASH
Ag332	Erythrocyte membrane-associated giant protein antigen 332	29	151	153	167
PFEMP	PFEMP DBL domain	114	116	114	150
RRM_1	RNA recognition motif. (a.k.a. RRM, RBD, or RNP domain)	92	107	102	119
VAR1	Mitochondrial ribosomal protein (VAR1)	0	0	0	116
LRR_1	Leucine Rich Repeat	11	44	35	100
DnaJ	DnaJ domain	50	49	47	53
AP2	AP2 domain	37	41	41	51
SART-1	SART-1 family	1	1	1	50
TPR_2	Tetratricopeptide repeat	2	37	12	48
s48_45	Sexual stage antigen s48/45 domain	29	24	29	48
AAA	ATPase family associated with various cellular activities (AAA)	42	42	41	47
HEAT	HEAT repeat	3	29	20	46
Kelch_2	Kelch motif	4	8	3	44
P_fal_TIGR01639	Protein of unknown function (P_fal_TIGR01639)	38	32	26	42
RCC1	Regulator of chromosome condensation (RCC1) repeat	14	12	16	42
SMC_N	RecF/RecN/SMC N terminal domain	5	7	7	37
IMCp	Inner membrane complex protein	9	0	0	32
Collagen	Collagen triple helix repeat (20 copies)	1	0	0	28
BicD	Microtubule-associated protein Bicaudal-D	0	0	0	24
Merozoite_SPAM	Merozoite surface protein (SPAM)	6	7	5	23
RAP	RAP domain	16	18	15	22
PPR	PPR repeat	2	12	12	21
E1-E2_ATPase	E1-E2 ATPase	16	12	13	19
DUF1777	Protein of unknown function (DUF1777)	1	5	1	19
Sel1	Sel1 repeat	16	15	13	19
S-antigen	S-antigen protein	3	4	4	18
Hydrolase_4	Putative lysophospholipase	10	0	0	18

Table C.1: **List of domains frequently found in the *P. falciparum* genome.**

Pfam Domain Name	Domain Description	Pfam24	CODD	dPuc	CASH
Borrelia_P83	Borrelia P83/100 protein	0	0	0	17
PPAK	PPAK motif	0	0	0	16
SPARC_Ca_bdg	Secreted protein acidic and rich in cysteine Ca binding region	1	0	0	16
Cornifin	Cornifin (SPRR) family	0	0	0	15
TSP_1	Thrombospondin type 1 domain	9	9	10	14
Lactamase_B	Metallo-beta-lactamase superfamily	6	6	6	14
Abhydrolase_1	alpha/beta hydrolase fold	6	9	8	13
Fibrinogen_BP	Fibrinogen binding protein	1	0	0	13
PT	PT repeat	0	1	5	12
PCI	PCI domain	9	8	8	12
eIF2A	Eukaryotic translation initiation factor eIF2A	3	6	3	11
PH	PH domain	3	8	5	11
Viral_helicase1	Viral (Superfamily 1) RNA helicase	2	0	0	11
EamA	EamA-like transporter family	4	4	5	11
DUF1565	Protein of unknown function (DUF1565)	0	0	0	11
Suf	Suppressor of forked protein (Suf)	1	1	0	11
Macoilin	Transmembrane protein	0	0	0	10
SET	SET domain	6	7	6	10
ResIII	Type III restriction enzyme, res subunit	2	4	3	10
PduV-EutP	Ethanolamine utilisation - propanediol utilisation	1	0	0	10
Transformer	Fruit fly transformer protein	0	0	0	9
CLASP_N	CLASP N terminal	1	0	0	9
FAST_1	FAST kinase-like protein, subdomain 1	0	6	0	9
Vps4_C	Vps4 C terminal oligomerisation domain	1	4	3	9
TryThrA_C	Tryptophan-Threonine-rich plasmodium antigen C terminal	4	0	0	9
IstB	IstB-like ATP binding protein	0	0	0	9
DMP1	Dentin matrix protein 1 (DMP1)	0	0	0	9
Hydrolase_3	haloacid dehalogenase-like hydrolase	3	5	6	9
Med15	ARC105 or Med15 subunit of Mediator complex non-fungal	0	0	0	9
Poxvirus_B22R	Poxvirus B22R protein	0	0	0	9
C1_1	Phorbol esters/diacylglycerol binding domain (C1 domain)	0	5	2	8
zf-B_box	B-box zinc finger	3	5	5	8
Peptidase_S9	Prolyl oligopeptidase family	4	1	1	8
TIL	Trypsin Inhibitor like cysteine rich domain	0	0	5	8
AhpC-TSA	AhpC/TSA family	5	4	4	8
Clathrin_propel	Clathrin propeller repeat	1	4	4	8
DUF1080	Domain of Unknown Function (DUF1080)	0	2	0	8
DUF577	Family of unknown function (DUF577)	0	0	0	7
NTP_transf_2	Nucleotidyltransferase domain	2	3	3	7
Cenp-F_N	Cenp-F N-terminal domain	0	0	0	7
DNA_primase_S	Eukaryotic and archaeal DNA primase small subunit	1	1	1	7
RHD3	Root hair defective 3 GTP-binding protein (RHD3)	2	1	1	7
Filamin	Filamin/ABP280 repeat	2	4	2	7
V-SNARE_C	Snare region anchored in the vesicle membrane C-terminus	2	0	0	6
PAP_central	Poly(A) polymerase central domain	1	1	1	6
Trehalose_recp	Trehalose receptor	0	0	0	6
BRCT	BRCA1 C Terminus (BRCT) domain	2	3	2	6
BRAP2	BRCA1-associated protein 2	0	0	0	6
DUF258	Protein of unknown function, DUF258	0	0	0	6
zf-UBP	Zn-finger in ubiquitin-hydrolases and other protein	3	3	2	6
SidE	Dot/Icm substrate protein	0	0	0	6
Trypan_PARP	Procyclic acidic repetitive protein (PARP)	1	0	0	6
CPSF_A	CPSF A subunit region	2	2	2	6
GLTT	GLTT repeat (6 copies)	0	0	0	6

