International Journal of Computer Applications (0975 – 8887)
Volume 29– No.5, September 2011

A DNA based Approach to find Closed Repetitive Gapped Subsequences from a Sequence Database

B.Lavanya
Department of Computer Science
University of Madras
Chennai 600 005, INDIA

A.Murugan
Department of Computer Science
Dr.Ambedkar Government College
Chennai - 600 039. INDIA

ABSTRACT

In bioinformatics, the discovery of transcription factor binding affinities is important. This is done by sequence analysis of micro array data. The determination of continuous and gapped motifs accurately from the given long sequence of data, say genetic data is challenging and requires a detailed study. In this paper, we propose an algorithm that can be used for finding short continuous, short gapped, long continuous, long gapped and negative existence of motifs. We propose a new DNA algorithmic approach which solves the accurate determination of motifs continuous and gapped, parallely with optimum time. Using the proposed algorithm, firstly a modified Position Weight Matrix is generated according to the searched motif pattern, which contains the position of its appearance in the given database, using DNA operations. Then, this Position Weight Matrix is used for searching of continuous and gapped subsequences. The proposed algorithm can be used to search genetic, scientific as well as commercial databases. Implementation results showed the correctness of the algorithm. Finally, the validity of the algorithm is checked and its complexity is analyzed.

General Terms

Sequence Mining, Pattern Recognition, Data Mining.

Keywords

DNA computation, DNA operations, Motifs, PWM.

1. INTRODUCTION

Extracting frequent subsequences from a database of sequences [23], is an important data mining task with plenty of different application domains, such as bioinformatics, web usage mining, mobility data analysis, motif discovery, commercial database analysis, program execution traces, sequences of words in a text, DNA and protein sequence extraction. Motif discovery in sequences, typically involves the discovery of binding sites, conserved domains or otherwise discrimatory subsequences. In bioinformatics, the two predominant applications of motif discovery are sequence analysis and micro array data analysis. Less common applications include discovering structural motifs in proteins and RNA [8].

The task of discovering frequent subsequences as patterns in a sequence data base has become an important topic in data mining [22, 1, 12, 16, 28, 20, 7]. The proposed new approach can be used for mining closed repetitive non overlapping gapped subsequences from a sequence database apart from continuous sequences using DNA operations. The majority of the tools can be found at the extreme ends of the spectrum with tools that exhaustively enumerate regular expressions at one end and probabilistic tools, based on Position Weight Matrices(PWMs), at the other [24, 5, 11, 26, 25]. This partitioning of tools is due to a computational trade-off: more descriptive motif representations such as PWMs frequently make exhaustive searches computationally infeasible[11]. The definition of the search problem, especially the formulation of objective functions, leaves space for substantial improvement in the performance of the motif discovery tool [27].

PWM are broadly used in computational biology to model conserved sequence patterns. The most common application of PWMs is about gene regulation: the transcription of a gene is controlled by regulatory proteins that bind to transcription factor binding sites (TFBSs) on DNA. A PWM specifies the frequency distribution of nucleotides at each position of the binding sites, and is considered to be related to the energy of binding of the transcription factor to the DNA. The motif finding problem is to find a PWM representing binding sites of an unknown transcription factor, an initio from sequence data [24].

2. LITERATURE REVIEW

Our work is a variant of sequential pattern mining, first introduced by Agarwal and Srikant [22] and further studied by many, with different methods proposed, such as PrefixSpan by Pei et al [12] and SPAM by Ayres et al[2]. There are studies on mining only representative patterns, such as closed sequential patterns by Yan et al [28] and Wang and Han [13]. However, different from ours, sequential pattern mining ignores the (possible frequent) repetitions of patterns within a sequence. The support of a pattern is the number of sequences containing the given pattern. Consider an example: in a database of 100 sequences, S1 ... S50 = CABABABABABD and S51... S100 = ABCD. In sequential pattern mining, both AB and CD have support 100. It is undesirable to consider AB and CD equally frequent because AB appears more frequently than CD does in the whole database, as it repeats more frequently in S51,. . . , S100, which is useful in the applications mentioned above. In our repetitive support definition, we differentiate AB from CD: sup(AB)=5*50+50=300 and sup(CD)=100[6].

