

# Guiding Motif Discovery by Iterative Pattern Refinement

Zhiping Wang  
School of Informatics  
Indiana University  
901 East 10th Street  
Bloomington, IN 47408-3912  
(812) 856-2707  
zhipwang@indiana.edu

Mehmet Dalkilic  
School of Informatics  
Center for Genomics & Bioinformatics  
Indiana University  
901 East 10th Street  
Bloomington, IN 47408-3912  
(812) 856-3010  
dalkilic@indiana.edu

Sun Kim  
School of Informatics  
Center for Genomics & Bioinformatics  
Indiana University  
901 East 10th Street  
Bloomington, IN 47408-3912  
(812) 856-3009  
sunkim2@indiana.edu

## ABSTRACT

In this paper, we demonstrate that the performance of a motif discovery algorithm can be significantly improved by embedding it into a novel framework that effectively guides the motif discovery process. The framework is also general enough to allow any statistical motif discovery algorithm to be used. Motivation for this research comes from the fact that the statistical significance of patterns depends on the background probability which is largely determined by input sequences. Our framework guides motif discovery by inputting subsequences to an existing motif discovery algorithm, rather than using entire sequences. Subsequences are determined by motifs discovered using existing motif discovery and search algorithms. Then this technique is iteratively applied until convergence. A starting set of patterns is discovered by a simple, but effective pattern set generation algorithm. Our framework was implemented using MEME and MAST and tested with 108 PROSITE patterns. The result demonstrates that our framework significantly improves the performance of MEME.

## Categories and Subject Descriptors

I.5.1 [Pattern Recognition]: Models - *Statistical*

## General Terms

Algorithms, Performance, Experimentation

## Keywords

Motif, pattern, framework, bioinformatics

## 1. INTRODUCTION

Motifs are short, conserved patterns in a family of sequences, and they are important signatures for the family. Motifs mostly exist in regions where complex protein relations and interactions occur such as enzyme catalytic sites, ATP binding sites and protein-

protein interacting sites. They are also found in the fold along with conserved domains (or even residues), which are important for general 3D macromolecular structure and function, *e.g.*, membrane pore, membrane signaling and proteases.

Motifs can be computationally discovered as patterns common to a family of sequences either from sequences (sequence pattern) or from structures (structure patterns). Here, we discuss sequence motifs only. The most common techniques for motifs discovery can be divided into two categories, statistical [1, 8] and combinatorial [9, 10] pattern discovery. These motif discovery algorithms have successfully found many biologically relevant patterns. However, the quality of performance varies with the input sequences' characteristics *e.g.*, diversity. For example, all known motifs in disease resistance genes in *Arabidopsis thaliana* were successfully found using MEME after splitting the sequences into two distinct categories of resistance genes, but no motifs were found by inputting all disease resistance genes as a single input file to MEME [5].

In this paper, we propose a novel framework for motif discovery into which any statistical motif discovery algorithm can be embedded. Our framework uses motifs, discovered by existing motif discovery and search algorithms, to define subsequences and then apply this technique in an iterative fashion until convergence. An initial set of patterns are discovered by a simple, but effective pattern set generation algorithm. Our framework was implemented using MEME and MAST, and tested with 108 PROSITE patterns. The result showed that our framework significantly improves the performance of MEME.

### 1.1 Motivation for Our Research

Statistical pattern discovery techniques look for statistically significant signatures. One of the most widely used methods, based on Neymann-Pearson Lemma, is to compare two models: a model for the pattern (signal model) and a model for negative examples (noise model). Thus, the performance—accuracy—of most motif discovery algorithms depends on both signal and noise models. Given this signal-to-noise *perspective*, we can understand better why the performance of most motif discovery algorithms differ for different input sequences: input sequences determine the background noise model. For example, it is shown that the performance of motif discovery algorithms can be significantly improved by clustering input sequences into smaller groups and then submitting each group to motif discovery algorithms [5]. Recently T.G. Lescot *et al* [8] demonstrated the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'04, March 14-17, 2004, Nicosia, Cyprus.

Copyright 2004 ACM 1-58113-812-1/03/04...\$5.00.

degree of effect of background noise on motif detection. By the same token, we conjecture that motif discovery can be guided by subsequences, instead of using whole sequences. Indeed, this technique (subsequence selection) is manually used in practice by many bioinformaticians and biologists. For example, entire genome sequences are never used to find promotor binding sites—rather subsequences that are extracted from upstream regions of genes of the same type. However, it is quite difficult to select correct subsequence regions without prior knowledge, *e.g.*, genes of the same type. Motivation for our research is to use motifs, discovered by existing motif discovery and search algorithms, to define subsequences and then apply this technique in an iterative fashion until convergence.