Table C.1: **List of domains frequently found in the *P. falciparum* genome.**

Pfam Domain Name	Domain Description	Pfam24	CODD	dPuc	CASH
BK_channel_a	Calcium-activated BK potassium channel alpha subunit	1	3	3	6
BNR	BNR/Asp-box repeat	0	0	0	6
Taxilin	Myosin-like coiled-coil protein	0	0	0	5
RPAP2_Rtr1	Rtr1/RPAP2 family	1	1	1	5
Sfi1	Sfi1 spindle body protein	0	0	0	5
Ricin_B_lectin	Ricin-type beta-trefoil lectin domain	1	1	2	5
Nucleoside_tran	Nucleoside transporter	1	1	1	5
FeoB_N	Ferrous iron transport protein B	0	0	0	5
Rad51	Rad51	2	2	2	5
DUF827	Plant protein of unknown function (DUF827)	0	0	0	5
DUF3447	Domain of unknown function (DUF3447)	0	0	0	5
LRR_2	Leucine Rich Repeat	0	2	0	5
Arch_ATPase	Archaeal ATPase	1	0	0	5
CfAFP	Choristoneura fumiferana antifreeze protein (CfAFP)	0	0	0	5
Daxx	Daxx Family	1	0	0	4
Cenp-F_leu_zip	Leucine-rich repeats of kinetochore protein Cenp-F/LEK1	0	0	0	4
Fmp27_WPPW	RNA pol II promoter Fmp27 protein domain	0	0	0	4
SpoIIE	Stage II sporulation protein E (SpoIIE)	1	0	1	4
Abhydrolase_4	TAP-like protein	0	0	0	4
zf-TRAF	TRAF-type zinc finger	0	0	0	4
PRP21_like_P	Pre-mRNA splicing factor PRP21 like protein	1	0	0	4
Vicilin_N	Vicilin N terminal region	0	0	0	4
Apis_Csd	Complementary sex determiner protein	0	0	0	4
DUF445	Protein of unknown function (DUF445)	1	0	0	4
Prominin	Prominin	0	0	0	4
DUF1664	Protein of unknown function (DUF1664)	0	0	0	4
zf-C3H1	Putative zinc-finger domain	0	0	0	4
Ion_trans	Ion transport protein	0	0	0	4
SCP-1	Synaptonemal complex protein 1 (SCP-1)	0	0	0	4
EGF_2	EGF-like domain	0	0	0	4
STOP	STOP protein	0	0	0	4
DUF1162	Protein of unknown function (DUF1162)	0	0	0	3
ATG_C	ATG C terminal domain	0	0	0	3
GRP	Glycine rich protein family	0	0	0	3
MAP7	MAP7 (E-MAP-115) family	0	0	0	3
DBP	Duffy-antigen binding protein	0	0	0	3
Cse1	Cse1	0	0	0	3
GSPII_E	Type II/IV secretion system protein	0	0	0	3
Gp58	gp58-like protein	0	0	0	3
Cyclin_C	Cyclin, C-terminal domain	0	0	0	3
Pentapeptide_2	Pentapeptide repeats (8 copies)	0	0	0	3
SDA1	SDA1	0	0	0	3
TCO89	TORC1 subunit TCO89	0	0	0	3
DUF2220	Uncharacterized protein conserved in bacteria C-term(DUF2220)	0	0	0	3
DNA_pol3_delta	DNA polymerase III, delta subunit	0	0	0	3
DUF2353	Uncharacterized coiled-coil protein (DUF2353)	0	0	0	3
SusD	SusD family	0	0	0	3
TrkA_N	TrkA-N domain	0	0	0	3
YHS	YHS domain	0	0	0	3
ITAM	Immunoreceptor tyrosine-based activation motif	0	0	0	3
RNase_Zc3h12a	Zc3h12a-like Ribonuclease domain	0	0	0	3
Cys_rich_FGFR	Cysteine rich repeat	0	0	0	3

Table C.2: **List of the most frequent domain architecture found in the *P. falciparum* genome.**

