DEVELOPING A MOTIF BASED CLUSTERING ALGORITHM FOR SUPPORTING THE QUERY IN DATABASE OF DNA SEQUENCES

Hoang Kiem, Do Phuc
University of Natural Sciences, HCMC
Department of Information Technology

(Received on February 20th, 2001)

ABSTRACT: *We have developed a system for supporting the query in a database of DNA sequences. We would like to develop a system for grouping similar DNA sequences into clusters based on frequent motifs or motif phrases. Each cluster is represented by a cluster feature vector of maximal frequent motifs or motif phrases). A motif tree of cluster features is built. The similarity search will be divided into two steps. Firstly, the system will search the clusters which have the high matching with the query pattern. Secondly, the traditional matching techniques (FASTA or BLAST) will be used for matching between pattern and a small number of DNA sequences of selected cluster.*

Keywords: association rule, DNA sequence, frequent motif, maximal frequent motif, clustering, motif phrase, motif tree.

1.Introduction

Now, there are many web-sites such as NCBI(http://www.ncbi.nlm.nih.gov/), EMLG(http://www.rwth-aachen.de/emlg/, SWISS-PROT(http://www.expasy.ch)... containing a large volume of DNA or protein sequences. Therefore, the problem of developing a system supporting the query in a large DNA or protein database is really an important problem. Pearson and Lipman in 1988 proposed a dynamic programming based on FASTA method [3] for searching the local high scoring alignment starting from exact short word matches. Altschul et al. in 1990 proposed a BLAST method [3] for searching high scoring local alignment between a query pattern and a target database. With a large database of DNA sequences containing more than 100,000 sequences and each sequence may have more than 10,000 bases, the time consuming is significant. We would like to use the clustering algorithm as pre-processing step for pattern matching. [5]. In this paper, we focus on developing a motif based algorithms for building a system supporting the query in a large DNA database of DNA sequences. The key idea of our method is to group similar DNA sequences into cluster based on the maximal frequent motifs or motif phrases. A motif tree of order 4 is built. Firstly, the system will search the pattern in motif tree for locating the clusters. Secondly, the traditional matching methods (FASTA or BLAST) will work with the small number of DNA sequences of selected clusters instead of the whole DNA database. The rest of paper is organised as follows 2) Discovering motifs in DNA sequences 3) A motif reduction 4) Motif graph and motif based clustering method 5) Motif based searching method 6) Building a motif tree for improving the speed of searching 7) A solution supporting the similarity search 8) A discussion of time complexity 9) Experiment results10) Conclusion and future works.

2. Discovering motifs in DNA sequences

2.1. Frequent motifs, maximal frequent motifs, motif phrases

Let $\mathcal{A}$= {"A", "C", "T", "G"} be the set of bases forming the DNA sequences, each base is a nucleotide. Each DNA sequence is considered as a text string $s_1,s_2, ...s_n$ where $s_k \in \mathcal{A}$, k=1,..,n. We denote by |s| the length of sequence s.

Definition 1: Frequent motifs

Let S be a set of m DNA sequences, $P(s_i)$ be the set of all sub sequences of s_i , PU be the union of all $P(s_i)$ for i=1,...,m, $N(s_t)$ be the number of DNA sequences containing s_t.

Given $s_t \in PU$ and a threshold $\tau \in [0,1]$, s_t is called a frequent motif if $N(s_t)/m >= \tau$.
We denote $FR(S,\tau) = \{s_t \in PU \mid N(s_t)/m >= \tau\}$. It is easy to see that if $s_a \in FR(S,\tau)$ and $s_t \in P(s_a)$ then $s_t \in FR(S, \tau)$.

Definition 2: Maximal frequent motifs

Let $a \in FR(S,\tau)$, a is called a maximal frequent motif, if there is no $b \in FR(S,\tau)$, $a \neq b$, such that $a \in P(b)$. We denote $MF(S,\tau)$ as the set of maximal frequent motifs of S with threshold τ.
During discovering the frequent motifs $FR(S,\tau)$, we also save their positions in DNA sequences. We use this positional information of motif to improve the quality of pattern matching.
We define function pos(a,i,j) to be the position of the j^{th} occurrence of motif a in sequence i., otherwise pos(a,i,j) =0 if there is no j^{th} occurrence of motif a in sequence i.