## 1.2 An Experiment

Can subsequence selection guide motif discovery? We can witness this phenomenon in an experiment with a pattern PS00343 (L-P-x-T-G-[STGAVDE]) from the PROSITE [3, 4] database. A family of sequences with the PROSITE pattern is collected and aligned using CLUSTALW [11] and then submitted to a Logo generation program [12]. Figure 1 shows the motif logos [7] of the protein family. Figure 1(a) shows the motif logo for the multiple sequence alignment of the family. As one would expect, there are many peaks in the figure, and it is hard to tell which peak represents the conserved motif for this family. Figure 1(b) shows the motif logos that correspond to the multiple sequence alignment region, marked by blue arrow in Figure 1(a), that contains the PROSITE motif. The columns that correspond to the motif have high bit values, but are not very distinct compared to the bit values in the region surrounding the motif. Next, we extracted subsequences around the motif in each sequence in the family, and then submitted the set of subsequences to the sequence logo program. The resulting motif logo is shown in Figure 2. The columns that correspond to the motif have much higher bit values than those in the region surrounding the motif. Readers may notice that columns in the motif regions in Figure 1(b) and Figure 2 have different sets of characters. For example, the column corresponding to the first motif character in Figure 1(b) has L, T, G, V, S, A and K while one in Figure 2 only has L, thus resulting in different bit values in the logos. This is because CLUSTALW aligned sequences differently in Figures 1(b) and 2. This experiment shows that even a less powerful multiple sequence alignment program can detect motifs when only subsequences that contains the motif is submitted. Thus, it is quite clear that motif discovery can be guided by inputting subsequences that contain the motif.

## 1.3 Challenges

However, we do not have prior knowledge where motifs locate in the input sequences. Thus, guiding motif discovery by inputting subsequences that contain motifs is not obvious. Our framework is designed to address this problem. We first extract subsequences that are likely to contain motifs, and iteratively resample set of subsequences until convergence. To this end, we have to answer the following two issues.

1. Some subsequences in the initial set of subsequences should contain motifs. If not, the iterative resampling step will not converge to the set of subsequences that contain motifs. This issue is addressed in Sections 2.3 and 2.4.

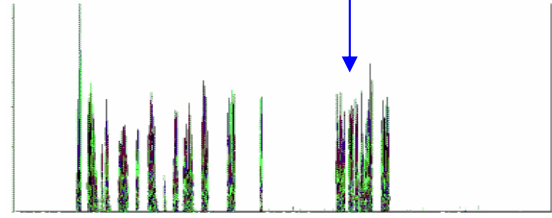


Figure 1a. Motif logo for a protein family.

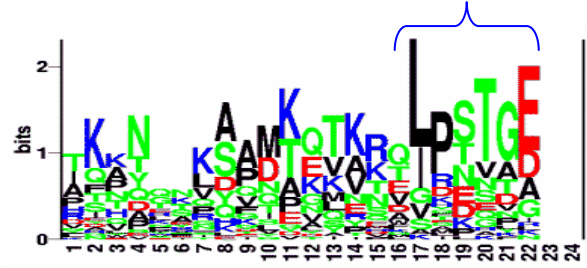


Figure 1b. Motif logo for a motif .

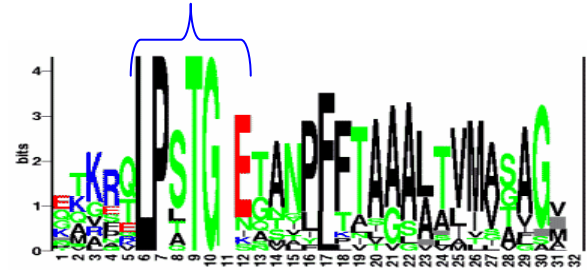


Figure 2. Motif logo for conserved subsequences of the protein family.

2. How can we resample subsequences? This issue is discussed in Sections 2.2 and 2.5.

## 2. METHOD

### 2.1 Test Data Preparation

PROSITE patterns were used to test the performance of our framework. We downloaded protein sequences database and pattern database from the PROSITE website [6]. For each PROSITE pattern, we parsed its true positive sequences to a family named as its PROSITE ID. Thus all sequences in one family contain the same pattern. We used our motif discovery framework on the protein families and compared the motifs from our framework with PROSITE patterns. We also ran MEME directly on these PROSITE families and got a motif for each of them. We then compared the performance of our framework to MEME (standalone).