DNA operations are simulated in [19], the proposed work uses the DNA operations CUT and PCR found in [19]. Mining Generalized Centre String using DNA operations, given a sequential database is done in [18]. In DNA sequence mining, Zhang et al [20] introduce gap requirement in mining periodic patterns from sequences. In particular, all the occurrences (both overlapping ones and non overlapping ones) of a pattern in a sequence satisfying the gap requirement are captured, and the support is the total number of such occurrences. El-Ramly et al [16] study mining user-usage scenarios of GUI programs composed of screen. Lo et al [7] proposed iterative pattern mining, which captures occurrences in the semantics of Message Sequence Chart, a standard in software modeling. PWMs are widely used to represent TFBS in promoter regions of genes [26, 9, 25]. Various ways of building a PWM have been carried out, some of them are found in [10, 24, 5, 11]. Similar works on PWM have been done by Segal et al 2003, Jensen and Liu 2004. Unfortunately many of these PWMs have low information content and match a huge number of sequences in the genome. However, most of these matches are likely to be false positives [17, 21]. There are also studies on mining repetition of patterns within sequences [1, 16, 20, 7] and mining closed repetitive gapped subsequences from a sequence database [6]. The time complexity of algorithms INSgrow and GSgrow [6] are

$O(|I|.\log L)= O(\sup(P).\log L)$, and $O(\sum_{P \square \text{Fre}}\sum_{e \square E} \sup(P).\log L)$

$= O(\sum_P \sup(P).E\log L)$ respectively, where P is the pattern, E is the additional factor and I is the left most support set.

3. PRELIMINARIES

In this paper, we study the continuous and gapped subsequences mining problem. Here we propose a new approach to paralley search for all models of any length, and also the positions of all their gapped instances in input sequence, using Position Weight Matrices (PWM). Our approach makes minimal (no) assumptions about the background sequences model and the mechanism by which elements affect gene expression. This provides a versatile motif discovery method, across all data types and genomes, with exceptional sensitivity and near-zero false-positive rates. In this work, we describe an approach for inferring motifs from gene expression data that aims at making as few a priori assumptions as possible. Our algorithm does not use any complex statistical models but rather involves directly quantifying the dependency between the presence or absence of a given motif in a regulatory region and the expression of the corresponding gene.

The exponential nature of some PWM problems is a limiting factor for using matrices of medium or large length. Score threshold computations for matrices whose length is greater than 15 usually require several seconds, and hours or days for matrices of length greater than 20 [10]. The JASPAR [4] and TRANSFAC [15] databases already contain almost 200 matrices of length 15 to 30 that is 20 of their total number of matrices. Moreover, the new techniques using data from next-generation sequencers should produce longer matrices [10]. Thus we need an efficient way to store the larger databases and access it parallel. Here we use DNA strands to store large data and DNA operations to access them parallely [18]. These experiments [4, 15, 3, 14] provide an opportunity to computationally improve the accuracy of existing position-specific matrices. It is this important problem that we address in the proposed study.

Let ϵ be a set of distinct events. A sequence S is an ordered list of events, denoted by $S = \langle e1,e2,...elength\rangle$, where $ei \in \epsilon$ is an event. For brevity, a sequence is also written as S = e1, e2, ...elength. We refer the ith event ei in the sequences, denoted by SeqDB = S1, S2, ..., Sn.

Definition 1 (Subsequence and Landmark): Sequence S = e1, e2, ...em is a subsequence of another sequence S′ = e’1, e’2, ...e’n $(m \leq n)$, denoted by $S \subseteq S'$ (or S′ is a super sequence of S) $1 \leq l1 \leq l2 \leq ... \leq lm \leq n$ such that S[i] = S′[li] (i.e., ei = e’li) for i = 1, 2, ..., m. Such a sequence of integers $\langle l1, l2, ...lm\rangle$ is called a landmark of S in S′.

Note that there may be more than one landmark of S in S′. By gapped subsequence we mean a subsequence, which appears in a sequence in the database, possibly with gaps between two successive events.

