

# Motif Discovery for Proteins Using Subsequence Clustering

Hardik A. Sheth  
School of Informatics  
Indiana University  
901 E. 10th St.  
Bloomington, IN 47408-3912  
hsheth@indiana.edu

Sun Kim  
School of Informatics  
Center for Genomics and Bioinformatics  
Indiana University  
901 E. 10th St.  
Bloomington, IN 47408-3912  
sunkim2@indiana.edu

## ABSTRACT

We propose an algorithm for discovering motifs using clustering of subsequences. In our previous approach, we were successful in guiding motif discovery by sampling subsequences and inputting them to an existing motif discovery tool MEME. In this paper, we show that clustering subsequences can also detect motifs without using other motif discovery tools. Generally, motif discovery algorithms do not perform well when the input set consists of non-homogeneous sequences. Clustering tools have the inherent ability to generate clusters of homogeneous sequences when the input sequences are non-homogeneous. For this reason, we use our clustering algorithm to generate aligned subsequence clusters and then rank them according to their information contents to produce final motifs. The algorithm was tested with PROSITE database and the results suggest that the algorithm is very effective in finding motifs even when input sequences are from different protein families.

## Categories and Subject Descriptors

I.5.1 [Pattern Recognition]: Models - Statistical

## General Terms

Algorithms, Performance, Experimentation

## Keywords

Motif Discovery, Subsequences, Clustering, Pattern, Motif

## 1. INTRODUCTION

Motifs are short, conserved subsequences that are part of a family of sequences. The use of protein sequence patterns (or motifs) to determine the function of proteins is an

essential tool for sequence analysis. The sequence of an unknown function might not be closely related to any protein of known structure to detect similarity by overall sequence alignment, but it can be found sometimes more accurately by the occurrence in its sequence of a particular cluster of residue types a.k.a pattern, motif, signature, or fingerprint. Some regions are conserved because of particular requirements on the structure of specific region of a protein which may be important, for example, for their binding properties or for their enzymatic activity.

Motifs can be discovered as subsequences that are common to the family of sequences from sequence patterns (subsequences). Since the dramatic increase of genetic data, clustering and motif finding techniques have become essential to the analysis of protein and nucleic acid sequences. As a result, a number of different clustering algorithms and motif discovery algorithms have been developed. Although these two techniques are being well studied, little work has been done to combine them and make an optimized motif discovery tool for general sequence analysis purposes. In this paper, we demonstrate how clustering of sequences can be used for motif prediction.

## 1.1 Background

Existing motif discovery algorithms can be classified into two groups, one group of algorithms, such as PRATT[4] and TEIRESIAS[9], that search for motifs by combinatorial enumeration of patterns and another group of algorithms, such as MEME[1] and GIBBS[6], that search for motifs that are statistically significant. Our interest in this paper is to further develop the second group of algorithms that search for statistically significant sequence patterns. Among them, Gibbs and MEME are the most widely used ones. Although these algorithms have been successful in predicting biologically meaningful sequence patterns, no algorithm works perfect as the motif discovery problem is very difficult to solve. The problem with MEME is that it takes a lot of time to find motifs because it uses multiple rounds of expectation maximization technique in search of motifs. The Gibbs algorithm has a random search behavior and each execution generates different motifs.

A more serious problem is that motif discovery algorithms do not perform well when the input set consists of non-homogeneous sequences. In particular, it is a challenge for all existing motif discovery algorithms to detect motifs when

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, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

BIOKDD '05, August 2005, Chicago, Illinois, USA

Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...\$5.00.

the input set contains sequences from multiple families since a motif occurs only in a subset of the input sequences and the number of motifs to be discovered is not known a priori. The problem can be stated as follows.

In a statistical sense, motif discovery is to look for signals compared to noise in the input sequences.

$$\log\left(\frac{M_{motif}}{M_{noise}}\right)$$

where  $M_{motif}$  is a model for the motif and  $M_{noise}$  represents the model for background noise.

Different input sequences affect both  $M_{motif}$  and  $M_{noise}$ .  $M_{motif}$  requires selection of subsequences that will construct candidate motifs and  $M_{noise}$  is largely determined by the input sequences. More specifically, both models are character frequencies at specific positions as shown in Table 2 and these character frequencies are determined, with some prior knowledge, by counting the number of characters in a specific column. Thus they are directly influenced by selection of subsequences and the input sequence set.

