Exploiting parallel data mining processing for protein annotation

Christos N.Gkekas, Fotis E.Psomopoulos, Pericles A.Mitkas

Department of Electrical and Computer Engineering
Aristotle University of Thessaloniki
Thessaloniki, 54 124, Greece
{chggr@ee.auth.gr, fpsom@danae.ee.auth.gr, mitkas@eng.auth.gr}

Abstract

Proteins are large organic compounds consisting of amino acids arranged in a linear chain and joined together by peptide bonds. One of the most important challenges in modern Bioinformatics is the accurate prediction of the functional behavior of proteins. In this paper a novel parallel methodology for automatic protein function annotation is presented. Data mining techniques are employed in order to construct models based on data generated from already annotated protein sequences. The first step of the methodology is to obtain the motifs present in these sequences, which are then provided as input to the data mining algorithms in order to create a model for every term. Experiments conducted using the EGEE Grid environment as a source of multiple CPUs clearly indicate that the methodology is highly efficient and accurate, as the utilization of many processors substantially reduces the execution time.

Keywords: Bioinformatics, Protein Classification, Data Mining, Grid Computing, Parallel Processing, Finite State Automata

1. Introduction

Proteins are large organic compounds consisting of amino acids arranged in a linear chain and joined together by peptide bonds. The sequence of amino acids in a protein is defined by a gene and encoded in the genetic code of every living organism. Proteins participate in every process within cells and thus they are essential parts of all organisms. Some of them are enzymes that catalyze biochemical reactions, whereas others have various structural or mechanical functions. Moreover, a lot of proteins are involved in cell signaling, cell adhesion, immune responses and the cell cycle. Finally, it is also very common for proteins to work together in order to achieve a particular function, and they often associate to form stable complexes.

A group of evolutionarily and/or functionally related proteins is defined as a protein family. Currently, there is an ongoing worldwide effort towards formally organizing proteins into families and describing their component domains and motifs. Reliable identification of protein families is essential to phylogenetic analysis, functional annotation and the exploration of protein function diversity in a given phylogenetic branch. It is a fact that all proteins in the same family share strong structural similarities and therefore exhibit similar behavior. Until recently, the biological properties of proteins had to be experimentally determined by means of costly and time-consuming in vitro methods. Bioinformatics on the other hand uses computational methods to address such problems.

The algorithmic means for establishing protein families on a large scale are based on a notion of similarity. Often, the only similarity we have access to, is sequence similarity. One of the most recent tools for protein function annotation is the Gene Ontology Project. This project aims at providing a controlled vocabulary to describe gene and gene product attributes in organisms. In order to assign Gene Ontology terms to new non-annotated protein sequences, the latter have to be either processed directly in a lab or characterized through similarity to already annotated sequences. At the moment, the amino acid sequence of more than 1,000,000 proteins has been obtained. On the contrary, the properties and functions of only 4% of these proteins are known. Therefore, the need for a systematic way to gain insight on protein properties by inspecting their amino acid sequence is obvious.

In this paper a novel methodology will be presented, which uses the motifs present in already annotated protein sequences in order to create models for the annotation terms (i.e. protein families) selected by the user. The models created can then be used to predict the annotation of new protein sequences. More specifically, the motif sequence of the protein under examination is extracted and run through all available term models. This process generates similarity scores, which constitute an accurate prediction of the protein's annotation.

The main difference of our methodology compared to algorithms proposed by both the artificial intelligence community and the pattern recognition community is that it utilizes Finite State Automata to solve the problem of protein function prediction. Alternative methods include amongst others statistical models [Duad (1973)], neural networks [Bishop (1995)], or decision trees [Wang (2001), Quinlan (1992)]. Moreover, there are also some algorithms in the literature that use Finite State Automata to address the aforementioned problem [Psomopoulos (2006)]. The main difference of those algorithms to the one presented in this paper is that the later allows the creation of annotation term models and the classification of proteins to be performed in parallel. Due to its parallel nature, the algorithm can be deployed over computer clusters therefore reducing the execution time significantly.

