Efficient Discovery of Structural Motifs from Protein Sequences with Combination of Flexible Intra- and Inter-block Gap Constraints

Chen-Ming Hsu[1], Chien-Yu Chen[2], Ching-Chi Hsu[3], and Baw-Jhiune Liu[1]

[1] Yuan Ze University, Department of Computer Science and Engineering, Chung-Li, Taiwan, 320, R.O.C.
{cmhsu, bjliu}@saturn.yzu.edu.tw
[2] National Taiwan University, Department of Bio-Industrial Mechatronics Engineering, Taipei, Taiwan, 106, R.O.C.
cychen@mars.csie.ntu.edu.tw
[3] Institute for Information Industry, Taipei, Taiwan, 106, R.O.C.
cchsu@iii.org.tw

Abstract. Discovering protein structural signatures directly from their primary information is a challenging task, because the residues associated with a functional motif are not necessarily clustered in one region of the sequence. This work proposes an algorithm that aims to discover conserved sequential blocks interleaved by large irregular gaps from a set of unaligned biological sequences. Different from the previous works that employ only one type of constraint on gap flexibility, we propose using combination of intra- and inter-block gap constraints to discover longer patterns with larger irregular gaps. The smaller flexible intra-block gap constraint is used to relax the restriction in local motif blocks but still keep them compact, and the larger flexible inter-block gap constraint is proposed to allow longer irregular gaps between compact motif blocks. Using two types of gap constraints for different purposes improves the efficiency of mining process while keeping high accuracy of mining results. The efficiency of the algorithm also helps to identify functional motifs that are conserved in only a small subset of the input sequences.

1 Introduction

Automatic discovery of patterns in unaligned biological sequences is an important problem in molecular biology. For a set of proteins that share a common function or structure, it is often that only a few of common residues are conserved among them [4]. In biology, a motif is a pattern that has a specific structure and is functionally significant [4]. Functional motifs are not necessarily found in only one region of the protein sequence. Instead, the conserved residues usually appear as clusters (it is called a motif block in this paper), and multiple clusters may simultaneously contribute to an important substructure [13]. Limited insertions and deletions are admitted within a motif block, and large insertions and deletions may happen between motif blocks during evolution.

Protein families can often be characterized by one or more such patterns, which each consist of one or more motif blocks [12, 18, 21, 22]. Many computational

W.K. Ng, M. Kitsuregawa, and J. Li (Eds.): PAKDD 2006, LNAI 3918, pp. 530 – 539, 2006.
© Springer-Verlag Berlin Heidelberg 2006

approaches have been introduced for the problem of motif identification [1, 2, 3, 6, 8, 11, 16, 17]. These approaches can be categorized based on the type of the motifs they discover, statistical or deterministic. In this paper, we focus on the problem of discovering deterministic patterns like some other web services, Pratt [6] and Teiresias [17]. A deterministic pattern can be matched or not matched by a sequence. In the mining process, a pattern is found if it matches more than a user-specified percentage of the input sequence set. This is the so-called minimum support constraint.

A sequential pattern is called sparse if a large number of wildcards exist between pattern components, and is treated as flexible, contrary to fixed, if different sequences match the same pattern with different sizes of gaps, where a gap is defined as a set of one or more successive wildcards. Discovering sparse and flexible patterns is a time-consuming task due to the large search space of solutions. So many related studies employ constraints to expedite the mining process [6, 14, 17, 20], among which the gap constraint is widely used to restrict the length of a fixed gap within some maximum and minimum values specified by the users. Jonassen *et al*. in 1995 first introduced the constraint of gap flexibility in Pratt program that allows limited variable spacing between pattern components [7]. Gaps of irregular lengths are important in biological patterns because variable sizes of loops can occur even in well-conserved regions. Setting flexibility as 2 satisfies most short patterns existing in protein sequences [7].