## 1.2 Research Question & Motivation

The question that we are looking to answer here can be stated as follows -

Can there be an efficient tool that discovers motifs from a heterogeneous set of sequences when -

1. Width of each motif is not known
2. Number of motifs to be found is not known ?

We try to explore this problem by applying the techniques of clustering and applying it to motif discovery question. We have been successful in guiding motif discovery by clustering sequences[3] and by sampling subsequences[11]. In this paper, we combine these two approaches in a different context. Once subsequences are selected by a method proposed in [11](see also Section 2.1 in this paper for an improved subsequence selection method), it is assumed that there is a single motif. However, it is possible that there may be more than one motif in the set of subsequences. To deal with this problem, we used a sequence clustering algorithm inspired by our previous approach[3]. Instead of clustering the input sequences directly as in, a set of subsequences of equal length are generated and then clustered using our clustering algorithm as described in Sections 2.2, 2.3, and 2.4. This procedure does not require information about how many motifs need to be sought, which is a unique feature of our motif algorithm.

## 2. METHODOLOGY

Given a large data set of  $N$  protein sequences  $S_1, S_2, \dots, S_N$ , the goal is to identify the conserved regions that represent this data set. Our algorithm is designed to proceed in the following manner -

1. Extract an initial set of patterns.
2. Select subsequences and align them.
3. Cluster those subsequences.
4. Extract conserved regions from shared ranges for each cluster.
5. Extend/Merge the conserved regions.

6. Rank those conserved regions based on information content.

7. Filter and select motifs.

### 2.1 Initial Pattern Discovery

From the set of input sequences, a set of initial patterns is collected. The set of initial patterns are exact patterns of a fixed length  $l$  that satisfy two conditions -

1. Patterns are statistically significant.
2. Patterns are present in certain number of input sequences.

In our experiments,  $l$  is fixed to 10 which is an empirically determined value - exact patterns longer than 10 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 single sequence but do not occur in any other sequences are not qualified.

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 discards the top half of sequences where patterns are already sampled while looking for patterns that meet the two conditions as described in Section 2.1.2.

#### 2.1.1 Two Conditions for Patterns.

The condition for statistical significance of a pattern in our previous paper was based on the first order Markov model. We improve this by using the second order Markov model. The challenge is that it is difficult to measure the second order dependency for a short pattern. For example, only the third character, or after, in a pattern of length 6 can have two preceding characters. Given that the second order dependency cannot be measured for the first two characters, it is not very effective to use the second order Markov model for a short pattern. Thijs et al[10] used characters preceding the motif, i.e., those outside the motif, to compute higher order Markov dependency and used the higher order model as background model to improve the performance of the Gibbs motif algorithm. Inspired by this work, we modified our statistical significance condition to use the second order Markov model as follows.

Let  $x$  be a sequence of amino acids, e.g.  $x = x_1x_2 \dots x_l$ . The probability of  $x$  for a given second order Markov model  $M$  is

$$P_M(x) = \prod_{i=1}^l P(x_i|x_{i-2}x_{i-1})$$

where  $P(x_1|x_{-1}x_0) = P(x_1)$  and  $P(x_2|x_1x_0) = P(x_2|x_1)$  if  $x_0$  and  $x_{-1}$  are not available. The probability of  $x$  for a given random model  $R$  is  $P_R(x) = \prod_{i=1}^l P(x_i)$ . Then the log-odd score of the sequence  $x$ , denoted  $E(x)$ , is defined as

$$E(x) = \log\left(\frac{P_M(x)}{P_R(x)}\right)$$

The log-odd score can be found for any pattern  $x_i x_{i+1} x_{i+2}$  of  $x$  by using the initial overall probabilities and setting

**Table 1: Algorithm for Initial Pattern Discovery**

```

Input:  $S$  // a set of sequences
Output:  $P$  // a set of patterns
 $P_q = \phi$  // a set of qualified patterns
 $S_r = \phi$  // sequence set represented by  $P_q = \phi$ 
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  $P_q$  (meet thresholds  $l, K, T$ ) in sequence set  $S'$ ;
    Rank  $S'$  according to the number of  $P_q$  in each sequence;
     $S''$  top half of  $S'$ ;
    for each qualified pattern  $P \in S''$ ,
         $P_q = P_q \cup \{P\}$ ;
    return  $S''$ 
}

```

$l = 10$ . We consider all patterns of length 10 in the input sequences and consider them statistically significant if their log-odd score is greater than threshold  $T$ , i.e.  $E(x) > T$ .

The second condition that patterns should be present in a certain number of sequences can be simply enforced by a support ratio  $K$ , the occurrence of a pattern in at least  $K\%$  of the 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.