2. Methodology Outline

The first step of the methodology is to obtain the motifs present in already annotated protein sequences. This is achieved by the use of the UNIPROT code of each protein and the InterProScan tool, taking under consideration every available sequence database (e.g. PRODOM, PROFILE, PFAM, etc). Through this process the initial protein data set is constructed, which contains for every protein its motif sequence and all the terms it has been annotated with. This data set is subsequently divided into two disjoint sets: the Training Set (DS_{train}) and the Test Set (DS_{test}).

$$DS_{train} \cap DS_{test} = \emptyset \qquad \textbf{(1)}$$

As the name implies, the Training Set is provided as input to the data mining algorithms used by the methodology in order to create a model for every annotation term. On the contrary, the Test Set is used in order to test the methodology's accuracy and effectiveness.

In the next step, the Training Set is further divided into several smaller protein data sets with each of them consisting solely of protein sequences that have been annotated with the same term. Therefore, a corresponding data set is constructed for every available term. The produced data set is part of the initial Training Set and will be later used as input to the data mining algorithms in order to derive a model for the specific term.

For each of the protein data sets created during the previous step, a Prefix Tree Acceptor (PTA) is constructed using the motif sequence of the proteins in the set. Utilizing the Alergia algorithm [Carrasco (1994)], this PTA is consequently transformed into a more generalized Stochastic Finite State Automaton (SFSA), which essentially models the corresponding term. Therefore, through this process, one model for each term is derived.

The final step of the methodology is executed independently and in parallel. It involves the utilization of all previously constructed models in order to predict the annotation of unknown proteins. To this end, the motif sequence of each of the unknown proteins is extracted and run through every available model. This process produces similarity scores for every term, which altogether constitute an accurate prediction of the proteins' annotation. Finally, the classification accuracy of this methodology can be obtained by applying it to the Test Set and comparing the predicted annotation with the actual one. A flow diagram of the methodology is shown in Fig. 1.

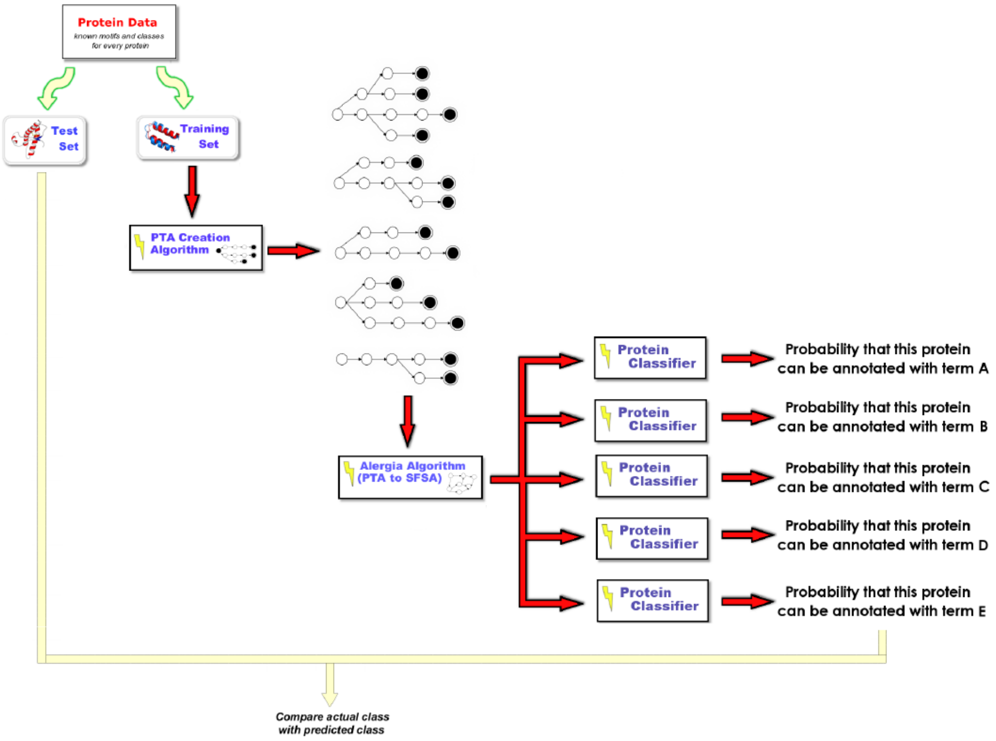
Protein Data
known motifs and classes for every protein
Test Set
Training Set
PTA Creation Algorithm
Alergia Algorithm (PTA to SFSA)
Protein Classifier
Probability that this protein can be annotated with term A
Protein Classifier
Probability that this protein can be annotated with term B
Protein Classifier
Probability that this protein can be annotated with term C
Protein Classifier
Probability that this protein can be annotated with term D
Protein Classifier
Probability that this protein can be annotated with term E
Compare actual class with predicted class