However, longer patterns consisting of several sequential blocks can be discovered only when a larger flexibility is allowed. According to our performance analysis on Pratt program, version 2.1 [6], we observed that the program consumes unreasonably much time when flexibility is set to a value larger than 4, as shown in Table 1. This result is because the branching factor of Pratt used in constructing the pattern tree is exponentially in proportion to the flexibility constraint. Pratt uses some other constraints, such as flexibility product, to narrow down the search space and to decrease the number of potential patterns generated. However, this largely reduces the solution space, and consequently longer patterns cannot be discovered.

Another common problem of current mining algorithms is a huge amount of memory is required for constructing a pattern tree and the associated data structures during mining process. Table 1 also shows the memory usage of Pratt v2.1 versus the flexibility constraint. This situation is getting even worse when lower support constraint is requested. Nevertheless, low supports are desired during mining process because some highly specific signatures are usually conserved in few members of a protein family.

Table 1. Performance analysis of Pratt v2.1 on a data set containing about one hundred of sequences with the support constraint set as 70%

Flexibility	Flexibility product	Memory used	Execution time
FL=2	FP=16	0.182 Gigabytes	2895 seconds
FL=3	FP=81	1.016 Gigabytes	36963 seconds
FL=4	FP=256	1.5 Gigabytes	207236 seconds
FL=5	FP=625	3.9 Gigabytes	The system crashed

This work presents the algorithm MAGIIC that aims to discover flexible long patterns from a set of unaligned biological sequences. We propose using combination of intra- and inter-block gap constraints to find patterns with large irregular gaps, provided that the derived patterns are still compact in local regions. The idea is motivated by the observation that highly conserved regions of biological sequences are usually separated by a set of large gaps with irregular lengths. The smaller flexible intra-block gap constraint is used to relax the local motif blocks but still keep them compact, and the larger inter-block gap constraint is proposed to allow longer gaps to exist between compact motif blocks. Using two types of gap constraints for different purposes largely improves the efficiency of the mining process.

2 Problem Definition

In this section, the problem of mining conserved sequential blocks with flexible intra- and inter-block gaps is defined. We first give the definition of a sequence.

Definition 1 (Sequence). A sequence over an alphabet Σ is a finite sequence of components belonging to Σ. For any sequence $\beta=\langle\beta_1\ldots\beta_m\rangle$, a sequence α is called a subsequence of β, denoted as $\alpha <_s \beta$, if α can be obtained by deleting zero or more components from sequence β. We use $\beta[i..j]$ to denote the substring (contiguous subsequence) of β, which starts at position i and ends at position j of β, for $1 \le i \le j \le m$. In particular, $\beta[1..i]$ is the prefix of sequence β that ends at position i, and $\beta[i..m]$ is the suffix of sequence β that begins at position i. The number of components in β is denoted as $|\beta|$. □

If we segment a sequence into one or more blocks, it can be expressed as a blocked sequence. Blocks belonging to the same sequence are called sequential blocks.

Definition 2 (Blocked sequence). A sequence $\alpha = \langle\alpha_1\ldots\alpha_m\rangle$ can be segmented into disjoint r blocks, $r \le m$, and be written as $\langle B_1\ldots B_r\rangle$, where $B_k = \alpha[e_{k-1}+1..e_k]$, $e_0 = 0$, $e_r = m$, and $e_k > e_{k-1}+1$, for $1 \le k \le r$. □

We next define what intra- and inter-block gaps are.

Definition 3 (Intra- and inter-block gaps). Let $\beta = \langle\beta_1\ldots\beta_m\rangle$ be a sequence, and $\alpha=\langle B_1\ldots B_r\rangle$ be a blocked sequence provided that $\alpha <_s \beta$. If we consider the blocked sequence α as a pattern, then β serves as an instance of α. The interval between any two adjacent blocks B_i and B_{i+1} on the sequence β is called an *inter-block gap*. The interval between any two adjacent components within a block of α on the instance β is called an *intra-block gap*. □

MAGIIC employs different constraints for intra- and inter-block gaps respectively.