### 2.1.2 Algorithm for Initial Pattern Discovery

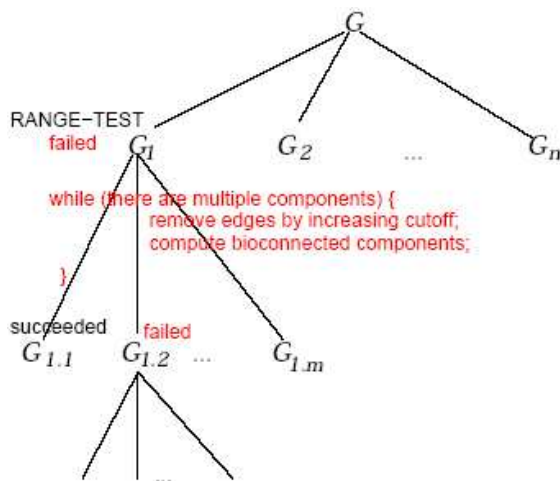
At each iteration step, we rank the sequences according to the number of qualified patterns they have, eliminate the top half of the sequences and leave the rest half for the next iteration step. Since patterns are sampled only from sequences not represented by the current pattern set, it is ensured that all sequences are represented by some patterns. The algorithm used for initial pattern discovery is shown in Table 1. In the table,  $l$  denotes the length of the subsequence,  $K$  represents the support ratio as explained above and  $T$  is the log-odds threshold above which a subsequence is considered significant.

## 2.2 Subsequence Selection and Pairwise Alignment

The set of patterns selected using the procedure in Section 2.1 form the initial set of subsequences. These subsequences are aligned using standard scoring matrix (e.g. BLOSUM62). The scoring or weight matrix is a standard method for representing the variation in a set of sequence patterns in a multiple sequence alignment, and as a tool for finding additional sequences with the same pattern in a database search. The odds score at every possible matching location along the subsequence may be used to find the probability of each sequence location. FASTA algorithm[8] is used to align those subsequences. The FASTA algorithm runs quite fast and is ideally suited for aligning large number of initial subsequences.

## 2.3 Clustering of Subsequences

Those aligned subsequences are then grouped into different clusters using our sequence clustering algorithm[5].



**Figure 1: The graph is iteratively refined until each biconnected component has a common shared region to all sequences in it.**

Given a set of sequences, our clustering algorithm builds a weighted graph based on similarities between those sequences. A node is created for each sequence  $s_i$  and an edge between two sequences,  $s_i$  and  $s_j$ , is created when the pairwise alignment score of  $s_i$  and  $s_j$  is more significant than a preset threshold. The alignment score is associated with the edge as weight so that clusters can be refined while increasing cutoff for edges. Our algorithm uses two graph properties for sequence clustering: *biconnected components* and *articulation points*. A biconnected component of a graph  $G$  is a maximal subgraph where there exist at least two edge disjoint paths for any pair of nodes, and an articulation point is a node that disconnects the graph if it is removed. A biconnected component corresponds to a family of sequences and an articulation point to a multi-domain protein. Each biconnected component is tested whether all sequences share a common shared region - this test is called RANGE-TEST. If there is no common shared region in a component, the cutoff score for the weight in the corresponding graph is increased until the graph is split into multiple biconnected components. The overall clustering procedure is depicted in Figure 1.

The set of input subsequences  $\mathcal{S}$  is split into multiple clusters  $C_1, C_2, \dots, C_n$  based on the sequence similarity among sequences in  $\mathcal{S}$  using our clustering algorithm such that all sequences in each cluster have a common shared region of length  $L$ , by default  $L = 6$  amino acids. In this way, subsequences are grouped together according to their sequence similarity defined by the scoring matrix, e.g., BLOSUM62.

A sample of the clusters generated by our sequence clustering algorithm is shown in Figure 2.

## 2.4 Extracting Conserved Regions

Our clustering algorithm produces clusters of these subsequences depending on their match score. However, we are not interested in all of these subsequences. The clustering algorithm also gives us a list of ranges which all the member subsequences of the cluster share. This eliminates all the clusters that are split and may not consist of conserved regions. Subsequence regions that are shared by all

```

...
CLUSTER 66 size= 5
  LGCC
  INCI ARTI
  RNCC
  KGCC
  RGCC
ENDCLUSTER

CLUSTER 67 size= 2
  VNCG ARTI
  VDCG
ENDCLUSTER

CLUSTER 68 size= 2
  VNCG ARTI
  INCI ARTI
ENDCLUSTER

CLUSTER 69 size= 2
  MVAA
  IVAA
ENDCLUSTER
...