A pattern $P = e_1, e_2, ...e_m$ is also a sequence. For two patterns P and P', if P is a subsequence of P', then P is said to be a subpattern of P', and P' a super-pattern of P. Definition 2 (Instances of Pattern): For a pattern P in a sequence database $SeqDB = S_1, S_2, ..., S_n$, if $\langle l_1, e_2, ...l_m\rangle$ is a landmark of pattern $P = e_1, e_2, ...e_m$ in $S_i \in$ SeqDB, pair $(i, \langle l_1, e_2, ...l_m\rangle)$ is said to be an instance of P in SeqDB, and in particular, an instance of P in sequence S_i.

Definition 3 (Overlapping Instances): Two instances of pattern P $= e_1, e_2, ...e_m$ in $SeqDB = S_1, S_2, ..., S_n$,$(i, \langle l_1, e_2, ...l_m\rangle)$ and $(i', \langle l_1, e_2, ...l'_m\rangle)$, are overlapping if (i) $i = i'$, AND (ii) $\exists 1 \leq j \leq m; l_j = l'_j$. Equivalently, $(i, \langle l_1, e_2, ...l_m\rangle)$ and $(i', \langle l'_1, e'_2, ...l'_m\rangle)$ are non-overlapping if (i′) $i = i'$, OR (ii′) $\forall 1 \leq j \leq m : l_j = l'_j$.

Definition 4 (Repetitive Support and Support Set): The repetitive support of a pattern P in SeqDB is defined to be $\sup(P) = \max (I)$ where $I \,\varepsilon\, SeqDB(P)$ is non-redundant. The non-redundant instance set I with $I = \sup(P)$ is called a support set of P in SeqDB.

Definition 5 (Position Weight Matrix): Given a finite alphabet Σ and a positive integer m, a PWM M is a matrix with $\|\Sigma\|$ rows and m columns. The coefficient M, (p, x) gives the score at position p for the letter x in Σ. The PWM defines a function from σ_m to $\mathbb{R}$, that associates a score to each word $u = u_1u_2...u_p$ of σ_m:

$$Score_M (u)=\Sigma_{mp-1} M (p, u_p),$$

Let α be a score threshold. We say that M has an occurrence in a text T at position k if $Score_M (T_k ...T_{k+m-1}) \geq \alpha$.

The most recurrent task is to predict binding sites in a large DNA sequence, that is to look for occurrences of a PWM, given a text.

4. DNA-BASED-ALGORITHM

The algorithm is a new approach to find motifs of short continuous, short gapped, long continuous, long gapped and negative existence of motifs.

```
Algorithm 1: DNA-based-algorithm
Input: S1, S2

Output: PWM strands, INST strand
```

```
Begin
let L ← length(S2) ;
 foreach element of S2 do
     Create threads for each element of S2
     parallely ;
     foreach thread do
       let S3 ← pcr(S1) ;
         PWM1[]...PWML[]←
          cut(S3, S2[element1...L]) ;
       End
     End
     Check for instances of S2 such that S2[1] <
     S2[2] < ... S2[L] in PWM ;
 foreach PWM1[$i_1$] PWM2[$i_2$] PWM3[$i_3$]
 ... PWML[$i_L$] do
     for($i_1$, $i_2$, $i_3$, ..., $i_L$ ranges from 0 to L)
     and (no PWM strand is empty) ;
     if PWM1[$i_1$] < PWM2[$i_2$] ...  <
                    PWML[$i_L$] then
     INST[1..N]←PWM1[$i_1$],
     PWM2[$i_2$]...,PWML[$i_L$];
      $i_1$ + +, $i_2$ + +, ..., $i_L$ ++;
    end
 end
 end
```

Inputs for Algorithm 1 are S1 (the DNA strand which contains the encoded sequence, for example A can be encoded as AT and B can be encoded as CG, as shown in Figure 2) and S2 (the subsequence for which the instances are to be searched for in S1). The algorithm outputs the Position Weight Matrix (PWM) strands and the INST strand that is, the positions at which the S2 is found in S1.