Definition 4 (Gap constraints). Let $\omega = (\gamma_{\min}, \gamma_{\max}, \tau_{\min}, \tau_{\max})$ be a set of constraints called gap constraints, which stand for the low and up bounds of an intra-block gap and the low and up bounds of an inter-block gap, respectively. Given a blocked sequence $\alpha=\langle B_1\ldots B_r\rangle$, we say that α satisfies the user-defined gap constraints ω if there exists a sequence $\beta=\langle\beta_1\ldots\beta_m\rangle$ such that α holds as a subsequence of β and the

Table 2. Parameters of MAGIIC

Parameter set θ		Description
λ		Minimum occurrences of a pattern
$\kappa_{\min}$		Minimum size of a motif block
$\kappa_{\max}$		Maximum size of a motif block
ω	$\gamma_{\min}$	Low bound of an intra-block gap
	$\gamma_{\max}$	Up bound of an intra-block gap
	$\tau_{\min}$	Low bound of an inter-block gap
	$\tau_{\max}$	Up bound of an inter-block gap
$n_{\min}$		Minimum size of a pattern
$n_{\max}$		Maximum size of a pattern

blocks in α satisfy the constraint ω, denoted as $\alpha <_{\omega} \beta$. The set $\beta/(\alpha)_{\omega}$ stands for all the substrings of β that match pattern α under the gap constraints ω. □

MAGIIC also employs some other basic constraints associated with pattern mining, including the support and size constraints. The parameter names are given in Table 2.

Definition 5 (Support and size constraints). The support of a blocked sequence α in a sequence database D under the gap constraints ω and the size constraints ($\kappa_{\min}$, $\kappa_{\max}$, n_{min}, n_{max}) is defined as the number of distinct input sequences $\beta \in D$ such that $\alpha <_{\omega} \beta$ and the blocked sequence $\alpha = \langle B_1 \ldots B_r \rangle$ satisfies $n_{min} \le |\alpha| \le n_{max}$, and $\kappa_{\min} \le |B_i| \le \kappa_{\max}$ for $1 \le i \le r$. The pattern α is frequent (conserved) in sequence database D if its support is grater than λ, where λ is the minimum support constraint. □

Finally we give the problem statement as follows.

Problem Statement. Given a sequence database D and the parameter set θ listed in Table 2, the algorithm will find the complete set of conserved blocked sequences (patterns) existing in the sequence database D under the constraints in θ. □

The derived patterns are expressed in the PROSITE language [5]. The notation x(a,b), $a < b$, is used for a size-bounded gap with minimum length of a and maximum length of b, and x(a) stands for a gap with a fixed length of a. The wildcard x(a) is omitted if $a = 0$, and is written as x if $a = 1$, i.e. x = x(1).

3 Method

This paper proposes a novel algorithm called MAGIIC, which is designed based on the PrefixSpan algorithm proposed by Pei *et. al.* in 2004 [15]. The contribution of MAGIIC comes from two parts. First, MAGIIC develops a new procedure called *bounded-prefix-growth* based on the *prefix-growth* procedure of the PrefixSpan algorithm. In order to identify patterns with large flexible gaps in biological data, the *bounded-prefix-growth* procedure incorporates intra- and inter-block gap constraints to speed up the mining process. Second, MAGIIC employs a newly designed projected database, called complete projected database, to guarantee that all the patterns satisfying the user-specified gap constraints will be found. In this section, we first briefly describe the PrefixSpan algorithm. After that, the concept of a complete projected database will be defined and how the intra- and inter-block gap constraints affect the scanning process of *bounded-prefix-growth* will be introduced.

PrefixSpan algorithm presents as a promising and efficient approach for many applications of sequential pattern discovery by avoiding generating a large amount of pattern candidates, and it consumes a relatively stable and small amount of memory space by using the pseudo projecting technique that records only the sequence identifiers and the associated event identifiers instead of constructing a physical projected database. The *prefix-growth* procedure of PrefixSpan employs a divided-and-conquer mechanism for pattern growing, which recursively reduces the size of the sequence database by generating the projected database of the current sequential pattern and then grows the sequential pattern in one particular projected database by exploring the local frequent components.