Domain Architecture	Pfam24	CODD	dPuc	CASH
VAR1-VAR1	0	0	0	42
PFEMP-PFEMP	30	30	30	33
PFEMP-PFEMP-PFEMP-Duffy_binding-Duffy_binding	0	0	0	20
VAR1-VAR1-VAR1	0	0	0	15
HEAT-HEAT	0	7	6	14
LRR_1-LRR_1-LRR_1	2	5	4	14
AP2-AP2	10	11	11	13
s48_45-s48_45	8	5	7	12
TPR_1-TPR_2-TPR_2	0	7	2	11
PPAK-PFEMP-PFEMP-Duffy_binding-Duffy_binding	0	0	0	10
efhand-SPARC_Ca_bdg	0	0	0	10
LRR_1-LRR_1-LRR_1-LRR_1-LRR_1	1	2	2	10
PPAK-PFEMP-PFEMP-PFEMP-Duffy_binding-Duffy_binding	0	0	0	8
IMCp-IMCp	0	0	0	8
Duffy_binding-Merozoite_SPAM	2	3	1	8
RCC1-RCC1-RCC1	2	2	2	8
RRM_1-DUF1777	0	3	0	8
RCC1-RCC1	2	2	4	8
PPAK-PFEMP-Duffy_binding-Duffy_binding	0	0	0	8
SART-1-SART-1	0	0	0	7
LRR_1-LRR_1-LRR_1-LRR_1-LRR_1-LRR_1	0	1	1	7
Abhydrolase_1-Hydrolase_4	0	0	0	7
Peptidase_S9-Hydrolase_4	2	0	0	7
NTP_transf_2-PAP_central	0	0	1	6
MMR_HSR1-PduV-EutP	1	0	0	6
RRM_1-Transformer	0	0	0	6
AP2-AP2-AP2	3	4	4	6
LRR_1-LRR_1	0	4	4	6
SMC_N-SMC_N	1	0	0	6
Ycf1-Ycf1-Ycf1	0	0	0	5
PFEMP-PFEMP-PFEMP-PFEMP-Duffy_binding	0	0	0	5
efhand-efhand-SPARC_Ca_bdg	0	0	0	5
Adap_comp_sub-Clat_adaptor_s	2	3	1	5
Pkinase-Poxvirus_B22R	0	0	0	5
VAR1-VAR1-VAR1-VAR1	0	0	0	5
LRR_1-LRR_1-LRR_1-LRR_1	0	1	1	5
zf-B_box-zf-B_box	1	1	2	4
TPR_2-TPR_2-TPR_2	0	1	0	4
Helicase_C-ResIII	1	1	1	4
efhand-efhand-Pkinase-SPARC_Ca_bdg	0	0	0	4
Kelch_1-Kelch_2-Kelch_2-Kelch_2-Kelch_2	0	0	0	4
Kelch_2-Kelch_2-Kelch_2-Kelch_2	0	0	0	4
RCC1-RCC1-RCC1-RCC1	2	2	1	4
VAR1-VAR1-VAR1-VAR1-VAR1	0	0	0	4
DUF1080-DUF1080	0	0	0	4
s48_45-s48_45-s48_45	2	2	2	4
DEAD-Helicase_C-SART-1	0	0	0	4
zf-CCCH-zf-C3H1	0	0	0	4
Kelch_2-Kelch_2	0	1	1	4
S-antigen-S-antigen	1	0	0	3
Duffy_binding-Duffy_binding-EBA-175_VI-DBP	0	0	0	3
Cyclin_N-Cyclin_C	0	0	0	3
SidE-SidE	0	0	0	3
BicD-BicD-BicD	0	0	0	3
AAA-AAA-CDC48_N-CDC48_2-Vps4_C	0	1	1	3
E1-E2_ATPase-E1-E2_ATPase-Hydrolase-Hydrolase_3	0	0	0	3
PFEMP-PFEMP-PFEMP-Duffy_binding-Duffy_binding-Duffy_binding-Duffy_binding	0	0	0	3
RHD3-RHD3	1	0	0	3
WD40-CPSF_A	0	0	0	3
TP6A_N-DUF2220	0	0	0	3
Filamin-Filamin	0	0	1	3
ABC_tran-ABC_tran-DUF258	0	0	0	3

Table C.2: **List of the most frequent domain architecture found in the *P. falciparum* genome**.

Domain Architecture	**Pfam24**	**CODD**	**dPuc**	**CASH**
ABC_tran-DUF258	0	0	0	3
Pkinase-Collagen	0	0	0	3
AAA-IstB	0	0	0	3
Abhydrolase_4-Hydrolase_4	0	0	0	3

Bibliography

[1] A. Dennis, D. Lipman, J. Ostell, and D. Wheeler, "Genbank," *Nucleic Acids Research*, vol. 33, no. 1, pp. 34–38, 2004.

[2] B. Boeckmann, A. Bairoch, R. Apweiler, M. Blatter, A. Estreicher, E. Gasteiger, M. Martin, K. Michoud, C. O'Donovan, and I. Phan, "The swiss-prot protein knowledgebase and its supplement trembl in 2003," *Nucleic Acids Research*, vol. 31, no. 1, pp. 365–370, 2003.

[3] A. Bairoch, B. Boeckmann, S. Ferro, and E. Gasteiger, "Swiss-prot: juggling between evolution and stability," *Briefings in Bioinformatics*, vol. 5, no. 1, pp. 39–55, 2005.

[4] M. Helen, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. Shindyalov, and P. Bourne, "The protein data bank," *Nucleic Acids Research*, vol. 28, pp. 235–242, 2000.

[5] G. Pandey, V. Kumar, and M. Steinbach, "Computational approaches for protein function prediction: A survey," Tech. Rep. 06-028, Department of Computer Science and Engineering, University of Minnesota, Twin Cities, 2006.

[6] J. Bernardes, J. Fernandez, and A. Vasconcelos, "Structural descriptor database: a new tool for sequence-based functional site prediction," *BMC Bioinformatics*, vol. 9, p. 492, 2008.

[7] C. Pierri, G. Parisi, and V. Porcelli, "Computational approaches for protein function prediction: A combined strategy from multiple sequence alignment to molecular docking-based virtual screening," *Biochimica et Biophysica Acta (BBA) - Proteins & Proteomics*, vol. 1804, pp. 1695–1712, 2010.