### 2.2 Overview of the Framework

Our framework is an iteration of the following steps:

*STEP1.* Extract a set  $S$  of subsequences around a set of motifs  $M$ .

*STEP2.* Input  $S$  to a motif discovery algorithm, producing a new set of motifs  $M'$ .

*STEP3.* Search entire sequences for more occurrences of  $\mathbf{M}'$ , producing  $\mathbf{M}''$ . Set  $\mathbf{M}'''$  to  $\mathbf{M}$  and go to step 1.

STEP1 uses a set of motifs resulting from STEP3 in an iterative fashion but an initial set of motifs is required to make the iteration to start. The initial motif set is constructed by collecting a set of statistically significant patterns (Line 1 in Algorithm1; see also Section 2.3) and then linking the initial patterns together to construct a set of “seed motifs” for the protein family (Line 2 in Algorithm1; see Section 2.4).

STEP 2 utilizes a motif discovery program of choice (MEME [1] in this case). STEP3 uses a motif searching program of choice (MAST [2] in this case) to find more occurrences of the motifs in the entire sequences. It should be emphasized that motif searching in STEP3 is done on the “entire” sequences (Line 8 in Algorithm 1) while motif discovery is done on the subsequences (Line 7 in Algorithm 1). The iteration is on the set of subsequences (Line 5 in Algorithm 1) that is defined by motifs (Lines 3 and 9 in Algorithm1). The whole framework is illustrated in Algorithm 1.

```

Input:  $\mathbf{S} = \{S_1, S_2, \dots, S_n\}$  // a set of sequences
Output:  $\mathbf{M} = \{M_1, M_2, \dots, M_m\}$  // a set of motifs
MOTIF-SEARCH-FRAMEWORK {
1)    $\mathbf{P} = \text{initial\_pattern\_discovery}(\mathbf{S}, \text{thresholds})$ 
2)    $\mathbf{M} = \text{seed\_motif\_construction}(\mathbf{P})$ 
3)    $S = \text{subsequence\_extraction}(\mathbf{M}, \mathbf{S})$ 
4)    $S' = \emptyset;$ 
5)   while ( $S' \neq S$ ) {
6)        $S' = S$ 
7)        $\mathbf{M} = \text{motif\_discovery}(S)$ 
8)        $\mathbf{M}' = \text{motif\_occurrence\_search}(\mathbf{M}, \mathbf{S})$ 
9)        $S = \text{subsequence\_extraction}(\mathbf{M}', \mathbf{S})$ 
    }
10)  output  $\mathbf{M};$ 
}

```

**Algorithm 1. Motif discovery framework.**

## 2.3 Initial Pattern Discovery

At this point, we collect a set of initial patterns in the set of input sequences. This set of patterns will be used to build a set of seed motifs later (see Section 2.4). The set of initial patterns are exact patterns of a fixed length  $L$  that meet two conditions:

1. Patterns should be statistically significant, and
2. Patterns should be present in a certain number of sequences.

In this paper,  $L$  is fixed to 3 since exact patterns longer than 3 do not occur frequently even in the conserved motif regions. The statistical significance of patterns is required since motifs will occur significantly more frequently than random patterns, which is the basis for most statistical motif discovery algorithm. The second condition is required since motifs are recurring patterns common in “multiple” sequences; patterns that occur multiple times in a sequence but do not occur in any other sequences are not qualified. This will be explained in Section 2.3.1.

In addition to the two conditions, the set of patterns should represent “all” input sequences while multiple different patterns can be sampled around motif regions. For this reason, we use a

procedure that iteratively discarding top half of sequences where patterns are already sampled while looking for patterns that meet the two conditions. This will be discussed in Section 2.3.2.

### 2.3.1 The two conditions for patterns

The statistical significance is measured by a ratio of the first order Markov model to a random model. As an example, let  $S$  be a sequence of amino acids, *e.g.*  $S = \alpha_1\alpha_2 \dots \alpha_n$ . The probability of  $\alpha_i$  in  $S$ , denoted  $P(\alpha_i)$ , is simply the ratio of the number of times  $\alpha_i$  occurs in  $S$  to  $n$ . The conditional probability of  $\alpha_i$  given  $\alpha_j$ , denoted  $P(\alpha_j | \alpha_i)$ , is the ratio of  $P(\alpha_j\alpha_i)$  to  $P(\alpha_i)$ . The first order Markov model for  $S$  is  $P_M(S) = P(\alpha_1) \times \prod_{i=1}^{n-1} P(\alpha_{i+1} | \alpha_i)$ . The random model for  $S$  is  $P_R(S) = \prod_{i=1}^n P(\alpha_i)$ . The log-odd score of the sequence  $S$ , denoted  $E(S)$ , is defined as  $E(S) = \log(P_M(S)/P_R(S))$ .