```

Figure 2: Sample Clustering Output

the member subsequences of the cluster are extracted and these are the conserved regions that we are interested in. A sample output of shared ranges generated is shown in Figure 3. In the figure, shared regions of subsequences can be aligned without gaps to create a position weight matrix or a profile. Such shared regions can be determined by the start and end positions of the range produced by our clustering algorithm; the shared region starts at position 1 and ends at position 5 for the subsequence 'GFIKCV 1 5' of the cluster 39.

```

...
SHARED RANGES for cluster 39
  GFIKCV 1 5
  GFIQCS 1 5
  FGFIKC 2 6
  YGFIQC 2 6
SHARED RANGES for cluster 49
  EGFKTL 2 6
  NGFKTL 2 6
  GFKTLE 1 5
SHARED RANGES for cluster 65
  VGDDVE 2 6
  PGDDVE 2 6
  GDDVEF 1 5
...

```

Figure 3: Sample shared ranges. Grayed areas show the conserved regions.

An example of a motif model of length 10 is shown in Table 2. Given that sequence  $x^i = x_1^i x_2^i \dots x_W^i$  is the  $i^{th}$  motif

Table 2: Example of a Motif Model of length 10. To save page space, only five character probabilities among 20 amino acids are shown for each column.

Amino Acid	Position 1	Position 2	Position 3	...	Position 10
G	$P_{G,1}$	$P_{G,2}$	$P_{G,3}$	...	$P_{G,10}$
A	$P_{A,1}$	$P_{A,2}$	$P_{A,3}$	...	$P_{A,10}$
L	$P_{L,1}$	$P_{L,2}$	$P_{L,3}$	...	$P_{L,10}$
M	$P_{M,1}$	$P_{M,2}$	$P_{M,3}$	...	$P_{M,10}$
...	...	...	...	...	...
T	$P_{T,1}$	$P_{T,2}$	$P_{T,3}$	...	$P_{T,10}$

with sequence length  $W$ ,  $P_{x_i j}$  represents the probability of  $j^{th}$  amino acid in sequence  $x_i$  occurring in the respective models.

## 2.5 Merge/Extend the conserved regions

The length of the subsequences used for clustering puts an upper limit on the length of the motifs found above. It is possible that the regions around the found motifs may also be conserved. We take care of such situation by extending the found motifs in both the directions. The entropy of the original motif model is compared to that of the motif model after extension and if there is an increase in the information content, we extend the motif in that direction. This technique is similar to one used in CASTOR[7] pattern discovery program.

Let  $E_{motif}$  denote the log-odds score of the found motif, calculated from the motif model or profile  $M_{motif}$  of length  $W$  (see Section 2.1.1 for exact definition). Similarly, let  $E_{new}$  represent the log-odds score of the model  $M_{motif+1}$  of length  $W + 1$  found after extending the motif model by one column. If  $\frac{E_{new}}{E_{motif}} > \theta$ , (where  $\theta$  denotes the threshold ratio), then we accept that column as part of the consensus region and  $M_{motif+1}$  forms the new motif model. This is repeated until there is no significant change in the information content.

It might be possible that two different motif models are overlapping or in close proximity to each other and we can merge such models. The strategy is to merge two motif models based on their correlation. The correlation between motif model  $A$  and  $B$  is given by -

$$\Gamma_{AB} = P_{AB}(d) - P_A P_B \quad (1)$$

where  $P_A$ ,  $P_B$  are the individual model probabilities and  $P_{AB}(d)$  is the probability of models  $A$  and  $B$  co-occurring at a distance  $d$  in the input set. So, if the correlation between two models is high, we merge them together to get a single motif model.

## 2.6 Ranking of Conserved Regions

Information content can be used to measure the degree of conservation at a site in a protein sequence alignment. The fewer the choice between occurrence of different residues at a site in the sequence, the more information it contains, and the more discriminatory it is for distinguishing real matches from random matches. The information content of the model can be expressed in terms of its entropy compared to that of the random model. The relative entropy  $H$  of a motif model  $M$ , as against the random model  $R$ , is

## Average number of false positives for different test cases

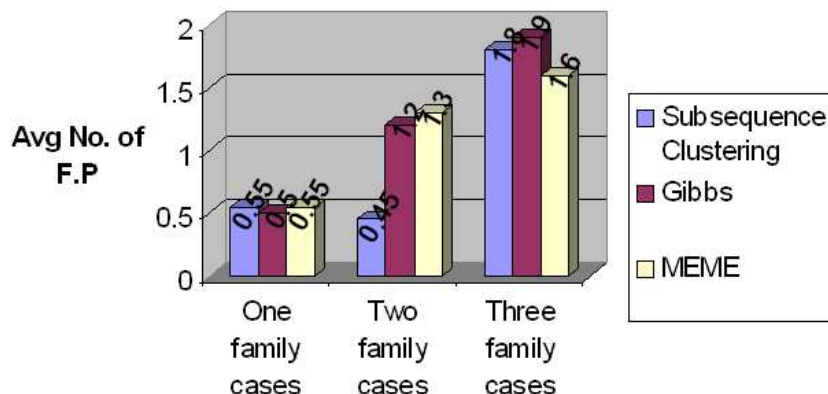


Figure 4: Average number of false-positives for various testcases

given by

$$H(M||R) = \sum_{j=1}^W \sum_{x \in AA} P_{x,j}^M \log\left(\frac{P_{x,j}^M}{P_{x,j}^R}\right) \quad (2)$$