[8] A. Andreeva, D. Howorth, S. Brenner, T. Hubbard, C. Chothia, and A. Murzin, "Scop database in 2004: refinements integrate structure and sequence family data," *Nucleic Acids Research*, vol. 32, pp. 226–229, 2004.

[9] S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman, "Basic local alignment search tool," *Journal Molecular Biology*, vol. 215, pp. 403–410, 1990.

[10] W. Pearson, "Rapid and sensitive sequence comparison with fastp and fasta," *Methods Enzymol*, vol. 183, pp. 63–98, 1985.

[11] R. Hughey and A. Krogh, "Hidden markov models for sequence analysis: extension and analysis of the basic method," *CABIOS*, vol. 12, pp. 95–107, 1996.

[12] S. Altschul, T. Madden, A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. Lipman, "Gapped blast and psi-blast: A new generation of protein database search programs," *Nucleic Acids Research*, vol. 25, pp. 3389–3402, 1997.

[13] J. Gough, K. Karplus, R. Hughey, and C. Chothia, "Assignment of homology to genome sequences using a library of hidden markov models that represent all proteins ok known structure," *Journal of Molecular Biology*, vol. 313, pp. 903–919, 2001.

[14] G. Yona and M. Levitt, "Within the twilight zone: a sensitive profile-profile comparison tool based on information theory," *Journal of molecular biology*, vol. 315, pp. 1257–1275, 2002.

[15] R. Sadreyev, D. Baker, and N. Grishin, "Profile-profile comparisons by compass predict intricate homologies between protein families," *Protein Science*, vol. 12, pp. 2262–2272, 2003.

[16] J. Soeding, "Protein homology detection by hmm-hmm comparison," *Bioinformatics*, vol. 21, pp. 951–960, 2005.

[17] B. Qian and R. Goldstein, "Performance of an iterated t-hmm for homology detection," *Bioinformatics*, vol. 20, pp. 2175–2180, 2004.

[18] V. Alexandrov and M. Gerstein, "Using 3d hidden markov models that explicitly represent spatial coordinates to model and compare protein structures," *BMC Bioinformatics*, vol. 5, pp. 1–10, 2004.

[19] J. Bernardes, A. Davila, V. Costa, and G. Zaverucha, "Improving model construction of profile hmms for remote homology detection through structural alignment," *BMC Bioinformatics*, vol. 435, pp. 1–12, 2007.

[20] C. Vogel, C. Berzuini, M. Bashton, J. Gough, and S. Teichmann, "Supra-domains: evolutionary units larger than single protein domains," *Journal of Molecular Biology*, vol. 336, pp. 809–823, 2004.

[21] W. McLaughlin, K. Chen, T. Hou, and W. Wang, "On the detection of functionally coherent groups of protein domains with an extension to protein annotation," *BMC Bioinformatics*, vol. 8, p. 390, 2007.

[22] M. Scott, D. Thomas, and M. Hallett, "Predicting subcellular localization via protein motif co-occurrence," *Genome Research*, vol. 14, pp. 1957–1966, 2004.

[23] L. Geer, M. Domrachev, D. Lipman, and S. Bryant, "Cdart: Protein homology by domain architecture," *Genome Research*, vol. 12, pp. 1619–1623, 2002.

[24] N. Terrapon, O. Gascuel, E. Marechal, and L. Breehelin, "Detection of new protein domains using co-occurrence: application to plasmodium falciparum," *Bioinformatics*, vol. 25, pp. 3077–3083, 2009.

[25] A. Ochoa, M. Llinas, and M. Singh, "Using context to improve protein domain identification," *BMC Bioinformatics*, vol. 12, p. 90, 2011.

[26] T. Jaakkola, M. Diekhans, and D. Haussler, "A discriminative framework for detecting remote protein homologies," *Journal of Computational Biology*, vol. 7, pp. 95–114, 2000.

[27] A. Ben-Hur and D. Brutlag, "Remote homology detection: a motif based approach," *BMC Bioinformatics*, vol. 19, pp. i26–i33, 2003.

[28] Y. Hou, W. Hsu, M. Lee, and C. Bystroff, "Efficient remote homology detection using local structure," *Bioinformatics*, vol. 17, pp. 2294–2301, 2003.

[29] C. Leslie, E. Eskin, A. Cohen, J. Weston, and W. Noble, "Mismatch string kernels for discriminative protein classification," *Bioinformatics*, vol. 20, pp. 467–476, 2004.

[30] L. Liao and W. Noble, "Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships," *Journal of Computational Biology*, vol. 10, pp. 857–868, 2004.

[31] Y. Hou, W. Hsu, L. Lee, and C. Bystroff, "Remote homolog detection using local sequence-structure correlations," *Proteins*, vol. 57, pp. 518–530, 2004.

[32] H. Saigo, J. Vert, N. Ueda, and T. Akutsu, "Protein homology detection using string alignment kernels," *Bioinformatics*, vol. 20, pp. 1682–1689, 2004.

[33] Q. Su, L. Lu, S. Saxonov, and D. Brutlag, "eblocks: enumerating conserved protein blocks to achieve maximal sensitivity and specificity," *Nucleic acids research*, vol. 33, pp. D178–D182, 2005.

[34] V. Atalay and R. Cetin-Atalay, "Implicit motif distribution based hybrid computational kernel for sequence classification," *Bioinformatics*, vol. 21, pp. 1429–1436, 2005.