Fig.1 provides an example of a projected database with respect to a pattern ⟨CG⟩. Fig.1(a) shows the original database *D*. The projected database addressed by the PrefixSpan algorithm does not record complete information regarding gaps between sequence components, because PrefixSpan does not consider gap constraints in its mining process. As the example shown in Fig.1(b), the ⟨CG⟩'s projected database only keeps the longest substring of each sequence in *D* whose prefix matches the pattern ⟨CG⟩. This information is not sufficient when gaps are considered in the pattern mining process. Thus as shown in Fig.1(c), a complete projected database collects all the substrings in database *D* with a prefix of pattern ⟨CG⟩ that satisfies the gap constraints. We next give the definition of a complete projected database.

Seq id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
S_1	**C**	T	**G**	E	Y	T	J	E	A	S	N	**C**	A	**G**	E	**G**
S_2	P	E	**C**	P	**G**	K	I	I	**C**	H	P	**G**	Q	**G**	R	K
S_3	S	C	W	V	S	Q	W	V	V	**C**	Q	**G**	W	**G**		

(a) The original database *D*

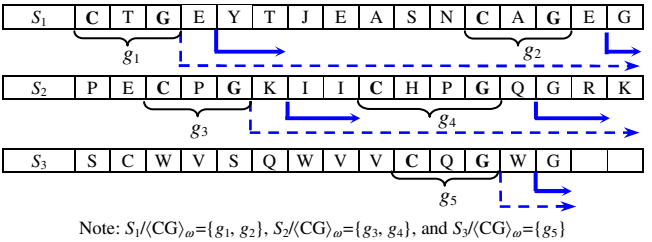
Seq id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
s_1	**C**	T	**G**	E	Y	T	J	E	A	S	N	C	A	G	E	G
s_2	**C**	P	**G**	K	I	I	C	H	P	G	Q	G	R	K		
s_3	**C**	Q	**G**	W	G											

(b) The projected database of the pattern ⟨CG⟩, according to the definition of PrefixSpan

Seq id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$s_{1,1}$	**C**	T	**G**	E	Y	T	J	E	A	S	N	C	A	G	E	G
$s_{1,2}$	**C**	A	**G**	E	G											
$s_{2,1}$	**C**	P	**G**	K	I	I	C	H	P	G	Q	G	R	K		
$s_{2,2}$	**C**	H	P	**G**	Q	G	R	K								
s_3	**C**	Q	**G**	W	G											

(c) The complete projected database of the pattern ⟨CG⟩

Fig. 1. Illustration of the complete projected database



Note: $S_1/\langle CG\rangle_\omega=\{g_1, g_2\}$, $S_2/\langle CG\rangle_\omega=\{g_3, g_4\}$, and $S_3/\langle CG\rangle_\omega=\{g_5\}$

Fig. 2. The scenario of scanning a projected database

Definition 6 (Complete projected database). Let α be a blocked sequence, and ω be a set of gap constraints. The α's complete projected database constructed from projecting the database D under gap constraints ω, denoted as $D|(\alpha)_\omega$, is a complete collection of sequences, each of which is the suffix of a sequence $\beta \in D$ and has a prefix of s, provided that $s \in \beta/(\alpha)_\omega$. □

We next describe how the *bounded-prefix-growth* procedure scans a complete projected database. The proposed procedure is called *bounded-prefix-growth* because its scanning range in the projected database is restricted by the gap constraints. In Fig.2, the dotted arrows represent the original scanning range of a projected database in PrefixSpan algorithm and the solid arrows show the scanning range of the *bounded-prefix-growth* procedure under the intra-block gap constraints ($\gamma_{min} = 1$, $\gamma_{max} = 2$). The scanning range of a complete projected database under gap constraints ω is much smaller. Most of the times, only the ranges under the intra-block gap constraints are considered when looking for the next frequent component. Only when the size of the currently growing block (the right most block) of the pattern satisfies the minimum block size constraint, larger scanning range with respect to the inter-block constraints will also be considered during the mining process.