Fig. 1. *Methodology Flow diagram*

3. Training Process

The training process accepts as input the data set for every annotation term and employs data mining algorithms in order to derive a model for each one. At first, all proteins in the data set are used to construct a Prefix Tree Acceptor. This PTA is later transformed into a more generalized Finite State Automaton (FSA) by merging some of its states that comply to certain criteria. The produced FSA constitutes a model for the corresponding term. The key characteristic of the training process is its parallel nature. As it is clearly illustrated in Figure 1, the construction of each term model is completely independent from the construction of the other ones. Therefore the whole training process can be easily parallelized, allowing for further reduction of the execution time. The next part of this section will attempt to describe each phase of the process in greater detail.

The first step of the training process is the construction of a Prefix Tree Acceptor (PTA) using all protein sequences found in the data set. A PTA is a finite state automaton, with a tree-like structure. Using a set of strings (protein motif sequences in our case), the PTA is constructed by merging common prefixes into the same branch of the tree. As an example, Fig. 2 shows a PTA constructed by the string set *{"xyx", "xyy", "xxy", "yxx"}*. The PTA consists of a starting node, four end nodes, and nine transitions among nodes. Moreover, it is obvious that the two strings *"xyx"* and *"xyy"* share a common branch for the first two letters (the identical prefix being *"xy"*), with a split at the last character to mark the differentiation of the two strings.

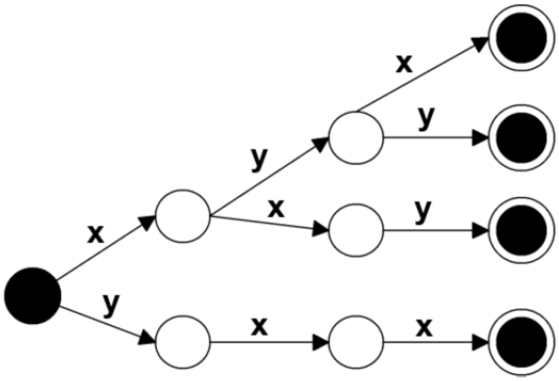

Fig. 2. *A sample PTA*

It is worth pointing out that the constructed PTA does not depend on the order by which the motif sequences were considered. If this order is changed in any way, the same PTA will be generated.

The next step is the transformation of the PTA into an equivalent, more generalized, finite state automaton. The procedure is performed using the Alergia algorithm [Carrasco (1994)], by merging equivalent states (nodes). Two states are considered to be statistically equivalent if they have the same transition probabilities for every symbol and their next states are also equivalent. This brief description of the Alergia algorithm actually implies that it is a recursive algorithm. Therefore one may expect that the execution time will grow exponentially. Experimentally though, it has been proven that the time complexity is linear and mainly depends on the size of the data set used (Fig. 3).

The final stage of the proposed methodology is the utilization of all previously generated models in order to assign annotation terms to new non-annotated protein sequences. This is the point where proteins from the test set are used in order to obtain the accuracy of the methodology.