[35] R. Kuang, E. Ie, K. Wang, K. Wang, M. Siddiqi, Y. Freund, and C. Leslie, "Profile-based string kernels for remote homology detection and motif extraction," *Journal of bioinformatics and computational biology*, vol. 3, pp. 527–550, 2005.

[36] H. Rangwala and G. Karypis, "Profile-based direct kernels for remote homology detection and fold recognition," *Bioinformatics*, vol. 21, pp. 4239–4247, 2005.

[37] T. Lingner and P. Meinicke, "Remote homology detection based on oligomer distances," *Bioinformatics*, vol. 22, pp. 2224–2231, 2006.

[38] Q. Dong, X. Wang, and L. Lin, "Application of latent semantic analysis to protein remote homology detection," *Bioinformatics*, vol. 22, pp. 285–290, 2006.

[39] T. Handstad, A. Hestnes, and P. Saetrom, "Motif kernel generated by genetic programming improves remote homology and fold detection," *BMC Bioinformatics*, vol. 8, p. 23, 2007.

[40] B. Liu, X. Wang, L. Lin, Q. Dong, and X. Wang, "A discriminative method for protein remote homology detection and fold recognition combining top-n-grams and latent semantic analysis," *BMC Bioinformatics*, vol. 9, p. 510, 2008.

[41] A. Shah, C. Oehmen, and B. Webb-Robertson, "Svm-hustle - an iterative semi-supervised machine learning approach for pairwise protein remote homology detection," *Bioinformatics*, vol. 24, pp. 783–790, 2008.

[42] B. Webb-Robertson, K. Ratuiste, and C. Oehmen, "Physicochemical property distributions for accurate and rapid pairwise protein homology detection," *BMC Bioinformatics*, vol. 11, p. 145, 2010.

[43] S. Muggleton and L. De RAEDT, "Inductive logic programming: Theory and methods," *Journal of Logic Programming*, vol. 19/20, pp. 629–679, 1994.

[44] A. Karwath and R. King, "Homology induction: the use of machine learning to improve sequence similarity searches," *BMC Bioinformatics*, vol. 3, p. 11, 2002.

[45] A. Karwath and R. King, "An automated ilp server in the field of bioinformatics," in *Proceedings of the Eleventh International Conference on Inductive Logic Programming. Lecture Notes in Artificial Intelligence 2157. Heidelberg: Springer-Verlag*, pp. 91–103, 2001.

[46] R. King, "Applying inductive logic programming to predicting gene function," *AI Magazine*, vol. 25, pp. 57–58, 2004.

[47] R. King, A. Srinivasan, and L. Dehaspe, "A data-mining tool for chemical data," *Journal of Computer-Aided Molecular Design*, vol. 15, pp. 173–181, 2001.

[48] J. Bernardes, A. Carbone, and G. Zaverucha, "A discriminative method for family-based protein remote homology detection that combines inductive logic programming and propositional models," *BMC Bioinformatics*, vol. 12, p. 83, 2011.

[49] J. Bernardes, G. Zaverucha, C. Vaquero, and A. Carbone, "Combining evolution and machine learning for functional annotation in plasmodium falciparum annotation," *Genome Research (in preparation)*, 2012.

[50] R. Finn, J. Mistry, J. Tate, P. Coggill, A. Heger, J. Pollington, L. Gavin, P. Gunasekaran, G. Ceric, K. Forslund, L. Holm, E. Sonnhammer, S. Eddy, and A. Bateman, "The pfam protein families database," *Nucleic Acids Research*, vol. 38, pp. D211–D222, 2010.

[51] I. Callebaut, K. Prat, E. Meurice, J.-P. Mornon, and S. Tomavo, "Prediction of the general transcription factors associated with rna polymerase ii in plasmodium falciparum: conserved features and differences relative to other eukaryotes," *BMC Genomics*, vol. 6, p. 100, 2005.

[52] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta, *Metalearning: Applications to Data Mining*. Cognitive technologies, Springer, 2009.

[53] K. Smith-Miles, "Cross-disciplinary perspectives on meta-learning for algorithm selection," *ACM Comput. Surv.*, vol. 41, pp. 1–25, 2009.

[54] G. Cooper and R. Hausman in *The Cell: A Molecular Approach*, ch. 7: RNA Synthesis and Processing, Sinauer Associates, 2009.

[55] G. Cooper and R. Hausman in *The Cell: A Molecular Approach*, ch. 8: Protein Synthesis, Processing, and Regulation, Sinauer Associates, 2009.

[56] L. Pauling, R. Corey, and H. Branson, "The structure of proteins: Two hydrogen-bonded helical configurations of the polypeptide chain," *PNAS*, vol. 37, pp. 205–211, 1951.

[57] S. Stephens, "Possible significance of duplication in evolution," vol. 4 of *Advances in Genetics*, pp. 247–265, Academic Press, 1951.

[58] S. Wright, "The roles of mutation, inbreeding, crossbreeding and selection in evolution," in *Proceedings of the VI International Congress of Genetrics*, vol. 1, pp. 356–366, 1932.

[59] M. White, "Models of speciation," *Science*, vol. 159, pp. 1065–1070, 1968.

[60] E. Koonin and M. Galperin in *Sequence - Evolution - Function Computational Approaches in Comparative Genomics*, ch. 2: Evolutionary Concept in Genetics and Genomics, Kluwer Academic, 2009.

[61] C. Gaboriaud, V. Bissery, T. Benchetrit, and J. Mornon, "Hydrophobic cluster analysis: An efficient new way to compare and analyse amino acid sequences," *FEBS Letters*, vol. 224, no. 1, pp. 149–155, 1987.