Step 2 gets the length, L, of S2, that is the number of elements in the subsequence. Steps 3-8 performs the task of PCR and CUT operations [19], for each of the element in S2. Parallely for every element of S2 a thread is generated, S1 is multiplied and stored in S3, the CUT operation is performed on S3 for the given S2[element] and the positions are stored in the respective PWM strands, that is, PWM1, PWM2, PWM3, ..., PWML, one for each element of S2, thus generating L PWM strands. In Steps 10-18, the entries in the PWM strands are checked for the order of presence of S2[elements], with respect to the positions in which they appear in S1. The entries in PWM are checked for their instances, such that the position of S2[1] < position of S2[2] etc till S2[L], thus we obtain the instances of S2 in S1 as shown in Figure 1.

S 1   A B A B C A B C C A B A B C C C A A B B A
[Given Sequence]

S 2   A B C   [Sequence to be searched]

A   1 3 6 10 12 17 18 21

B   2 4 7 11 13 19 20

C   5 8 9 14 15 16

Positions of the gapped sequences are

(1,2,5) (3,4,8) (6,7,9) (10,11,14) (12,13,15)

Figure 1: Illustration of Algorithm 1

The PWM used in the Algorithm 1, stores the positions of presence of elements in the subsequence S2, such that positions of presence of each element is stored in a DNA strand, by checking for PWM1[i_1] < PWM1[i_2] < ... < PWM1[i_L] until any of the PWM strand is empty, then the positions,(instances) are stored in INST strand. Since DNA operations are used for generation of PWM strands and DNA strands are used for the storage and access of elements of PWM, the computational difficulty of storage and retrieval of large data from large databases are minimum, unlike discussed in [11, 4, 15, 3, 14]. Thus the Algorithm 1 discovers the gapped motifs of all patterns parallely with optimum time complexity as discussed below.

Lemma: 1 Let n = |S1|, where n > 0 and L = |S2| where $0 \leq L \leq n$. Then |INST| = $\leq$ n/L.

The Given Encoded Sequence

S1   A T A T A T A T A T A T A T A T C G C G C G C
G C G C G A T C G A T C G A T C G

S2   A T C G   [Sequence to be searched]

AT   1 3 5 7 9 11 13 15 29 33 37

CG   17 19 21 23 25 27 31 35 39

Positions of the gapped sequences are

(1,17) (3,19) (5,21) (7,23) (9,25) (11,27) (13,31) (15,35) (29,39)

Figure2: Illustration of Algorithm 1 using nucleotides

The Given Sequence   A B C A B C A B C A A B B C C

Sequence to be searched   A B C D

A   1 4 7 10 11

B   2 5 8 12 13

C   3 6 9 14 15

D

Since D is empty, we conclude that the subsequence ABCD does not exists or negative existence of a subsequence.

Figure 3: Negative Existence of Given Sequence

Proof: If S2 occurs in equal probability in S1, where n =|S1| and L = |S2| then $|PWM_1| == |PWM_2| == |PWM_L|$ therefore |INST | = n/L, otherwise |INST | < n/L .

From Figure 1, n = 21 and L = 3. If all elements of S2 occur with equal probability in S1 then |INST | is 7 that is n/L.

4.1 Special Case: Negative existence of subsequence

The Algorithm 1 can be used for finding the continuous sequences and gapped sequences of any type. For example the proposed work can be used for finding instances of the sort, ABC, A-BC, A-B-C, AB-C and even the non existence of the given sequence that is the negative existence of the subsequence as shown in the Figure 3.

Theorem1: Algorithm 1 can be used to identify the non existence of a subsequence.

Proof: From step 12 of Algorithm 1 for finding the instances, if any one of the PWM strand is empty, it concludes the non existence of the given subsequence. Figure 3, shows that strand D is empty, hence algorithm 1 concludes that the subsequence ABCD does not exists in the given sequence that is negative existence of ABCD.

4.2 Time complexity

Let n be |S1| and L be |S2|. Time complexity for pcr and cut operations, is O(1) at its best case, and O(n/M) at its average case [19].

TC(Algorithm1)=max(O(max(PCR,CUT)),O(INST))

If PWM □ Ø, then
TC(INST)=(n/L) iff |INST|=n/L
TC(INST) = $\max(|PWM_i|)$ where $1 \le i \le L$ iff |INST| < (n/L)

Therefore at best case
TC(Algorithm1)=O(n/L) to $O(\max|PWM_i|)$
At average case
TC(Algorithm1) = (O(n/M) + O(n/L)) to (O(n/L) + $O(\max|PWM_i|)$)

If PWM $\in$ Ø, that is , for the negative existence of the given subsequence, the
TC(Algorithm1) = O(PCR + CUT) that is O(n/M) at the average case.