Definition 3: Motif phrases

A motif phrase is a couple of maximal frequent motifs denoted as <a, b> satisfying:

i) In i^{th} sequence, there are two positions u,v such that pos(b, i,u) > pos(a,i,v)

ii) There is no frequent motif c and a position w such that
$$pos(a,i,v) + |a| <= pos(c,i,w) < pos(b, i,u)$$

Example 1: Given $S = \{s_1, s_2, s_3\}$ containing three DNA sequences as follows:

s1 = "ACACCCCAC"
s2= "AAAGCCCGCACGGG"
s3= "ACATCCCTAAATGGG"

With τ= 0.66, we discover 13 frequent motifs:
FR(S, 0.66) = {"A"; "C"; "G"; "AA"; "AC"; "CA"; "CC"; "GG"; "AAA" ; "ACA"; "CAC"; "CCC"; "GGG"} and 5 maximal motifs:
MF(S,0.66) = { "AAA" ; "ACA"; "CAC"; "CCC"; "GGG"}
We build a matrix where rows and columns are elements of $MF(S,\tau)$, and use criteria (i) and (ii) for finding motif phrases.. With τ= 0.66, we discover motif phrases "ACA*CCC", "CCC*CAC" where * stands for an arbitrary sub-sequence of PU and * does not contain a frequent or maximal motif.We denote by $MFM(S,\tau)$ the set of maximal frequent motifs and motif phrases discovered from S. In the above example, $MFM(S,\tau)$ will be:
MFM(S, τ) = { "AAA" ; "ACA"; "CAC"; "CCC"; "GGG" ; "ACA*CCC" , "CCC*CAC"}

2.2. Problem statement and our proposed algorithm

Given a set of DNA sequences and a threshold $\tau \in [0,1]$, find all frequent motifs and maximal frequent motifs with threshold τ. We develop an algorithm [5] for finding frequent motifs and maximal frequent motifs in a set of DNA sequences. Let $L(S,k,\tau)$ be a set of all frequent motifs with the threshold τ and k is the length (number of bases) of these motifs. The proposed algorithm is as follows:

```
Answer =$\varnothing$
Generate $L(S,1,\tau)$ from   {"A", "C", "T", "G"}
 For ( k=2; $L(S,1,\tau)$ <> {} ; k++) do
   begin
      Generate $L(S,k,\tau)$ from $L(S,k-1,\tau)$
       Answer = $\cup_k L(S,k,\tau)$
   end
Return Answer
```

a) **Generate $L(S,1,\tau)$:** The one letter motifs are possible "A", "C", "T", "G", we check each of them, if it satisfies the definition of motif, save it into $L(S,1,\tau)$.

b) **Generate $L(S,k,\tau)$ from $L(S,k-1,\tau)$ and $L(S,1,\tau)$:** It is obvious that $s_a \in F(S,\tau)$ and $s_t \in P(s_a) \Rightarrow s_t \in F(S,\tau)$, we employ this proposition to generate $L(S,k,\tau)$ from $L(S,k-1,\tau)$ and $L(S,1,\tau)$.

The proposed algorithm is summarized as follows:

```
Create a  matrix which row and column  are $L(S,1,\tau)$.
$L(S,k,\tau) = \varnothing$
 For(each $s_y \in L(S,k-1, \tau)$) do
      For (each $s_x \in L(S,1,\tau)$ ) do    begin
              $s_t = s_y + s_x$ // string concatenation
              If ($N(s_t)/m$  >= $\tau$)  and |st|  ==  k )  then
                  SaveFreqMotif ($s_t$, $L(S,k,\tau)$)
      end;
   Answer = $L(S,k,\tau)$
  Return Answer
```

where SaveFreqMotif(s_t,$L(S,k,\tau)$) is the function for saving the frequent motif s_t into $L(S,k,\tau)$. From $FR(S,\tau)$, we use the definition of maximal frequent motifs for building $MF(S,\tau)$ and the definition motif phrase for building $MFM(S,\tau)$.