where  $P_{x,j}^M$  and  $P_{x,j}^R$  are the probability of the  $j^{th}$  nucleotide in  $x_i$  occurring in motif model  $M$  and random model  $R$  respectively, and  $AA$  is the alphabet of 20 amino acids. This favors longer sequence alignment because the information content will be high for longer sequences due to the first summation in the equation.

Besides, the support value of the models is also important because the motif should be able to represent a large number of input sequences. The support value (or *quorum*) represents the number of sequences covered by the motif.

The models corresponding to the conserved regions, extracted from the shared ranges, are ranked based on their support value (or *quorum*). More the support value, higher is the rank of the motif. The models covering the same number of sequences are further ranked based on their relative entropy.

### 2.7 Filter and select motifs

We further filter the predicted motifs based on their support weight. In a heterogenous dataset, one protein family might have more sequences than another. In order to reduce the bias towards larger families, we assign weights to each sequence and calculate the weight for each motif using

$$W = \sum_i w_i \quad (3)$$

where  $W$  is the weight of the motif and  $w_i$  is the weight of sequence  $i$ , covered by the motif.

Initially all the sequences have similar weight(= 1). The support weight is calculated for each motif, starting from the top-ranked motif. If a sequence is covered by a motif, its weight is reduced by half. So the subsequent motifs covering the same sequences will have considerably lesser weight. We discard motifs whose support weight  $W < 0.25N$  where  $N$  is

the number of sequences in the input set. Filtering based on support weight helps to greatly reduce the number of false positives when there is a high representation from one of the protein families. The motifs that remain are the actual motifs found by the algorithm.

## 3. EXPERIMENTS

In order to evaluate the correctness and efficiency of our proposed algorithm, the algorithm was applied on the collections of various PROSITE[2] protein families.

We consider three different scenarios for our experiments: Test family contains sequences from -

1. *single* protein family.
2. *two* different protein families.
3. *three* different protein families.

The protein families to be included in the test set is chosen randomly. For each of the testfamily, we try to find motifs using our subsequence clustering algorithm. To compare our algorithm against other established motif-discovery algorithms, we use Gibbs motif sampling algorithm[6] and MEME[1] on the same test set. Standard parameters are used for both gibbs and MEME; the number of motifs specified during the runs was 1,2 and 3 for one family, two family and three family scenarios respectively. Additional parameters for Gibbs and MEME -

1. Length of each motif = 15

Following parameter values were used in our algorithm -

1. Length of subsequences,  $l = 10$
2. Threshold for finding subsequences,  $T = 0.01$
3. Support ratio for finding subsequences,  $K = 0.05$
4. Cutoff value for clustering algorithm,  $C = 100$

### One family case – Subsequence Clustering



### One family case – Gibbs

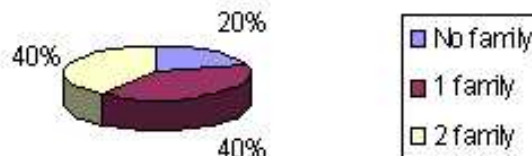


### One family case – MEME

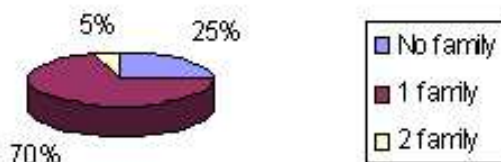


Figure 5: Number of families identified by (a) Subsequence Clustering approach, (b) Gibbs and (c) MEME for single-family testcases