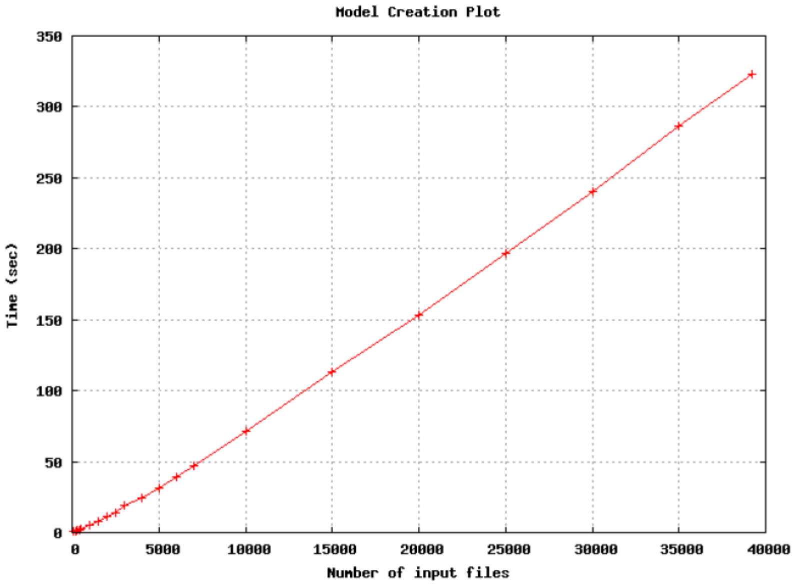

Fig. 3. *Model creation*

The accuracy can be defined as the number of proteins whose annotation was correctly predicted over the total number of proteins in the test set. In more detail, if **A**, **B**, **C** and **D** are the terms that constitute the annotation of a protein in the Test Set and the methodology predicts that **B**, **C**, **D**, **E** and **F** are the most probable terms, the accuracy is defined as *the number of correctly predicted terms over the total number of terms predicted*, in this case, 3/5 = 0.6 or 60%. Another way to calculate accuracy is to divide the number of correctly predicted terms over the total number of terms of the protein. In this case the result would be 3/4 = 0.75 = 75%. The difference between those two methods of calculating accuracy is that, in the second case, no misclassification penalty exists. Therefore the first way of calculating accuracy is preferred:

$$\text{Accuracy} = \frac{\text{Correct Terms Predicted}}{\text{All Terms Predicted}} \qquad \textbf{(2)}$$

The overall accuracy of the methodology can be estimated if one adds up the accuracy that was calculated for every protein and divides the result with the total number of proteins in the Training Set. Namely:

$$\text{Total Accuracy} = \frac{\sum_{i=1}^{N} \text{Accuracy for protein i}}{N}, \qquad \textbf{(3)}$$

where N is the total number of proteins in the Training Set.

A detailed description and mathematical proof of the algorithm used to calculate the probability that a certain motif sequence can be accepted by a Finite State Automaton can be found in [Hingston (2002)].

4. Experiments

In order to validate the correctness of the proposed methodology, a series of experiments were conducted. In these experiments, the Gene Ontology classification schema was exclusively used, but generally, any classification scheme (such as PFam and SCOP) is valid and supported by the methodology.

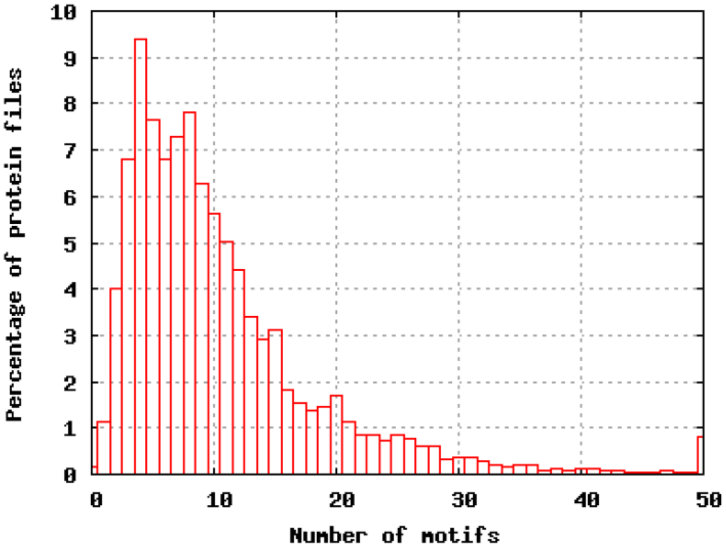

Fig. 4. *Number of motifs distribution in protein dataset*

An important parameter, that significantly affects the execution time, is the number of motifs of each protein to be classified. It was mentioned previously that the time needed to classify a protein depends exponentially on the number of motifs it comprises. Consequently, a threshold must be applied on this parameter, so that large proteins are discarded and the execution time of the application remains at acceptable levels. In Fig. 4 some statistics of the initial Protein Data Set are presented. It is obvious from the diagram that the majority of proteins comprises less than 20 motifs. In addition to this remark, approximately 51% of all proteins in the data set contain 8 motifs and less. Therefore, the threshold value for the maximum number of motifs a protein under classification is allowed to have, was set to 8 motifs.