3. A motif reduction

Given m_a, m_b of $MFM(S, \tau)$, we build the association rules $m_a \rightarrow m_b$ and $m_b \rightarrow m_a$. The confidence of these rules are calculated by [2,6]:

$$CF(m_a, \rightarrow m_b)= |m_a \cap m_b| / |m_a|$$

$$CF(m_b \rightarrow m_a)= |m_b \cap m_a| / |m_b|$$

Where |x| is the number of DNA sequences containing motif x. Let $\rho(m_a)$ be the set of all DNA sequences containing m_a and $\rho(m_b)$ be the set of all DNA sequences contain m_b. It is easy to see that:

- $CF(m_a \rightarrow m_b) = 1$ if $\rho(m_a) \subseteq \rho(m_b)$
- $CF(m_b \rightarrow m_a) = 1$ if $\rho(m_b) \subseteq \rho(m_a)$
- $CF(m_a \rightarrow m_b) = 1$ and $CF(m_b \rightarrow m_a) = 1$ iif $\rho(m_a) = \rho(m_b)$

We reduce $MFM(S, \tau)$ by testing each couple of elements m_a, m_b of $MFM(S, \tau)$ as follows:
If $CF(m_a \rightarrow m_b) = 1$, $CF(m_b \rightarrow m_a) = 1$ and m_a is a sub sequence of m_b, we delete m_a
If $CF(m_a \rightarrow m_b) = 1$, $CF(m_b \rightarrow m_a) = 1$and m_b is a sub sequence of m_a then we delete m_b.
With $MFM(S, \tau)$ in example 1, we can reduce <C> because:

CF("C"→"CCC") = 1 and CF("CCC" → "C") = 1

We also reduce "ACA" because:

CF("ACA,CCC" →"ACA") = 1 and CF("ACA" → "ACA,CCC") = 1

We denote by $MFMR(S, \tau)$ the reduction of $MFM(S,\tau)$.

4. Motif graph and motif based clustering method

Given a set S of DNA sequences and a threshold τ, we discover $MFMR(S,\tau)$. A a motif graph $G(V,E)$ is created [4,7,10] where $V=MFMR(S,\tau)$ is the set of nodes representing a maximal frequent motif or motif phrase. Two nodes m_a, m_b are connected if $CF(m_a \rightarrow m_b) >= \alpha$ and $CF(m_b \rightarrow m_a) >= \alpha$ and α is a given threshold in [0,1]. The value α will determine the possibility of concurrent occurrence of motif m_a, m_b in a same DNA sequence. A motif cluster is defined as being a connected component in the motif cluster graph. Suppose that we have:
$MFMR(S,\tau)$ = { "CCC"; "CCC*CAC", "GGG"; :"CAC"; "ACA*CCC"; "AAA" }
The motif graph is shown in figure 1.

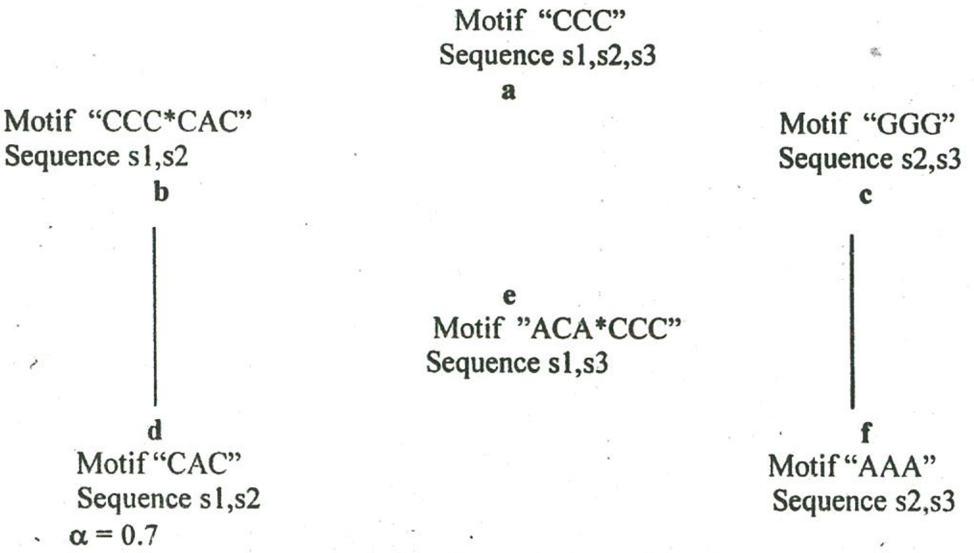

Figure 1: A motif graph

The value α reflects the degree of concurrent occurrence of maximal frequent motifs or motif phrases in DNA sequences. In the above graph, we choose α = 0.7. From the motif graph, we discover the connected components, each connected component is a cluster (group of similar DNA sequences). Each cluster is represented by a set of maximal frequent motifs or motif phrases and this set is called cluster feature. With threshold α = 0.7, we discover 4 connected components(4 clusters) as shown in table 1:

Table 1: The motif clusters

Cluster number	Cluster feature	Sequences
1	a	s1,s2,s3
2	e	s1,s3
3	b, d	s1,s2
4	c, f	s2,s3

5. Motif based searching method

- **Case a:** Find a query pattern p in S, we will choose clusters where p is an element of their cluster features.

Example 2: If query pattern p is **"CCC"**, cluster1 is selected because its cluster feature is also **"CCC"** and the sequences of this cluster are s1,s2,s3.

- **Case b:** Find a query pattern P and there are no cluster features containing P. we will select clusters where their cluster features containing as much as sub sequences of P.

Example 3: If query pattern p is **"CCCG"**, cluster1 is selected because its cluster feature is **"CCC"** which is highest matching with **"CCCG"**. The sequences of this cluster are s1,s2,s3.

- **Case c:** Find a set of query patterns called P, we will select clusters where P containing subset of their cluster feature.

Example 4: If query pattern P is {**"ACA*CCC"**} , cluster2 is selected because its cluster feature is also **"ACA*CCC"** and the sequences of this cluster are s1,s2,s3.

6. Building a motif tree for improving the speed of searching

For improving the speed of searching, we also build a motif tree. If set of cluster features is {"AAA", "AAG", "ACA","ACC","ATA"} we build up a cluster feature tree of order 4. A typical motif tree structure is shown in figure 2.

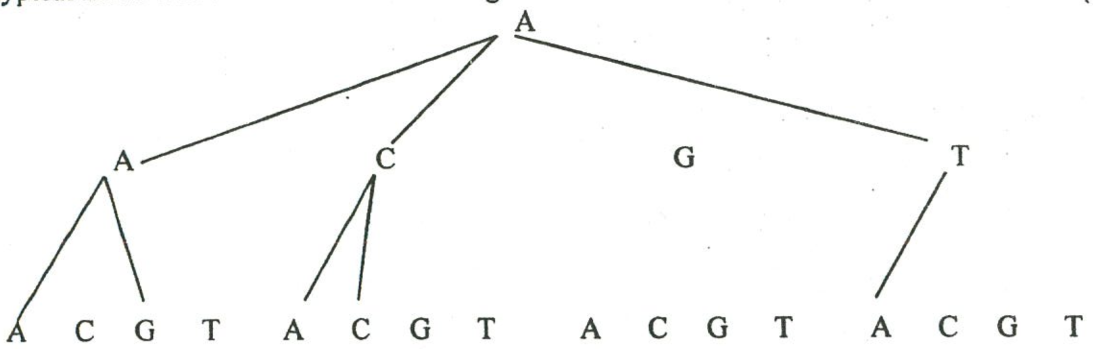

Figure 2: A motif tree order 4

In the first step of processing the query,. the system will search the pattern in the motif tree for locating the clusters which have highest matching with the pattern.

7. A solution supporting similarity search

As mentioned above, FASTA or BLAST techniques are used for searching the local high scoring alignment between query pattern and sequence of DNA database. When we receive a query pattern, we will use motif tree for selecting the clusters and finally we use FASTA or BLAST for matching between pattern and each sequence of selected cluster with a small number of DNA sequences instead of the whole DNA database.

8. A Discussion of Time Complexity

When we receive a query pattern, the searching is proceeded in the tree structure of order 4. The time complexity for searching is $O(\log_4(n))$ where n is the cardinality of $MFMR(S,\tau)$. If we use the FASTA or BLASTA method which based on the dynamic programming, the time complexity will be $O(mvk)$ where m is the number of DNA sequences, v is the length of query pattern and k is the length of DNA sequence. Therefore, our proposed method can improve the searching time.