The log-odd score can be found for any subsequence  $\alpha_i\alpha_{i+1}\alpha_{i+2}$  of  $S$  by using the initial overall probabilities and setting  $n = 3$ . We consider all patterns of length  $L = 3$  in the input sequences and consider them statistically significant if their log-odd score is positive, *i.e.*,  $E(S) > 0$ .

The second condition that patterns should be present in a certain number of sequences can be simply enforced by a support value  $K$ , the occurrence of a pattern in at least  $K$  different sequences. This avoids the case that one pattern occurs many times in one sequence, but rarely appears in other sequences. So the support value is set to make sure the qualified patterns are common features for the family.

### 2.3.2 Algorithm for initial pattern discovery

Since patterns define subsequences that are the starting point of the iterative procedure (Algorithm 1), we have to make sure that all sequences are represented by patterns. To ensure this, we use another iterative procedure (Algorithm 2). At each iteration step, we rank the sequences according to the number of qualified patterns they have, eliminate the top half of the sequences and leave the rest half for the next iteration step. Since patterns are sampled only from sequences not represented by the current pattern set, it is ensured that all sequences are represented by some patterns. The resulting patterns will be used to build an initial seed motif (explained in the next subsection).

## 2.4 Seed Motif Construction and Extension

From a set of statistically significant patterns, we will build seed motifs so that the iterative motif search can be initiated. The basic idea is to collect a set of similar patterns that can cover all input sequences. The similarity between patterns is measure by computing a score using an amino acid scoring matrix (BLOSUM62), *i.e.*, a global alignment of two patterns without gaps.

To build seed motifs, we first rank the patterns according to their support values and use the pattern in the top ranking as a starting point for one seed motif. Actually, a starting pattern is a seed motif now. All the other patterns, which are not present in the sequences covered by the seed motif, become *candidate patterns*. We calculate the similarity score of each candidate pattern to the

patterns in the seed motif, then choose a pattern with the highest score and add it to the motif. Here we also set a threshold ( $T$ ) to limit the degree of similarity between the added patterns and the seed motif. We repeatedly add one pattern at a time to the seed motif until no more similar patterns can be added to the seed motif.

```

Input: S // A set of sequences
Output: P // a set of patterns
 $P_q = \emptyset$  // the set of qualified patterns
 $S_r = \emptyset$  // sequence set represented by qualified patterns
while( $|S_r| < n$ ){
   $S_r = S_r \cup \text{qualified\_pat}(L, K, T, S - S_r)$ 
}
return  $P_q$ 

qualified\_pat( $L, K, E, S'$ ) {
  find qualified patterns (meet thresholds  $L, K, T$ ) in
  sequence set  $S'$ ;
  rank  $S'$  according to number of qualified patterns in
  each sequence;
   $S'' = \text{top half of } S'$ ;
  for each qualified pattern  $P \in S''$ ,  $P_q = P_q \cup \{P\}$ ;
  return  $S''$ 
}

```

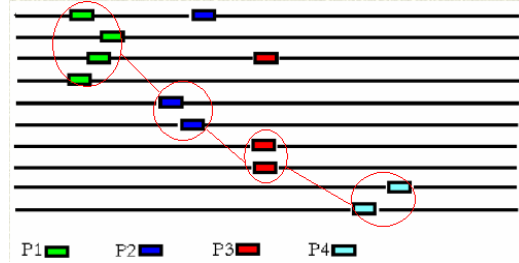
**Algorithm 2. Initial pattern discovery algorithm.**

To calculate the similarity score  $S_i$  of a candidate pattern to the seed motif, we first compare the candidate pattern  $P_i$  with each pattern  $P_j$  in the seed motif and get a score  $S_{ij}$  based on a score matrix, *i.e.* BLOSUM62. So, the score  $S_{ij}$  shows how similar the candidate pattern  $P_i$  is to each pattern  $P_j$  already in the seed motif, *i.e.*, a similarity score of a global alignment of the two patterns without gaps using the BLOSUM62 scoring matrix. However, since some patterns have higher support ratio in the seed motif, they should have larger impact on the candidate pattern similarity calculation. So we assign a weight support ratio  $W_j$  to each pattern of seed motif. The final formula to calculate the similarity score for candidate pattern  $P_i$  to the seed motif is:

$$S_i = \sum_{j=1}^n S_{ij} W_j \text{ where } n \text{ is the number of patterns.}$$