### Two family case – Subsequence Clustering



### Two family case – Gibbs



### Two family case – MEME

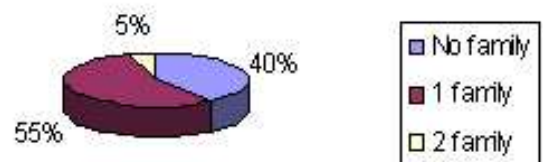
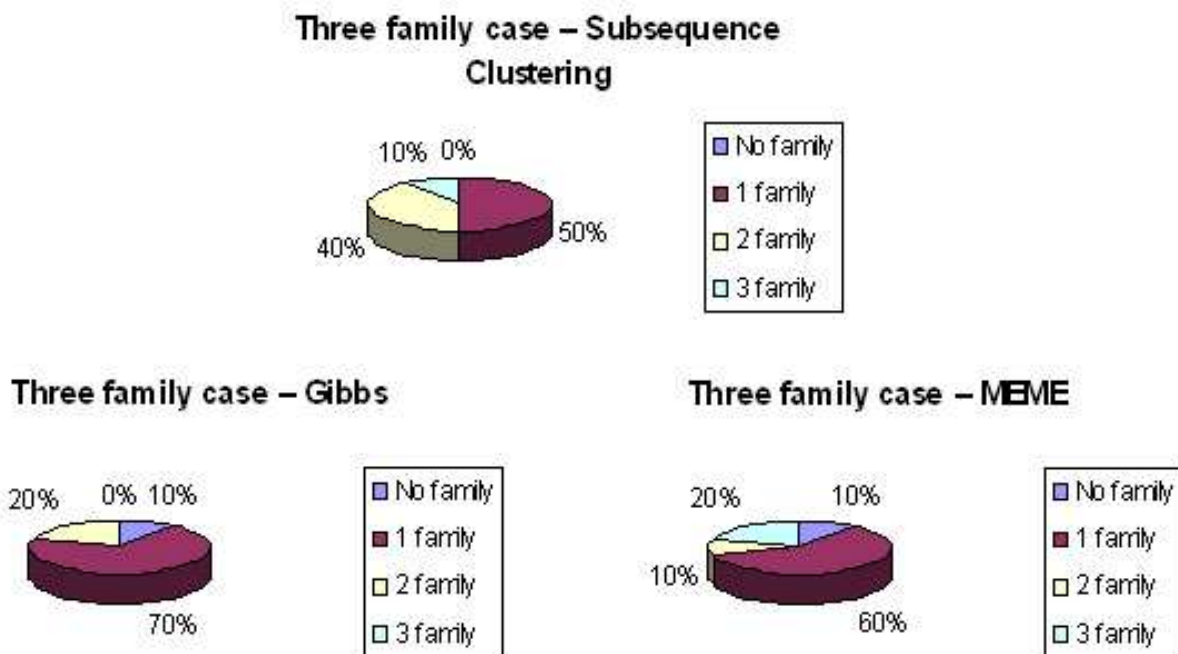


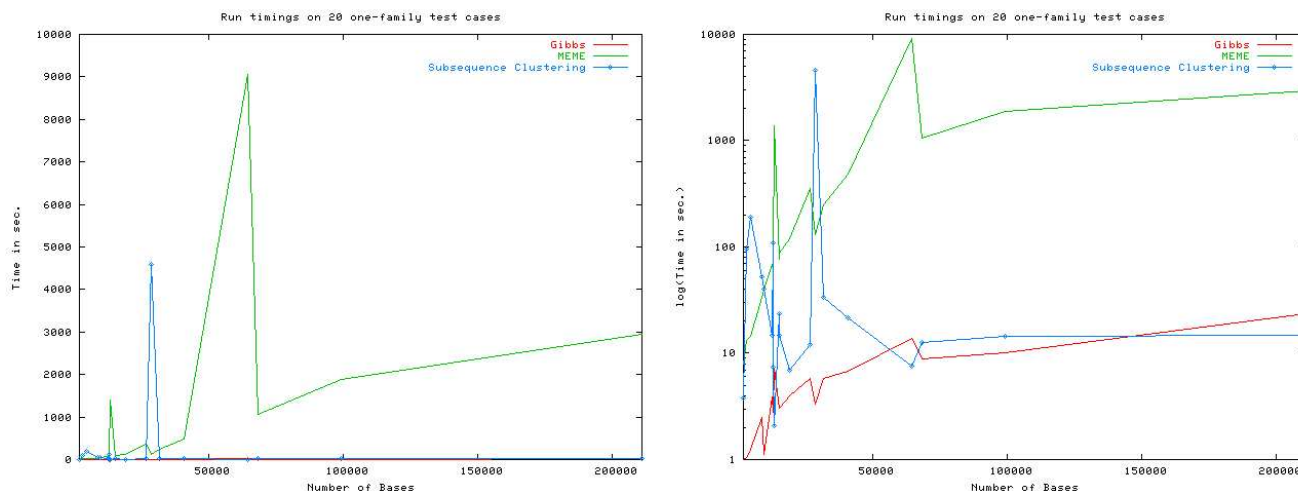
Figure 6: Number of families identified by (a) Subsequence Clustering approach, (b) Gibbs and (c) MEME for two-family testcases

5. Minimum overlap length for clustering algorithm,  $L = 6$

We consider the motif found to be the actual motif if the predicted motif covers at least half of the PROSITE regular



**Figure 7: Number of families identified by (a) Subsequence Clustering approach, (b) Gibbs and (c) MEME for three-family testcases**



**Figure 8: (a) Linear and (b) Semi-log plot of run timings for 20 one-family testcases based on the number of bases**

expression. The results of our experiments for one family, two family and three family scenarios is shown in Figure 5, Figure 6 and Figure 7 respectively. We also compare the time taken by each algorithm to find motifs in the data set. The comparisons are shown in Figure 8 through Figure 10. The average number of false positives reported by the three algorithms is plotted in Figure 4.