9. Experimental Results

We test our proposed method on a set of DNA promoter sequence of UCI. repository.

9.1. Discovering frequent motifs, maximal frequent motifs, motif phrases

With threshold $\tau = 0.02$, we discovered 340 frequent motifs and 256 maximal frequent motifs.

Some of frequent motifs are listed are as follows:

A,C,T,G,AA,AC,AT,AG,CA,CC,CT,CG,TA,TC,TT,TG,GA,GC,GT,GG,AAA,AAC,AAT,
AAG,ACA, ACC, ACT,ACG,ATA,ATC,ATT,ATG,AGA,AGC,AGT,AGG,CAA,CAC,CAT,
CAG, CCA, CCC, CCT,CCG,CTA,CTC,CTT,CTG,CGA,CGC,CGT,CGG,TAA,TAC,TAT,

Some maximal frequent motifs are listed as follows:

ACTT,ACTG,ACGA,ACGC,ACGT,ACGG,ATAA,ATAC,ATAT,ATAG,ATCA,ATCC
ATCT,ATCG,ATTA,ATTC,ATTT,ATTG,ATGA,ATGC,ATGT,ATGG,AGAA,AGAC,
AGAT,AGAG,AGCA,AGCC,AGCT,AGCG,AGTA,AGTC,AGTT,AGTG,AGGA,AGGC,

Some of the motif phrases are as follows:.

AAAA*AAAT; AAAA*ACTA; AAAA*ACTT; AAAA*CACT; AAAA*CTTT

9.2. Motif reduction

We discover association rules with CF (a→b) >= 1.0 and CF (b→a)>= 1.0, a∈P(b) or b∈P(a). Some rules are as follows: **"A"→"AT", "A"→ "CA", "C→ "CA", "T"→"AT"**. Therefore, we can delete "A" ,"C", "T" from FR(S,0.02).

9.3. Motif graph and motif clusters

We build a motif graph with $\alpha = 0.61$, a portion of graph matrix is shown in table 2:

Table 2: A portion of motif graph with $\alpha = 0.61$

#	Node1	Node2	CF(Node1->Node2)	CF(Node2->Node1)
13	AAAG	GAAA	0.87	0.68
14	AACA	ACAT	0.65	0.79
15	AACG	CGCT	0.64	0.62
16	ACCG	CGGA	0.69	0.73
17	ATTT	TTTT	0.75	0.66

We discover the connected components in this graph. Some cluster features are listed as follows:

a. Cluster feature 1 : AAAC; CAAA

b. Cluster feature 2: AAAT; TAAA; TCTT; TTTC; TTTT; TTTG; GTTT

c. Cluster feature 3: AAAG; GAAA; GGTT

d. Cluster feature 4:AATG; ATGC; GCGC; CCTC; CTTG; TTGC; TTGT; TAAT;

f. Cluster feature 6:ACTG; ATGA; TATG; CTGT; TTAT; TGTG; GTGA

. . .

9.4. Similarity query in motif tree

When we want to search DNA sequences containing patterns AAAC we will focus on cluster 1 because this pattern is the subset of its cluster feature. DNA sequences of cluster 1 are shown in table 3.

Table 3: DNA sequences of Cluster 1

Seq#	DNA Sequence Promoter
4	AATTGTGATGTGTATCGAAGTGTGTTGCGGAGTAGATGTTAGAATACTAAC**AAAC**TC
8	TTTCTACA**AAAC**ACTTGATACTGTATGAGCATACAGTATAATTGCTTCAACAGAACA
18	AAACAATTTCAGAATAGACAA**AAAC**TCTGAGTGTAATAATGTAGCCTCGTGTCTTGC
21	GACACCATCGAATGGCGCA**AAAC**CTTTCGCGGTATGGCATGATAGCGCCCGGAAGAG
26	TTGTCATAATCGACTTGT**AAAC**CAAATTGAAAAGATTTAGGTTTACAAGTCTACACC
33	ATGCGCAACGCGGGGTGACAAGGGCGCGC**AAAC**CCTCTATACTGCGCGCCGAAGCTG
44	GCCTTCTCCA**AAAC**GTGTTTTTTGTTGTTAATTCGGTGTAGACTTGTAAACCTAAAT
49	GGCCAAAAAATATCTTGTACTATTTACA**AAAC**CTATGGTAACTCTTTAGGCATTCCT
51	CCATCAAAAAAATATTCTCAACATAAAA**AAAC**TTTGTGTAATACTTGTAACGCTACAT