Figure 3 is an example of building one seed motif for a protein family. The protein family shown in Figure 3 contains four conserved patterns (P1, P2, P3 and P4), which are indicated by different colors. In this scenario, P1 starts as the seed motif since it has the highest support value. P2, P3 and P4 become candidate patterns. Then the formula above is used to calculate the scores of P2, P3 and P4 to the seed motif {P1}. Suppose P2 has the highest score which is also larger than threshold ( $T$ ). Then P2 is added to the seed motif. So the seed motif becomes {P1, P2}. Similarly, P3 and P4 are added to the seed motif. Now we have a seed motif, which is linked together by the red circles in the figure, for the protein family.

Since seed motifs are actually built up by adding the most frequent conserved patterns together, it is likely that at least some of them should be a part of the true motif in the protein sequences. Finally, we extract subsequences of a certain length (40 bases) around the patterns in the seed motif.



**Figure 3. A seed motif is built for a protein family.**

## 2.5 Motif Discovery and Iterative Refinement

Once a set of subsequences is extracted around the seed motif, we can initiate the iterative pattern refinement procedure:

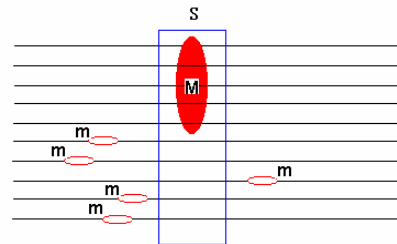
**STEP1.** Extract a set  $S$  of subsequences around a set of motifs  $\mathbf{M}$ .

**STEP2.** Input  $S$  to a motif discovery algorithm, producing a new set of motifs  $\mathbf{M}'$ .

**STEP3.** Search entire sequences for more occurrences of  $\mathbf{M}'$ , producing  $\mathbf{M}''$ . Set  $\mathbf{M}''$  to  $\mathbf{M}$  and go to step 1.

The iterative refinement is necessary since there is no guarantee that the subsequences (**STEP1**) contain all occurrences of true motifs. Figure 4 illustrates such a situation where only five occurrences of true motif are included in the subsequences (the boxed region with the label  $S$ ). Once a model for motif is built in **STEP2** using a motif discovery algorithm (MEME in this case), we can search more occurrences of the model in the “entire” sequences; in Figure 4, five more occurrences of the model (labeled as  $m$ ) can be found by a motif occurrence search program (MAST in this case).

As long as more motif occurrences are found in **STEP3**, the iteration can refine the motif model. However, two important questions for this iterative pattern refinement framework discussed in Section 1.2 must be addressed. In this paper, we empirically demonstrate that our framework does work for the PROSITE patterns; initial subsequences contain some occurrences of true motifs (Line 3 in Algorithm 1) and the iteration converges to the regions where true motifs locate (Lines 5 through 9 in Algorithm1). Theoretical study for the two issues remains as a future study.



**Figure 4. Motifs in the subsequences (labeled M) and outside the subsequences (labeled m).**

### 3. EXPERIMENT AND RESULT

The framework was implemented on a Pentium IV dual processor 1.7 GHz 4GB RAM machine running Red Hat Linux 8.0. MEME, a parallel version, was run on an IBM SP cluster (<http://sp-www.iu.edu>) for entire input sequences because in general it takes too long on the single processor unit. The initial pattern finding and seed motif construction algorithms were written in Perl (Section 2.3 and 2.4). To test the performance of the framework, we used 108 PROSITE patterns and corresponding families as the test data to discover motifs (Section 2.1 for more detail). For each PROSITE pattern, we ran our framework and standalone MEME on the sequence family corresponding to the pattern. Because of time constraints, our motif framework performed only single iteration.