5. PERFORMANCE

Algorithm 1 has been implemented and tested with both simulated and real database. The random DNA sequences of size varying from 100 to 25000 are generated from http://old.dnalc.org/bioinformatics/dnalcnulceotideanalyzer.htmrandomizer and http://old.dnalc.org/bioinformatics.org/sms/randdna.html. The real data is collected from EMBL database in FASTA format. The genome sequences of 3021 viruses are tested for the existence of all required gapped and continuous patterns. The database is got from http://www.ebi.ac.uk/genomes/virus.html.

Algorithm 1 proved to be efficient and accurate in finding all positions of existence, of the given subsequence. Tested with both randomly generated and real motifs our work could discover all motifs present, with its positions of existence. All implementations are performed on a dual core computer and 5 GB main memory using Java language. The operating system is Windows XP. The proposed algorithm is a new approach for solving the above problem. The resulted data of these experiments are consistent with the complexity analysis given in the previous section. The limitation of this algorithm is that the maximum number of threads generated is dependent on the system architecture.

6. APPLICATIONS

Since the proposed work uses DNA strands for its operations, the storage and retrieval processes can be implemented easily and parallely, whatever may be the size of the database. The Algorithm 1 can be used for finding short continuous, short gapped, long continuous, long gapped motifs for example ABC, A-BC, A-B-C, AB-C and even the non existence of the given subsequence i.e the negative existence of the subsequences. Since the applications for searching, for the existence of subsequences given a large database of commercial or genetic information are unlimited, the searching for negative existence has its importance in many industrial, research and scientific applications. Especially in medical and genetics field the finding of all patterns of motifs, can be used to predict, analyze and conclude the existence or future liability of any disease or abnormality present in the patient data. This work can also be applied to analysis of rule based systems, expert systems, rule mining, pattern mining, program execution traces, algorithm behavioral patterns, credit card data analysis or any other commercial data analysis, etc.

7. CONCLUSION

In this paper, we have designed a new approach and performed the implementation to solve it in a highly parallel way, for finding all subsequences, and can be extended to many other data mining applications also. In the future, it is possible to apply this approach for motifs search in multiple sequences parallelly and to solve more real time problems in molecular biology.

8. REFERENCES

[1] H.M. Annila, H.Toivonen, and A.I.Verkamo, 1997, Discovery of frequent episodes sequences. Data Mining and Knowledge Discovery, 1(3):259-289.

[2] J. Ayres, J.Flannick, J.Gehrke, and T.Yiu.,2002, Sequential Pattern mining using a bitmap representation. Int. Conf. on Knowledge Discovery and Data Mining, pages 429-435

[3] C. M. Bergman, J.W. Carlson, and Celniker, 2005, DNase I footprint database: A systematic genome Annotation of Transcription Factor Binding sites in the Fruitfly Bioinformatics, 21 : 1747 – 1749.

[4] J.C. Bryne, E. Valen, MH. Tang, T. Marstrand, O.Winther, da Piedade, A. Krogh, B. Lenhard, and A. Sandelin., 2008, JASPAR, the open access database of transcription factor-binding profiles: new content and tools in the 2008 update. Nucleic Acid Res, pages 102-6.

[5] Isabelle da Piedade, Man-Hung Eric Tang, and Olivier Elemento., 2009, DISPARE: discriminative pattern refinement for position weight matrices. BMC Bioinformatics, 10(388):1471-2105.

[6] Bolin Ding, David Lo, Jiawei Han, and Siau-Cheng Khoo., 2009, Efficient mining of closed repetitive gapped subsequences from a sequence database. Int. Conf. on Bioinformatics and Biomedical Engineering, pages 1024-1035, june.

[7] D.Lo, S.C.Khoo, and C.Liu., 2007, Efficient mining of iterative patterns for software specification discovery. Int. Conf. on Knowledge Discovery and Data Mining, pages 460-469.

[8] L. Holm et al., 1992, A database of protein structure families with common folding motifs. Protein Science, pages 1691-1698.