10. Conclusion and future works

We have developed a system supporting the query in a large DNA sequence database. Firstly, we develop algorithm for discovering the frequent motifs, maximal frequent motifs, motif phrases. Secondly, we reduce set of frequent motifs, maximal frequent motifs, motif phrases. Then we group similar DNA sequences based on maximal frequent motif and motif phrases. A motif tree order 4 is created for improving the speed of searching. We also present the experimental results on a DNA promoter sequence data set of UCI and analyse the time

complexity of our proposed algorithm. We also use the same method for Protein Database in which the alphabet l has 20 amino acids instead of 4 nucleotides of DNA sequences. We continue to develop incremental algorithms for discovering frequent motifs, maximal frequent motifs and motif- based clusters in upgraded DNA or protein database.

PHÁT TRIỂN THUẬT TOÁN GOM NHÓM DỰA TRÊN ĐOẠN LẶP HỖ TRỢ TRUY VẤN THÔNG TIN TRONG CSDL CHỨA CÁC TRÌNH TỰ SINH HỌC ADN

Hoàng Kiếm, Đỗ Phúc
Đại học Khoa học Tự nhiên, TP. Hồ Chí Minh

TÓM TẮT: *Chúng tôi phát triển thuật toán hỗ trợ truy vấn thông tin trong CSDL chứa các trình tự DNA. Chúng tôi gom nhóm các trình tự DNA tương tư nhau thành các cluster dựa trên các đoạn lặp hay nhóm các đoạn lặp. Mỗi cluster được biểu diễn bằng vector đặc trưng cho cluster có thành phần là các đoạn lặp phổ biến tối đại hay nhóm các đoạn lặp phổ biến tối đại. Chúng tôi tạo cây chứa các đoạn lặp. Tiến trình tìm kiếm tương tự gồm hai bước. Đầu tiên, hệ thống sẽ tìm cluster so khớp tốt nhất với mẫu truy vấn. Sau đó, các kỹ thuật so khớp truyền thống như FASTA hay BLAST sẽ được dùng để so khớp các trình tự của cluster được chọn.*

REFERENCES

[1] A Chouchoulas and Q. Shen, A Rough Set based approach to text classification, In Proceedings of RFDGRC'99 international conference, Yamaguchi-UBE, Japan, (1999).

[2] Charu C. Aggarwal, Philip S. Yu, Mining Large Item sets for Association rules, Bulletin of the Technical Committee on Data Engineering, Vol. 21, No.1, March, (1998)

[3] R. Durbinans Anders Krogh: Biological sequence analysis: probabilistic models of proteins and nucleotide acids, Cambridge University Press, (1998)

[4] Eui-Hong Ha, George Harypis: Hyper-graph based Clustering in high-dimensional data sets: a summary of results, Bulletin of the technical Committee on data engineering, March 1998, Vol.21, No.1, IEEE computer society, (1998).

[5] Hoang Kiem, Do Phuc: Developing the motif based algorithm for discovering the knowledge in a set of DNA sequences:, in Proceedings of the international conference SCI'2000, Florida, USA, (2000).

[6] Hoang Kiem, Do Phuc: A method for discovering binary and fuzzy association rules from database, In Proceedings of the international conference AFSS2000, Tsukuba, Japan, (2000).

[7] Oren Eli Zamir: Clustering Web Documents: A phrase based method for grouping searching engine result: Ph. D thesis, University of Washington, USA (1999).

[8] Roderic Guigo: Computational gene identification: an open problem, Computer Chem, Vol. 21, No.4, Elsevier Science, Great Britain, (1997).

[9] R. Agrawal , R. Srikant, Fast Algorithm for Mining Association Rules in large database, Research report RJ, IBM Almaden Research Center, San Jose, CA ,(1994).

[10] Ramkumar G.D. and Arun Swanmi: Clustering Data Without Distance Functions: Bulletin of the Technical Committee on Data Engineering, March 1998, Vol. 21, No.1, IEEE computer society, (1998).