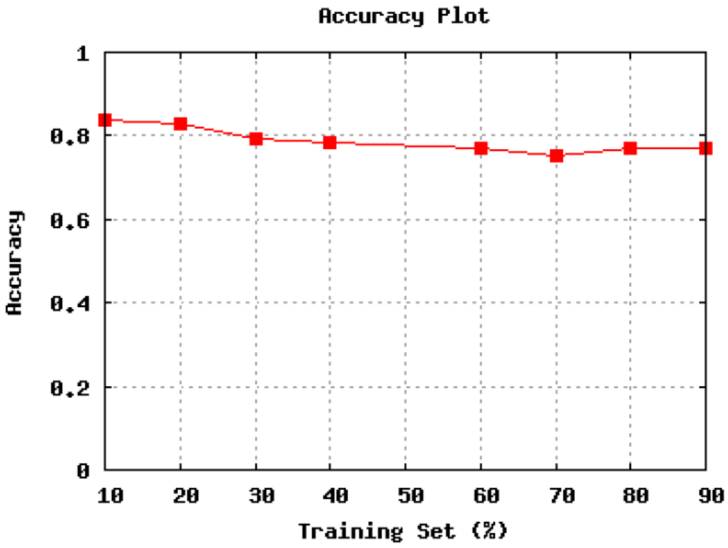

***Fig. 5.** Mean accuracy*

Three protein data sets were defined, containing 10.000, 20.000 and 30.000 protein files respectively. These data sets were subsequently divided into Training and Test Sets according to the following ratios (*Training Set % / Test Set %*): 90/10, 80/20, 70/30, 60/40, 40/60, 30/70, 20/80 and 10/90. Moreover, a variable number of CPUs was used (1, 4, 8 and 16), so that the behavior of our application under various circumstances could be properly evaluated. The results obtained so far seem to be rather encouraging; in all cases the accuracy of the results was relatively high and the overall execution time was satisfactory.

All experiments were conducted using the EGEE Grid infrastructure as a source for multiple CPUs. The Grid is the ideal environment for execution, due to the fact that, by design, the algorithm is an embarrassingly parallel one, allowing for multiple models to be trained simultaneously. The application was executed on available computer clusters in different experiment configurations. The initial data set is stored and replicated as a single compressed file on multiple storage elements (SEs) of the Grid. This compressed file is downloaded locally and decompressed before the application begins.

The mean accuracy for every Training Set / Test Set ratio is presented in Fig. 5. Therein, small variance in accuracy is observed regarding the percentage of the dataset used for training, fluctuating between 0.8364 and 0.7542. The fact that the value of accuracy remains stable across various Training Set / Test Set ratios shows that the proposed methodology produces very accurate results even if the Training Set is a very small portion of the initial Protein Data Set and relatively small compared to the Test Set. Thus, the algorithm is not affected by the Training Set / Test Set ratio and is able to work flawlessly under conditions where the training information is rather limited.