[9] G.Hertz and G.Stormo., 1999, Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. Bioinformatics, 15(7-8):563-577.

[10] Mathieu Giraud and Jean-Stephane Varre. , 2009, Parallel position weight matrices algorithms. International Symposium on Parallel and Distributed Computing, pages 65-69.

[11] L.Kyle Jensen, P. Mark Styczynski, Isidore Rigoutsos, and N. Gregory Stephanopoulos, 2006, A generic motif discovery algorithm for sequential data. Bioinformatics, 22(1):21-28.

[12] J.Pei, J.Han, B.Mortazavi-Asl, H.Pinto, Q.Chen, U.Dayal, and M.C.Hsu., 2001 Prefixspan: Mining sequential patterns efficiently by prefix projected pattern growth. Int. Conf. Data Engineering, (215-224).

[13] J.Wang and J.Han, 2004, BIDE: efficient mining of frequent closed sequences. Int. Conf. on Data Engineering, pages 79-90, Aug.

[14] M.H. Kuo and C.D Allis., 1999, In vivo cross-linking and immunoprecipitation for studying dynamic protein:DNA associations in a chromatin environment. Methods, 19:425-433.

[15] V. Matys, E. Fricke, R. Geffers, E. Gossling, M. Haubrock, R. Hehl, K. Hornischer, D. Karas, A. Kel, O. Kel-Margoulis, D. Kloos, B. Lewicki-Potapov, H. Michael, R.Munch, I. Reuter, S. Rotert, H. Saxel, M. Scheer, S. Thiele, and E. Wingender, 2003, TRANSFAC: transcriptional regulation from patterns to profiles. Nucleic Acid Res, 31:374-378.

[16] M.El-Ramly, E.stroulia, and P.Sorenson, 2002, From run-time behavior to usage scenarios: an interaction -pattern mining approach. Knowledge Discovery and Data Mining, pages 315-324.

[17] M.Tompa, N.Li, TL.Bailey, GM.Church, B.De Moor, E. Eskin, AV.Favorov, MC.Frith, Y.Fu, WJ.Kent, VJ.Makeev, AA.Mironov, WS.Noble, G.Pavesi, G.Pesole, M.Rgnier, N.Simonis, S.Sinha, G.Thijs, J.Van Helden, M.Vandenbogaert, Z.Weng, C.Workman, C.Ye, and Z.Zhu,2005, Assessing computational tools for the discovery of transcription factor binding sites. Nat Biotechnology, 23:137-144.

[18] A. Murugan and B.Lavanya.,2010, DNA algorithmic approach to solve GCS problem. Journal of Computational Intelligence in Bioinformatics, 3(2):239-247.

[19] A. Murugan, B.Lavanya, and K. Shyamala, 2011, A novel programming approach for DNA computing, International Journal of Computational Intelligence Research, 7(2):199-209.

[20] M.Zhang, B.Kao, D.Cheung, and K.Yip., 2005, Mining periodic patterns with gap requirement from sequences. SIGMOD Int. Conf. on Management of Data,. pages 623-633.

[21] Li N and M.Tompa., 2006, Analysis of computational approaches for motif discovery. Algorithms Mol Biol, pages 1-8.

[22] R.Agarwal and R.Srikant., 1995, Mining sequential patterns. Int.Conf. on Data Engineering.

[23] R.Agarwal and R.Srikant., 1996, Mining sequential patterns: Generalizations and performance improvements Extending Data Base Technology, pages 3-17.

[24] Saurabh Sinha, 2006,. On counting position weight matrix matches in a sequence, with application to discriminative motif finding. Bioinformatics, 22(14):454-463.

[25] R. Staden, 1984, Computer methods to locate signals in nucleic acid sequences. Nucleic Acids Res, 12:505-519.

[26] G. Stormo, 2000, DNA binding sites: representation and discovery. Bioinformatics, 16:16-23.

[27] M. Tompa., 1999, An exact method for finding short motifs in sequences with application to ribosome binding site problem. Proc. Seventh Int'l Conf Intelligent Systems for Molecular Biology, pages 262-271.

[28] X.Yan, J.Han, and R.Afhar., 2003, Colspan: Mining closed sequential patterns in large datasets. SIAM Int. Conf. Data Mining, pages 166-177.