[62] B. Hall, *Phylogenetic trees made easy: a how-to manual*. Sinauer Associates, 2004.

[63] C. Wu, A. Nikolskaya, H. Huang, L. Yeh, D. Natale, C. Vinayaka, Z. Hu, R. Mazumder, S. Kumar, P. Kourtesis, R. Ledley, B. Suzek, L. Arminski, Y. Chen, J. Zhang, J. Cardenas, S. Chung, A. Castro, G. Dinkov, and W. Barker, "Pirsf: family classification system at the protein information resource," *Nucleic Acids Research*, vol. 32, pp. D112–D114, 2004.

[64] G. Yona, N. Linial, and M. Linial, "Protomap: automatic classification of protein sequences and hierarchy of protein families," *Nucleic Acids Research*, vol. 28, pp. 49–55, 2000.

[65] D. Haft, D. Selengut, and O. White, "The tigrfams database of protein families," *Nucleic Acids Research*, vol. 31, pp. 371–373, 2003.

[66] F. Corpet, F. Servant, J. Gouzy, and D. Kahn, "Prodom and prodom-cg: tools for protein domain analysis and whole genome comparisons," *Nucleic Acids Research*, vol. 28, pp. 267–269, 2000.

[67] C. Sigrist, L. Cerutti, E. Castro, P. Langendijk-Genevaux, V. Bulliard, A. Bairoch, and N. Hulo, "Prosite, a protein domain database for functional characterization and annotation," *Nucleic Acids Research*, vol. 38, pp. D161–D166, 2010.

[68] T. Attwood, M. Beck, A. Bleasby, K. Degtyarenko, and D. P. Smith, "Progress with the prints protein fingerprint database," *Nucleic Acids Research*, vol. 24, pp. 182–188, 1996.

[69] F. Pearl, C. Bennett, J. Bray, A. Harrison, N. Martin, A. Shepherd, I. Sillitoe, J. Thornton, and C. Orengo, "The cath database: an extended protein family resource for structural and functional genomics," *Nucleic Acids Research*, vol. 31, pp. 452–455, 2003.

[70] C. Wu, C. Xiao, Z. Hou, H. Huang, and W. Barker, "iproclass: an integrated, comprehensive and annotated protein classification database," *Nucleic Acids Research*, vol. 29, pp. 52–54, 2001.

[71] R. Apweiler, T. Attwood, A. Bairoch, A. Bateman, E. Birney, M. Biswas, P. Bucher, L. Cerutti, F. Corpet, M. Croning, R. Durbin, L. Falquet, W. Fleischmann, J. Gouzy, H. Hermjakob, N. Hulo, I. Jonassen, D. Kahn, A. Kanapin, Y. Karavidopoulou, R. Lopez, B. Marx, N. Mulder, T. Oinn, M. Pagni, F. Servant, C. Sigrist, and E. Zdobnov, "Interpro-an integrated documentation resource for protein families, domains and functional sites," *Bioinformatics*, vol. 16, no. 12, pp. 1145–1150, 2000.

[72] S. Henikoff and J. Henikoff, "Amino acid substitution matrices from protein blocks.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 89, pp. 10915–10919, 1992.

[73] T. Smith and M. Waterman, "Identification of common molecular subsequences," *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195–197, 1981.

[74] E. Gumbel, *Statistics of extremes*. Columbia University Press, 1958.

[75] M. Gribskov, A. McLachlan, and D. Eisenberg, "Profile analysis: detection of distantly related proteins," *Proceedings of the National Academy of Sciences*, vol. 84, pp. 4355–4358, 1987.

[76] D. Bashford, C. Chothia, and A. Lesk, "Determinants of a protein fold: Unique features of the globin amino acid sequences," *Journal of Molecular Biology*, vol. 196, pp. 199–216, 1987.

[77] R. Hughey and A. Krogh, "Hidden markov models for sequence analysis: extension and analysis og the basic method," *Applications in the Biosciences*, vol. 12, pp. 95–107, 1996.

[78] S. Eddy, "Profile hidden markov models.," *Bioinformatics*, vol. 14, pp. 755–763, 1998.

[79] M. Brown, R. Hughey, A. Krogh, I. Mian, K. Sjlander, and D. Haussler, "Using dirichlet mixture priors to derive hidden markov models for protein families," in *Proc.of First Int. Conf. on Intelligent Systems for Molecular Biology*, vol. 1, p. 4755, 1993.

[80] J. Thompson, D. Higgins, and T. Gibson, "Improved sensitivity of profile searches through the use of sequence weights and gap excision," *Computer applications in the biosciences : CABIOS*, vol. 10, no. 1, pp. 19–29, 1994.

[81] A. Krogh and G. Mitchison, "Maximum entropy weighting of aligned sequences of proteins or dna," in *In Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, pp. 215–221, AAAI Press, 1995.

[82] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Proceedings of the IEEE*, pp. 257–286, 1989.

[83] A. Rolf, A. Bairoch, C. Wu, W. Barker, B. Boeckmann, S. Ferro, E. Gasteiger, H. Huang, R. Lopez, M. Magrane, M. Martin, D. Natale, C. O'Donovan, N. Redaschi, and L. Yeh, "Uniprot: the universal protein knowledgebase," *Nucleic Acids Research*, vol. 32, pp. D115–D119, 2004.

[84] B. Scholkopf, C. Burges, and A. Smola, "Advances in kernel methods: support vector learning," in *MIT Press*, 1999.