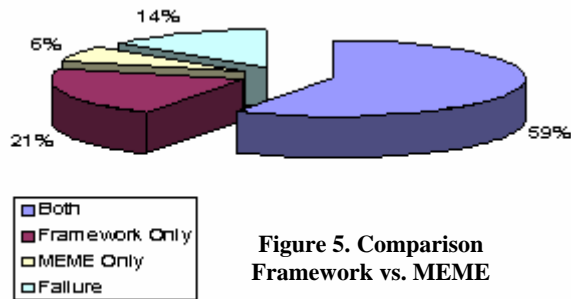


Figure 5. Comparison Framework vs. MEME

Our results are as follows. Both MEME and our framework discovered 63 PROSITE patterns. Our framework found additionally 23 patterns that MEME failed to find. MEME found 7 patterns that our framework failed to find. Both MEME and our framework failed to find 15 patterns. The result is summarized in Figure 5. From the figure we can observe that MEME's failure rate 35% versus our framework at 20%. MEME reports multiple motifs associating a rank with each. To make our experiment more rigorous, we choose only the top motif reported by both MEME and our framework. We have provided an abridged listing of families discovered by our framework and MEME at the following link ([http://biokdd.informatics.indiana.edu/papers/ACM\\_SAC04](http://biokdd.informatics.indiana.edu/papers/ACM_SAC04)).

Among the 22 failed cases, our framework did discover 21 of them, though their rank was not top.

### 4. DISCUSSION

We proposed a novel motif search framework into which any existing statistical motif search algorithm can be embedded. The prototype implementation using MEME and MAST successfully improved the accuracy of MEME on a test data set from the PROSITE patterns. This framework is general enough to include any motif discovery and search algorithms that report multiple motifs with a statistical score.

Several issues remain as future study. How likely subsequences induced by our initial pattern discovery algorithm can include true motifs? Is convergence to true motif regions guaranteed once the initial set of subsequences contains true motifs? How our framework will perform under multiple iterations? On the theoretical side, we are interested in formalizing and understanding the role of noise. We plan to perform an exhaustive empirical study using the whole PROSITE pattern set as well as a

theoretical study on these issues. We also plan to embed different motif discovery and search programs into our framework.

### 5. ACKNOWLEDGMENTS

Dalkilic partially supported by NSF IIS-0082401, INGEN, IBM Life Sciences and Kim by NSF Career DBI-0237901, INGEN.

### 6. REFERENCES

- [1] Timothy L. Bailey and Charles Elkan, "Fitting a mixture model by expectation maximization to discover motifs in biopolymers", *Second International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pp. 28-36, 1994
- [2] Timothy L. Bailey and Michael Gribskov, "Combining evidence using p-values: application to sequence homology searches", *Bioinformatics*, Vol. 14, pp. 48-54, 1998.
- [3] Sigrist C.J., Cerutti L., Hulo N., Gattiker A., Falquet L., Pagni M., Bairoch A., Bucher P., "PROSITE: a documented database using patterns and profiles as motif descriptors", *Brief Bioinform*, 3: 265-274, 2002.
- [4] Falquet L., Pagni M., Bucher P., Hulo N., Sigrist C.J., Hofmann K., Bairoch A., "The PROSITE database, its status in 2002", *Nucleic Acids Res.*, 30: 235-238, 2002.
- [5] Irfan Gunduz, Sihui Zhao, Mehmet Dalkilic and Sun Kim, "Motif Discovery from Large Number of Sequences: A Case Study with Disease Resistance Genes in Arabidopsis thaliana", *The 2003 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS'03)*, pp 29-34, 2003.
- [6] <http://www.expasy.org/prosite/>
- [7] J. Gorodkin, L. J. Heyer, S. Brunak and G. D. Stormo, "Displaying the information contents of structural RNA alignments: the structure logos", *Comput. Appl. Biosci.*, Vol. 13, no. 6 pp 583-586, 1997.
- [8] Thijs G., Lescot M., Marchal K., Rombuats S., De Moor, B. Rouze P, Moreau Y., "A higher order background model improves the detection of regulatory elements by Gibbs Sampling", *Bioinformatic*, 17(12) pp. 1113-1127, 2001.
- [9] I.Jonassen, J.F.Collins, D.G.Higgins, "Finding flexible patterns in unaligned protein sequences," *Protein Science* 4, pp1587-1595, 1995
- [10] I. Rigoutsos and A. Floratos, "Combinatorial pattern discovery in biological sequences: The TEIRESIAS algorithm," *Bioinformatics* 14: pp55-67, 1998
- [11] Thompson,J.D., Gibson,T.J., Plewniak,F., Jeanmougin,F. and Higgins,D.G. (1997) "The ClustalX windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools." *Nucleic Acids Research*, 24:4876-4882.
- [12] <http://www.cbs.dtu.dk/~gorodkin/appl/plogo.html>