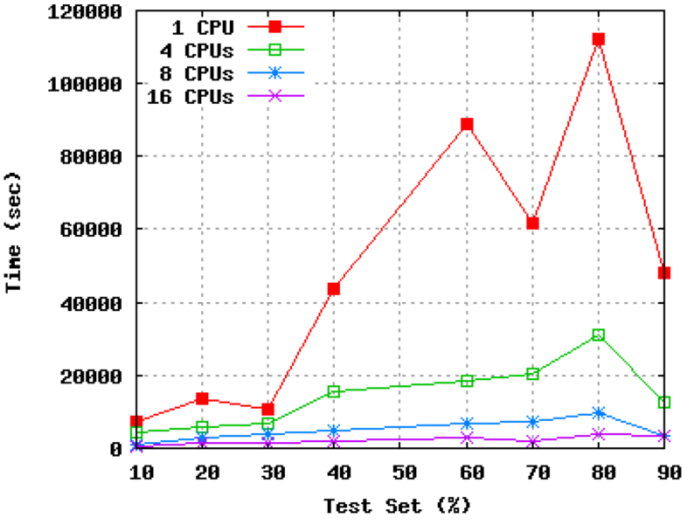

Fig. 6. *Execution Time*

Figure 6 demonstrates the time required to process all protein files of the Test Set. It is obvious that the largest the Test Set is, the more protein files it contains and thus more time is needed for the application to produce results. It is illustrated that significant reduction of the processing time can be achieved if the application runs utilizing more than a single CPU. More specifically, the processing time of the experiment where 8 processors were used is approximately half in comparison to the processing time of the experiment were 4 processors where used. The same principle applies also to the experiments where 16 processors and 8 processors were utilized. This observation proves that the utilization of more than one processor was performed very effectively and that one can expect further reduction of the execution time if more processors are used to run the parallel application.

5. Conclusions and Future Work

In this paper, a parallel data mining methodology for protein function prediction was presented. This methodology applies data mining techniques to protein data from already annotated protein sequences in order to construct a model for every Gene Ontology term. This model is actually a Finite State Automaton, which can then be used in order to predict the annotation of new non-annotated protein sequences. The parallel nature of this methodology has allowed the creation of an MPI-enabled application that can be easily deployed over a Computer Grid, leading to significant reduction of the execution time.

The experiments were conducted using the EGEE Grid environment as a source of multiple CPUs, and clearly indicate that the methodology is highly efficient and accurate. Moreover, the utilization of many processors has reduced the execution time substantially.

References

1. C. Bishop (1995), Neural Networks for Pattern Recognition, Oxford University Press, New York, 1995.
2. R. C. Carrasco, J. Oncina (1994), *Learning stochastic regular grammar by means of state merging method*, Proc. The Second International Colloquium on Grammatical Inference (ICGI '94), Alicante, Spain, Lecture Notes in Artificial Intelligence LNAI 862, pp. 139 - 152, Springer - Verlag.
3. R. Duad, P. Hart (1973), Pattern Classification and Scene Analysis, Wiley, New York.
4. P. Hingston (2002), *Using Finite State Automata for Sequence Mining*, 25th Australian Computer Science Conference, Melbourne, Australia, 105-110.
5. F. Psomopoulos, S. Diplaris, P. A. Mitkas (2004), *A Finite State Automata Based Technique for Protein Classification Rules Induction*, In: Proceedings of the Second European Conference on Data Mining and Text Mining in Bioinformatics, ECML/PKDD.
6. F. E. Psomopoulos, P. A. Mitkas (2005), *A protein classification engine based on stochastic finite state automata*, Lecture Series on Computer and Computational Sciences, Vol. 1.
7. F. E. Psomopoulos, P. A. Mitkas (2006), *PROTEAS: A Finite State Automata based data mining algorithm for rule extraction in protein classification*, 5th Hellenic Data Management Symposium.
8. J. R. Quilan (1992), Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA.
9. D. Wang, X. Wang, V. Honavar, D. Dobbs (2001), *Data-driven generation of decision trees for motif-based assignment of protein sequnces to functional families*, In: Proceedings of the Atlantic Symposium on Computational Biology, Genome Information Systems & Technology.
10. The Gene Ontology Project (http://www.geneontology.org/)
11. Message Passing Interface – MPI (http://www-unix.mcs.anl.gov/mpi/)
12. Enabling Grids for E-sciencE - EGEE(http://www.eu-egee.org)
13. Structural Classification of Proteins - SCOP (http://scop.mrc-lmb.cam.ac.uk/scop/)
14. Uniprot - Universal Protein Resource (http://www.ebi.uniprot.org)
15. InterProScan Sequence Search (http://www.ebi.ac.uk/InterProScan)
16. PROSITE (http://ca.expasy.org/prosite/)
17. Pfam (http://pfam.sanger.ac.uk/)
18. PRINTS (http://www.bioinf.manchester.ac.uk/dbbrowser/PRINTS/)