[85] J. Weston, A. Elisseeff, D. Zhou, C. Leslie, and W. Noble, "Protein ranking: from local to global structure in the protein similarity network," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, pp. 6559–6563, 2004.

[86] L. Dehaspe and L. D. Raedt, "Mining association rules in multiple relations," in *Proceedings of the 7th International Workshop on Inductive Logic Programming*, vol. 1297, pp. 125–132, Springer-Verlag, 1997.

[87] J. Quinlan, "C4.5: Programs for machine learning," *Machine Learning*, vol. 16, pp. 235–240, 1994.

[88] U. Syed and G. Yona, "Using a mixture of probabilistic decision trees for direct prediction of protein function," *Annual Conference on Research in Computational Molecular Biology*, vol. 28, pp. 289–300, 2003.

[89] C. Ferreira, J. Gama, and V. Costa, "Ruse-warmr: Rule selection for classifier induction in multi-relational data-sets," in *Proceedings of the 20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2008)*, vol. 1, pp. 379–386, IEEE Computer Society, 2008.

[90] S. Brenner, P. Koehl, and M. Levitt, "The astral compendium for sequence and structure analysis," *Nucleic Acids Research*, vol. 28, pp. 254–256, 2000.

[91] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *23rd International Conference on Machine Learning (ICML)*, pp. 233–240, 2006.

[92] N. Shan-Hwei and R. Wolf, *Foundations of Inductive Logic Programming*, vol. 1228. Springer, 1997.

[93] L. De Raedt, *Logical and Relational Learning*. Springer, 2008.

[94] R. Agrawal, T. Imielinski, and R. Srikant, "Association rules between sets of items in large databases," in *Proceedings of the ACM SIGMOD Intl. Conf. on Management of Data, Washington, Estados Unidos*, pp. 207–216, 1993.

[95] D. Higgins, J. Thompson, T. Gibson, J. Thompson, D. Higgins, and T. Gibson, "Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting,position-specific gap penalties and weight matrix choice," *Nucleic Acids Research*, vol. 22, pp. 4673–4680, 1994.

[96] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, pp. 80–83, 1945.

[97] S. Lee and L. De Raedt, "Constraint based mining of first order sequences in seqlog," in *Database Support for Data Mining Application*, pp. 155–176, Springer, 2004.

[98] A. Bahl, B. Brunk, J. Crabtree, M. Fraunholz, B. Gajria, G. Grant, H. Ginsburg, D. Gupta, J. Kissinger, P. Labo, L. Li, M. Mailman, A. Milgram, D. Pearson, D. Roos, J. Schug, C. Stoeckert, and P. Whetzel, "Plasmodb: the plasmodium genome resource. A database integrating experimental and computational data," *Nucleic Acids Research*, vol. 31, pp. 212–215, 2003.

[99] C. Aurrecoechea, J. Brestelli, B. Brunk, J. Dommer, S. Fischer, B. Gajria, X. Gao, A. Gingle, G. Grant, O. Harb, M. Heiges, F. Innamorato, J. Iodice, J. Kissinger, E. Kraemer, W. Li, J. Miller, V. Nayak, C. Pennington, D. Pinney, D. Roos, C. Ross, C. Stoeckert, C. Treatman, and H. Wang, "Plasmodb: a functional genomic database for malaria parasites," *Nucleic Acids Research*, vol. 37, pp. D539–D543, 2009.

[100] S. Date and C. Stoeckert, "Computational modeling of the plasmodium falciparum interactome reveals protein function on a genome-wide scale," *Genome Research*, vol. 16, pp. 542–549, 2006.

[101] F. Lu, H. Jiang, J. Ding, J. Mu, J. Valenzuela, J. Ribeiro, and X. zhuan Su, "cdna sequences reveal considerable gene prediction inaccuracy in the plasmodium falciparum genome," *BMC Genomics*, vol. 8, p. 255, 2007.

[102] Y. Joubert and F. Joubert, "A structural annotation resource for the selection of putative target proteins in the malaria parasite," *Malaria Journal*, vol. 7, p. 90, 2008.

[103] I. Letunic, R. Copley, B. Pils, S. Pinkert, J. Schultz, and P. Bork, "Smart 5: domains in the context of genomes and networks," *Nucleic Acids Research*, vol. 34, pp. D257–D260, 2005.

[104] C. Yeats, M. Maibaum, R. Marsden, M. Dibley, D. Lee, S. Addou, and C. Orengo, "Gene3d: modelling protein structure, function and evolution," *Nucleic Acids Research*, vol. 34, pp. D281–D284, 2005.

[105] H. Mi, B. Lazareva-Ulitsky, R. Loo, A. Kejariwal, J. Vandergriff, S. Rabkin, N. Guo, A. Muruganujan, O. Doremieux, M. Campbell, H. Kitano, and P. Thomas, "The panther database of protein families, subfamilies, functions and pathways," *Nucleic Acids Research*, vol. 33, pp. D284–D288, 2005.

[106] C. Yeats, O. C. Redfern, and C. Orengo, "A fast and automated solution for accurately resolving protein domain architectures," *Bioinformatics*, vol. 26, pp. 745–751, 2010.

[107] E. Bischoff and C. Vaquero, "In silico and biological survey of transcription-associated proteins implicated in the transcriptional machinery during the erythrocytic development of plasmodium falciparum," *BMC Genomics*, vol. 11, p. 34, 2010.

[108] T. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems*, vol. 1857, pp. 1–15, Springer Berlin/Heidelberg, 2000.