The arguments of *bounded-prefix-growth* include a pattern as a blocked sequence α and its complete projected database $D|(\alpha)_\omega$. This procedure takes the blocked pattern α as input and tries to extend it under the user-specified constraints ω. In each call of *bounded-prefix-growth*, the search space of finding the next frequent component is bounded by the intra- and inter-block gap constraints. A component is conserved (frequent) if its occurrences in the projected database $D|(\alpha)_\omega$ satisfy the minimum support threshold. Each frequent component is appended to the current blocked sequence one at a time, and the resulted new blocked sequence (α') is used as the argument for the next call of *bounded-prefix-growth*, accompanied with a smaller projected database $D|(\alpha')_\omega$. Adding one more component to the current blocked pattern thus reduces the size of the complete projected database.

4 Results and Discussions

The performance of MAGIIC is compared with two well known packages on this problem, Teiresias [17] and Pratt v2.1 [6]. All the experiments provided here were conducted on a machine with a 3GHz Intel Pentium CPU and memory of 2GBs, running Linux Server. Regarding the parameter setting of MAGIIC, the users can set the following three constraints as a large number as long as the consuming time is acceptable on their machines. In this paper, we set both the maximum size of a motif block (κ_{max}) and the maximum size of a pattern (n_{max}) as 100, and change the up bound of an inter-block gap (τ_{max}) incrementally during mining process because this parameter affects the performance of MAGIIC significantly. Furthermore, in order to reduce the confusion of setting the other parameters, we set the low bound of an inter-block gap (τ_{min}) just one larger than the up bound of an intra-block gap (γ_{max}). Since only limited insertions and deletions within motif blocks are allowed during evolution, we set the low/up bound of an intra-block gap ($\gamma_{min}/\gamma_{max}$) as 0/2.

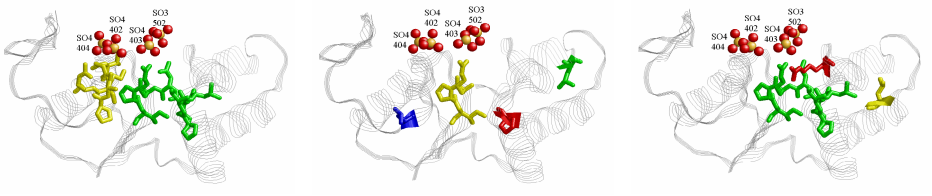
Table 3. Study on the first data set. (Arsenate reductase and related)

(a) Patterns discovered by different algorithms

ID	Patterns discovered by different algorithms	Support
	Pattern discovered by MAGIIC	
(1)	P-x-C-x(0,2)-S-x(0,2)-R-x(72,75)-P-x(1,2)-L-x(1,2)-R-P-I	38
	Pattern discovered by Teiresias (K=70, L=6, W=100, other parameters as default,)	
(i)	L-x(19)-P-x(4)-RPI-x(19)-L	38
	Pattern discovered by Pratt v2.1 (C%=70, PX=100, FN=4, FL=5, FP=12, other parameters as default)	
(ii)	R-x(18,20)-L-x(7)-P-x-L-x(2)-R-P-I	35
(iii)	G-x-[DEST]-x(2)-[AI]-x(2)-R-x(0,1)-K-x(4,7)-L-[ADGN]-[ILMV]-[ADEN]-x-[DEGN]-x-[FILM]-[PST]-x(3)-[FL]-x(2)-[FILM]-[IMV]-x(3)-P-x-[ILM]-[IL]-x-[RS]-P-I-[ILMV]-x-[DT]	35

(b) Executing time and usage of memory space for each algorithm with support = 70%

Method	MAGIIC	Teiresias	Pratt v2.1
Runtime in seconds	2	15	588
Memory used in Megabytes	3	15	150



(a) Pattern (1) found by MAGIIC (b) Pattern (i) found by Teiresias (c) Pattern (ii) found by Pratt v2.1

Fig. 4. Viewing the derived patterns of the first data set in a three-dimensional structure (1I9D.pdb). The patterns are plotted in *sticks*, and blocks are shown with different colors.