#### 4. DISCUSSION

As is evident from the graphs, the outcome of the three

algorithms is comparable in case of single-family test cases. However, as the heterogeneity of the input set increases, results start showing a great deal of variation. The subsequence clustering approach is successful in discovering motifs representing the different families in more cases than Gibbs and MEME algorithm. In two family test scenario, our algorithm found motifs representing both families in 40% of the test cases, whereas Gibbs and MEME succeeded in only 5% of the test cases. The trend was similar in three family scenario, where gibbs couldn't find motif for all the

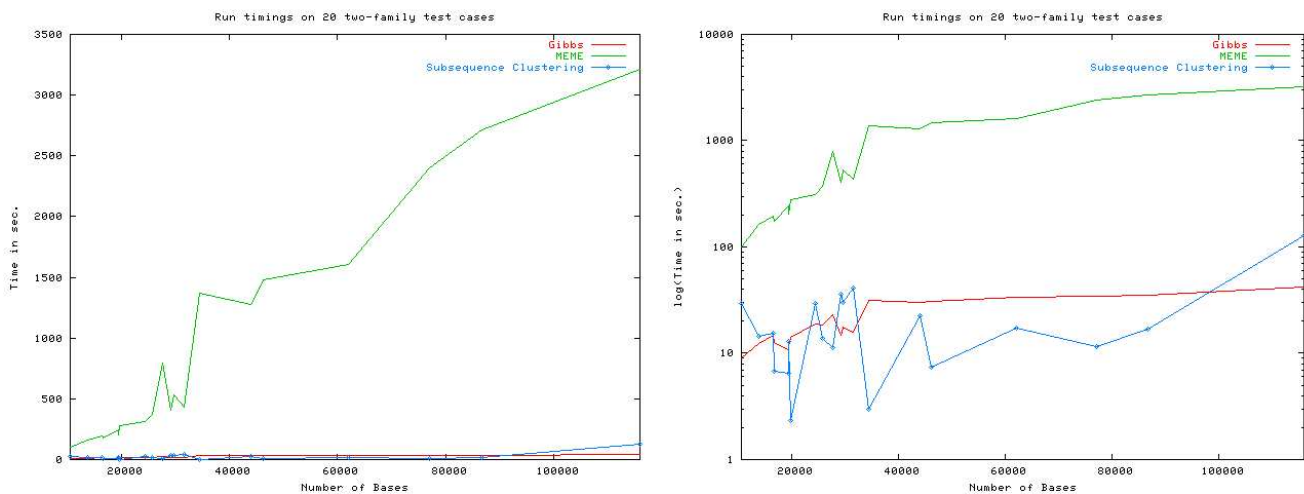


Figure 9: (a) Linear and (b) Semi-log plot of run timings for 20 two-family testcases based on the number of bases