[109] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[110] Y. Freund and R. Schapire, "Experiments with a new boosting algorithm," in *International Conference on Machine Learning*, pp. 148–156, 1996.

[111] D. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, pp. 241–259, 1992.

[112] S. Dzeroski and B. Zenko, "Is combining classifiers with stacking better than selecting the best one?," *Machine Learning*, vol. 54, pp. 255–273, 2004.

[113] L. Lam and S. Suen, "Application of majority voting to pattern recognition: an analysis of its behavior and performance," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 27, pp. 553–568, 1997.

[114] J. Platt, N. Cristianini, and J. Shawe-taylor, "Large margin dags for multiclass classification," in *Advances in Neural Information Processing Systems 12*, pp. 547–553, 2000.

[115] J. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*, pp. 61–74, MIT Press, 1999.

[116] A. Anand, G. Pugalenthi, and P. Suganthan, "Predicting protein structural class by svm with class-wise optimized features and decision probabilities," *Journal of Theoretical Biology*, vol. 253, pp. 375–380, 2008.

[117] R. Marler and J. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, pp. 369–395, 2004.

[118] F. Waltz, "An engineering approach: hierarchical optimization criteria," *IEEE Trans. Autom. Control*, vol. 12, pp. 179–180, 1967.

[119] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 1–27, May 2011.

[120] B. Baum, "Combining trees as a way of combining data sets for phylogenetic inference, and the desirability of combining gene trees," *Taxon*, vol. 41, pp. 3–10, 1992.

[121] N. Saitou and M. Nei, "The neighbor-joining method: a new method for reconstructing phylogenetic trees.," *Molecular Biology and Evolution*, vol. 4, pp. 406–425, 1987.

[122] R. Page, "Modified mincut supertrees," in *Proceedings of the Second International Workshop on Algorithms in Bioinformatics*, pp. 537–552, Springer-Verlag, 2002.

[123] E. Muller and B. Wittmann-Liebold, "Phylogenetic relationship of organisms obtained by ribosomal protein comparison," *Cellular and Molecular Life Sciences*, vol. 53, pp. 34–50, 1997.

[124] P. Keeling, G. Burger, D. Durnford, B. Lang, R. Lee, R. Pearlman, A. Roger, and M. Gray, "The tree of eukaryotes," *Trends in Ecology & Evolution*, vol. 20, pp. 670–676, 2005.

[125] O. Lichtarge, H. Bourne, and F. Cohen, "An evolutionary trace method defines binding surfaces common to protein families," *Journal of Molecular Biology*, vol. 257, pp. 342–358, 1996.

[126] S. Lockless and R. Ranganathan, "Evolutionarily conserved pathways of energetic connectivity in protein families," *Science*, vol. 286, pp. 295–299, 1999.

[127] G. Suel, S. Lockless, M. Wall, and R. Ranganathan, "Evolutionarily conserved networks of residues mediate allosteric communication in proteins," *Nat Struct Mol Biol*, vol. 10, pp. 1072–8368, 2003.

[128] J. Baussand and A. Carbone, "A combinatorial approach to detect coevolved amino acid networks in protein families of variable divergence," *PLoS Comput Biol*, vol. 5, p. e1000488, 09 2009.

[129] A. Carbone and L. Dib, "Co-evolution and information signals in biological sequences," *Theoretical Computer Science*, vol. 412, pp. 2486–2495, 2011.

[130] K. Forslund and E. Sonnhammer, "Predicting protein function from domain content," *Bioinformatics*, vol. 24, pp. 1681–1687, 2008.

[131] E. Frank, Y. Wang, S. Inglis, G. Holmes, and I. Witten, "Using model trees for classification," *Machine Learning*, vol. 32, pp. 63–76, 1998.

[132] J. Bernardes, A. Davila, V. Costa, and G. Zaverucha, "Hmmer-struct: Adding structural properties to profile hmms." http://arxiv.org/abs/0704.2010, 2007.

[133] C. Bowler, A. Allen, J. Badger, J. Grimwood, K. Jabbari, A. Kuo, U. Maheswari, C. Martens, F. Maumus, R. Otillar, E. Rayko, A. Salamov, K. Vandepoele, B. Beszteri, A. Gruber, M. Heijde, M. Katinka, T. Mock, K. Valentin, F. Verret, J. A. Berges, C. Brownlee, J.-P. Cadoret, A. Chiovitti, C. J. Choi, S. Coesel, A. D. Martino, J. Detter, C. Durkin, A. Falciatore, J. Fournet, M. Haruta, M. Huysman, B. Jenkins, K. Jiroutova, R. Jorgensen, Y. Joubert, A. Kaplan, N. Kroger, P. Kroth, J. L. Roche, E. Lindquist, M. Lommer, V. Martin-Jezequel, P. Lopez, S. Lucas, M. Mangogna, K. McGinnis, L. Medlin, A. Montsant, M.-P. Secq, C. Napoli, M. Obornik, M. Parker, J.-L. Petit, B. Porcel, N. Poulsen, M. Robison, L. Rychlewski, T. Rynearson, J. Schmutz, H. Shapiro, M. Siaut, M. Stanley, M. Sussman, A. Taylor, A. Vardi, P. von Dassow, W. Vyverman, A. Willis, L. Wyrwicz, D. Rokhsar, J. Weissenbach, E. Armbrust, B. Green, Y. V. de Peer, and I. Grigoriev,

"The phaeodactylum genome reveals the evolutionary history of diatom genomes," *Nature*, vol. 456, pp. 239–244, 2008.