This paper employs two well annotated data sets to demonstrate the capability of MAGIIC algorithm in identifying conserved structural motifs. The first input set collects 50 proteins of the InterPro family IPR006660, Arsenate reductase and related, (one fragment has been removed) from Swiss-Prot (http://www.expasy.org/sprot/) (version 48.1). With the support constraint set as 70%, the minimum size of a motif block as 4, and the up bound of an inter-block gap constraint as 100, MAGIIC found one pattern with 38 supporting sequences, as shown in Table 3(a), denoted as pattern (1). In fact, this pattern can be found as long as the user-specified constraints are more relaxed than the above settings. It can be observed in Fig. 4(a) that the two motif blocks of pattern (1) are clustered together when the protein is folded, and they are really closed to the ligands bound together with this protein. This shows that the derived motif reveals its structural and functional meanings. It has been reported in [9] that the cysteine in the first block is important in binding one of the sulfate anion (SO4). We will show in the following that it is not found by other mining programs.

Table 3(a) also provides the best pattern found by Teiresias and Pratt v2.1, labeled as pattern (i)-(iii), which are derived by using reasonable parameter settings regarding this data set. It can be observed in Fig. 4(b) and (c) that patterns (i) and (ii) do not capture the signature of the substructure with respect to the ligands bound with this protein. The main problem of Teiresias is only fixed gaps are considered. Even Pratt

considers the flexibility of gaps, allowing large flexibility on every gap enlarges the search space and increases the executing time rapidly, which forces the users to give up searching for a pattern with a large gap like x(72,75). On the other hand, both Teiresias and Pratt v2.1 can handle equivalence to extend the length of the patterns, such as the pattern (iii) identified by Pratt v2.1. However, it does not really help to identify the structural motif associated with the functional site. Table 3(b) provides the executing time and memory usage of each algorithm. It is always the case that lowering support constraint enlarges the search space and thus consumes more computing time and space. Incorporating two gap constraints also makes MAGIIC more efficient than the other two packages in finding some highly specific signatures which are conserved in few members of a protein family.

The second training data was retrieved from Swiss-Prot database (release 48.1) by querying the keywords IPR001305, PF00684, and PS00637, which are associated with the CXXCXGXG domain signature of DnaJ proteins. The keywords are the entry IDs of InterPro (release 11.0), Pfam (version 18.0) and PROSITE (release 19.11) databases, respectively. We randomly select 100 proteins from the retrieved 272 sequences (totally we have 275 sequences, but three short fragments have been excluded.) And later we will show that the other 172 proteins can be completely found by using the pattern derived by MAGIIC.

With the support constraint set as 70%, the minimum size of a motif block as 4, and the up bound of an inter-block gap constraint as 20, MAGIIC found a pattern with 72 supporting sequences, denoted as pattern (1) in Table 4(a). This pattern contains four repeats of the motif block {C-x(2)-C-x-G-x-G}, which is recognized as the DnaJ central cysteine-rich (CR) domain. The DnaJ CR domain consists of two zinc centers, each of which is composed of four conserved cysteines [10]. We can see in Fig. 5(a) that each pair of the blocks in pattern (1) forms a zinc binding site, where the first and fourth blocks contribute to the first one and the second and third blocks contribute to the second one. It has been studied in [19] that the second binding site is more important than the first one. By increasing the support constraint as 100% and relaxing the minimum size of a motif block as 3, MAGIIC found another pattern, listed in Table 4(a) as pattern (2), which contains only three motif blocks. We observed that some DnaJ proteins lost the first block of the CR domain during evolution and some other lost the last block. This is consistent with the observation in [19] that the second and third blocks are more important to the function of DnaJ proteins.

The pattern of the entry PS00637 in PROSITE is provided in Table 4(a), denoted as pattern (i). As shown in Fig. 5(b), this pattern does not capture the feature of the isolated cysteine-rich domain, same as the best patterns found by Pratt v2.1 and Teiresias, also provided in Table 4(a). The selectivity of the derived patterns is evaluated by employing the ScanProsite (http://www.expasy.org/tools/scanprosite/) web service to scan protein sequences in Swiss-Prot database (release 48.1). The results are shown in Table 4(b). The precision rate is defined as TP / (TP + FP) and the recall rate as TP/ (TP + FN), where TP is short for true positives, FP for false positives, and FN for false negatives. It can be observed that the patterns found by Teiresias are not as good as MAGIIC or Pratt, and the pattern (ii) found by Pratt is specific enough to recognize the DnaJ proteins. However, pattern (2) still does not correctly capture the structural motif of the cysteine-rich domain. Also, Pratt v2.1 consumes much more resources than MAGIIC and Teiresias, as shown in Table 4(c).