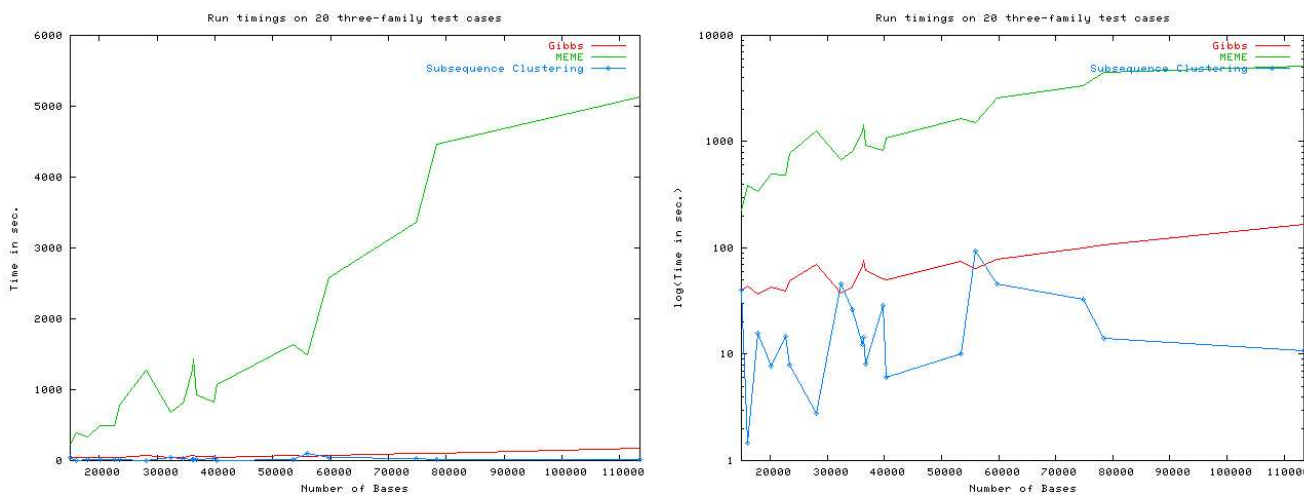


Figure 10: (a) Linear and (b) Semi-log plot of run timings for 20 three-family testcases based on the number of bases

three families in the input set in any of the test cases.

The average number of false positives produced by the three algorithms in different test cases is similar. However, we also emphasize that our method does not need to know the number of motifs expected. This is quite advantageous when there is no information available about the dataset.

As far as the runtime performance goes, our subsequence clustering algorithm works a lot faster than MEME. It is worth noting that subsequence clustering algorithm and gibbs were run on 4-processor Linux machine having 4GB of memory whereas MEME was run parallelly on 2 nodes (8-processor) of IBM Scalable POWERparallel System (SP). The performance of our algorithm was comparable to gibbs in one family case and a little better in two and three family cases.

## 5. CONCLUSION

A motif discovery algorithm by clustering subsequences has been presented. The performance of our motif algorithm was measured using sequence families with patterns

from PROSITE. The results of our algorithm were found to be comparable to those of gibbs and MEME motif algorithms and outperformed gibbs and MEME algorithms when input set contains non-homogeneous sequences, i.e., two family test cases. Our algorithm also outputs multiple conserved regions for the input sequence set without being instructed on how many motifs should be discovered. This is a very encouraging result and can prove to be very helpful in cases when the input sequence set consists of sequences from unknown protein families or from a collection of protein families. This is a significant advantage over existing motif discovery algorithms that essentially find only the specified number of motifs.

## 6. ACKNOWLEDGMENTS

We would like to thank Dr. Haixu Tang and Zhiping Wang for their support and helpful discussions on this project. This work is partially supported by NSF DBI-0237901, Outstanding Junior Faculty Award from Indiana University and

## 7. REFERENCES

- [1] T. Bailey and C. Elkan. Unsupervised learning of multiple motifs in biopolymers using em. *Machine Learning*, 21(1-2):51–80, October 1995.
- [2] L. Falquet, M. Pagni, P. Bucher, N. Hulo, C. Sigrist, K. Hofmann, and A. Bairoch. The prosite database, its status in 2002. *Nucl. Acids Res.*, 30:235–238, 2002.
- [3] I. Gunduz, S. Zhao, M. Dalkilic, and S. Kim. Motif discovery from large number of sequences: A case study with disease resistance genes in arabidopsos thaliana. *METMBS*, pages 29–34, 2003.
- [4] I. Jonassen. Efficient discovery of conserved patterns using a pattern graph. *CABIOS*, 13:509–522, 1997.
- [5] S. Kim. Graph theoretic sequence clustering algorithm and their applications to genome comparisons. *Computational Biology and Genome Informatics*, World Scientific, 2003.
- [6] C. Lawrence, S. Altschul, M. Bogouski, J. Liu, A. Neuwald, and J. Wooten. Detecting subtle sequence signals: A gibbs sampling strategy for multiple alignment. *Science*, 262:208–214, 1993.
- [7] A. H. Liu and C. Andrea. Castor: Clustering algorithm for sequence taxonomical organization and relationships. *Journal of Computational Biology*, 10(1):21–46, 2003.
- [8] W. R. Pearson and D. J. Lipman. Improved tools for biological sequence comparison. *PNAS*, 85:2444–2448, 1998.
- [9] I. Rigoutsos and A. Floratos. Combinatorial pattern discovery in biological sequences: The teiresias algorithm. *Bioinformatics*, 14:55–67, 1998.
- [10] G. Thijs, M. Lescot, K. Marchal, S. Rombauts, B. D. Moor, P. Rouz, and Y. Moreau. A higher-order background model improves the detection of promoter regulatory elements by gibbs sampling. *Bioinformatics*, 17:1113–1122, 2001.
- [11] Z. Wang, M. Dalkilic, and S. Kim. Guiding motif discovery by iterative pattern refinement. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 162–166, New York, NY, USA, 2004. ACM Press.