Table 4. Study on the second data set (the CXXCXGXG domain signature of DnaJ proteins)

(a) Patterns discovered by different algorithms

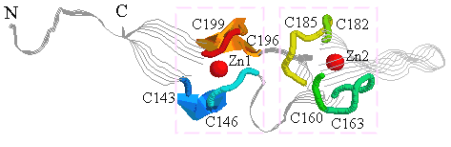
ID	**Pattern discovered by MAGIIC**	Support
(1)	C-x(2)-C-x-G-x-G-x(8,14)-C-x(2)-C-x-G-x-G-x(12,19)-C-x(2)-C-x-G-x-G-x(5,12)-C-x(2)-C-x-G-x-G	72
(2)	C-x(2)-C-x-G-x-G-x(8,19)-C-x(2)-C-x-G-x(7,20)-C-x(2)-C-x-G	100
ID	**Pattern of PS00637 in PROSITE database**	
(i)	C-[DEGSTHKR]-x-C-x-G-x-[GK]-[AGSDM]-x(2)-[GSNKR]-x(4,6)-C-x(2,3)-C-x-G-x-G	-
	Pattern discovered by Pratt (C%:70,PX:20,FL:8, FN:3, FP:256 and other parameters as default)	
(ii)	G-x(7,12)-C-x(2)-C-x-G-x-G-x(6,14)-C-x(2)-C-x-G	100
	Pattern discovered by Teiresias (K=70, L=4, W=30 and other parameters as default)	
(iii)	C-x(2)-C-x-G-x-G-x(6)-C-x(2)-C-x-G-x-G-x(12)-P-x(14)-G	70
(iv)	C-x(2)-C-x-G-x-G	100

(b) Analysis on the selectivity and sensitivity of patterns

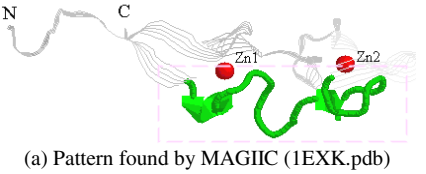
Method	ID	TP	FN	FP	Precision %	Recall %	Expected random matches
MAGIIC	(1)	219	53	0	100	80.50	4.342022e-14
MAGIIC	(2)	272	0	1	99.63	100	2.950594e-06
PROSITE	(i)	188	76	0	100	71.21	2.857160e-05
Pratt	(ii)	271	1	5	98.18	99.63	3.565674e-03
Teiresias	(iii)	223	49	0	100	81.98	2.180195e-07
Teiresias	(iv)	272	0	366	42.63	100	55

(c) Performance and usage of memory space with support = 70%

Method	MAGIIC	Teiresias	Pratt v2.1
Runtime in seconds	76	217	858760
Memory used in Megabytes	3	18	4000



(a) Pattern found by MAGIIC (1EXK.pdb)



(a) Pattern found by MAGIIC (1EXK.pdb)

Fig. 5. Structure of the patterns for the DnaJ proteins. The zinc atoms are plotted as red spheres and the patterns in colored cartoon display.

5 Conclusion

Functional motifs composed of several sequential blocks are difficult to find. Current mining technologies might individually find each motif block but fail to connect them with large irregular gaps. On the other hand, allowing large flexible gaps might derive patterns with the conserved residues largely scattered. MAGIIC employs intra- and inter-block gap constraints to discover clusters of conserved residues present in protein sequences. The efficiency of MAGIIC remains even when the constraints are relaxed. This is important because setting lower support constrains or larger gap flexibilities helps to identify the signature of protein functional sites. The spatial information of the sequential motifs also helps to detect critical substructures of proteins that share similar functions. Thus, how to incorporate MAGIIC in the study of protein binding or protein-protein interaction deserves further study.

References

1. Blanchette, M., Schwikowski, B., Tompa, M.: An exact algorithm to identify motifs in orthologous sequences from multiple species. Proc. Int. Conf. Intell. Syst. Mol. Biol. 8 (2000) 37-45
2. Blekas, K., Fotiadis, D.I., Likas, A.: Greedy mixture learning for multiple motif discovery in biological sequences. Bioinformatics. 19 (2003) 607-617
3. Brazma, A., Jonassen, I., Eidhammer, I., Gilbert, D.: Approaches to the automatic discovery of patterns in biosequences. J. Comput. Biol. 5 (1998) 277-305
4. Eidhammer, I., Jonassen, I. Taylor, W.R.: Protein Bioinformatics: An Algorithmic Approach to Sequence and Structure Analysis. John Wiley & Sons. (2004)
5. Falquet, L., et al.: The PROSITE database, its status in 2002. Nucl. Acids Res. 30 (2002) 235-238
6. Jonassen, I.: Efficient discovery of conserved patterns using a pattern graph. Comput. Appl. Biosci. 13 (1997) 509-522
7. Jonassen, I., Collins, J.F., Higgins, D.: Finding flexible patterns in unaligned protein sequences. Protein Science. 4(8) (1995) 1587-1595
8. Liu, X., Brutlag, D.L. Liu, J.S.: BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. Pac. Symp. Biocomput. (2001) 127-138
9. Martin, P., et al.: Insights into the Structure, Solvation, and Mechanism of ArsC Arsenate Reductase, a Novel Arsenic Detoxification Enzyme. Structure. 9 (2001) 1071-1081, 2001.
10. Martinez-Yamout, M., Legge, G.B., Zhang, O., Wright, P.E., Dyson, H.J.: Solution structure of the cysteine-rich domain of the Escherichia coli chaperone protein DnaJ. J. Mol. Biol. 300(4) (2000) 805-818
11. Narasimhan, G., Bu, C., Gao, Y., Wang, X., Xu, N., Mathee, K.: Mining protein sequences for motifs. J. Comput. Biol. 9 (2002) 707-720
12. Neuwald, A.F., Green, P.: Detecting patterns in protein sequences. J. Mol. Biol. 239 (1994) 698-712
13. Ogiwara, A., Uchiyama, I., Yasuhiko, S., Kanehisa, M.: Construction of a dictionary of sequence motifs that characterize groups of related proteins. Protein Eng. 5 (1992) 479-488
14. Pei, J., Han, J.: Constrained frequent pattern mining : a pattern-growth view. ACM SIGKDD Explorations (Special Issue on Constraints in Data Mining). 4(1) (2002) 31-39
15. Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U, Hsu, M.-C.: Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach. IEEE Transactions on Knowledge and Data Engineering. 16 (2004) 1424-1440
16. Pevzner, P.A, Sze, S.H.: Combinatorial approaches to finding subtle signals in DNA sequences. Proc. Int. Conf. Intell. Syst. Mol. Biol. 8 (2000) 269-278
17. Rigoutsos, I, Floratos, A.: Combinatorial pattern discovery in biological sequences: The Teiresias algorithm. Bioinformatics. 14 (1998) 55-67
18. Saqi, M.A.S, Sternberg, M.J.E.: Identification of sequence motifs from a set of proteins with related function. Protein Eng. 7 (1994) 165-171
19. Shi, Y.Y., Tang, W., Hao, S.F., Wang, C.C.: Contributions of cysteine residues in Zn2 to zinc figers and thioldisulfide oxidoreductase activities of chaperone DnaJ. Biochemistry. 44 (2005) 1683-1689
20. Silvestri, C., Orlando, S., Perego, R.: A new algorithm for gap constrained sequence mining. Proceedings of the 2004 ACM Symposium on Applied Computing, special track on Data Mining. (2004) 540-547
21. Su, Q.J., Lu, L., Saxonov, S., Brutlag, D.L.: eBLOCKs: enumerating conserved protein blocks to achieve maximal sensitivity and specificity. Nucl. Acids Res. 33 (2005) D178-182
22. Wang, J.T.L., et al.: Discovering active motifs in sets of related protein sequences and using them for classification. Nucl. Acids Res. 22 (1994) 2769-2775