Efficient Algorithms for Mining DNA Sequences

Guojun Mao
Information Scholl, Central University of Finance and Economics, Beijing, P R China

Abstract - *Most data mining algorithms have been designed for business data such as marketing baskets, but they are less efficient for mining DNA sequences. Unlike transaction sequences in business, DNA sequences typically have a small alphabet but a much long size, and so mining DNA sequences faces different challenges from other applications. This paper deals with the problem of mining key segments from DNA sequences, and by designing a compact data structure called Association Matrix, our algorithms maintains a less memory consumption as well as get more precise mining results. The Association Matrix is a novel in-memory data structure, which can be proved by experiments so compact that it can deal with super long DNA sequences in a limited memory usage. Using sliding window techniques, we can also transfer a super long DNA sequence into a series of formal short sequences. Thus, we not only may process a single long DNA sequence in a high efficiency, but also can mine valuable patterns from multiple length-varied DNA sequences. Based on these models, we design the algorithms for mining key segments from a single long DNA sequence and from multiple DNA sequences, and related experiments show these algorithms are scalable with changing of different minimum association degrees.*

Keywords: Data mining, mining DNA sequence, association matrix, key segment.

1 Introduction

The development of molecular biology in the last decades has made various biological data coming. Using data mining techniques to analyze biological data has been becoming an important research problem. However, it is fact that most classical data mining algorithms were designed for business transaction data as the first motivation, and it also has been proved that they are not more efficient for analyzing DNA sequences.

In general, a DNA sequence can be represented as an alphabetical string in the biological databases, but such a string often has different features from a business transaction sequence.

First, the DNA sequences always have a small alphabet. That is, there are four different nucleic acids: Adenine(A), Thymine(T), Guanine(G), and Cytosine(C). In contrast, in a marketing database, the transaction sequences are defined in a large alphabet that represents hundreds or thousands goods.

Second, a DNA sequence often is a very long sequence. For example, the human genome is made of roughly three billion of nucleic acids. In contrast, in a marketing database, a transaction sequence mostly is comprised of a lot of shorter sequences from 10 to 20 items.

Third, a super long DNA sequence with a small alphabet often contains a few of appearance-frequent short sequences (Segments), which always pay important structural or functional roles. Therefore, finding such segment patterns from the DNA sequences should be a different type of study project with other business data mining.

According to the special features of DNA sequences, this paper will design a called Association Matrix data structure, and making use of such a data structure, some novel algorithms to efficiently mining key segments from one or multiple DNA sequences will be constructed.

The contributions of this paper are as follows:

- It introduces a novel data structure called Association Matrix for mining key segments from DNA sequences. To the best of our knowledge, this is the first such structure to mining DNA sequences, and it also has many potential applications in analyzing other super long sequences.
- Using the sliding windows technique, multiple length-varied DNA sequences can be transferred into a series of formal short sequences, and so the efficient and effective algorithm for mining common patterns from multiple DNA sequences is constructed.

The rest of this paper is organized as follows. Section 2 gives the related work introduction. In section 3, we present basic theoretic methods for abstracting and expressing related problems. Section 4 gives the algorithms for mining key segments from a long DNA sequence or from multiple length-varied DNA sequences. In Section 5, we evaluate the performance of the proposed methods by experiments. Section 6 concludes this paper and future work.

2 Related Work

In bioinformatics, finding similarity of several sequences has been broadly studied, and a detailed survey on this technique was given by Hirosawa [1]. When an entire sequence is similar to another, their similarity is useful, but their local relations are difficult to be found by similarity computing. In fact, many biological problems such as poly-regions in DNA need search out important segments on a

DNA sequence. Papapetrou et al developed three efficient detection methods for it [2]: The first applies recursive segmentation that is entropy-based; the second uses a set of sliding windows that summarize each sequence segment using several statistics; the third employs a technique based on majority vote. These methods provide the basic ideas of mining key segments from DNA sequences.

Applying data mining techniques to DNA sequences has also become an important research focus. Specially, with appearance and development of DNA sequences or genomic databases, data mining can provide a basic technical support in view of computing bio-information. For example, Bell et al used frequency mining methods to discover common strings from DNA sequences [3].

In fact, mining frequent sub-sequences and association rules is a basic and important task in data mining, and it is a distillation of such techniques to detect strings that are very repetitive within a given sequence or across sequences. In addition, the problem of principal component analysis and discriminated analysis of DNA features were presented, some of which employed sequence classification techniques of data mining [4], [5]. Habib et al gave the methods of DNA motif comparison that was based on Bayesian algorithms of data mining [6].

From the view of data miming, one related problem to this paper is to mining frequent sequences from sequential databases. The concept of sequential patterns was first introduced and discussed as a data mining problem in 1995 [7]. Algorithm GSP was developed for mining sequential patterns that is a breadth-first search and button-up method [8]. Free-Span is another efficient algorithm for mining sequential patterns [9], which has less effort than GSP in candidate sequence generation, but still makes use of the spirit of GSP. Up to now, many effective algorithms for mining sequential data were presented [10]-[13]. The above research jobs were mainly based on business transaction databases, and focused on improving the performance of mining sequential databases that often include many shorter sequences.

Another relation-closed technique with this paper is mining frequent sub-sequences from long sequences. A series of research efforts on this field has been made by Mannila and his colleagues, including Bayesian analysis techniques on event sequences [14], frequent episodes in event sequences [15], and similarity evaluation between event sequences [16]. Other typical works include: segmenting long time series in an online way [17]; mining common rules from multiple sequences with the window size constraint 18]; finding frequent sequential patterns over a large scale of data by using Path-Tree [19]; discovering frequent patterns with periodic wildcard gaps [20].

3 Terminology and definitions

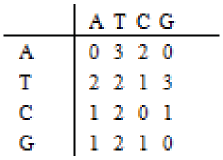
As is known to all that a cell uses DNA to store their genetic information, and a DNA molecule is composed of two linear strands coiled in a double helix. Each strand is made of a linear sequence with adenine(A), thymine(T), cytosine(C), or guanine(G), and two strands abide by base pairing rules (A with T and C with G). Therefore, modern bioinformatics has organized a DNA molecule into a character string and stored them in databases in order to be used in science research.

Definition 1 (DNA Sequence). Given alphabet set {A , G, C, T}, a DNA sequence is denoted by $s = <x_1, x_2, \dots, x_L>$, $x_i \in$ {A, G, C, T} for all i =1, 2, … , L. Also, for any a sequence $t= <y_1, \dots, y_{k-1}, y_k>$ in {A, G, C, T}, if exists i in s to have $x_{i+j-1} = y_j$ (j=1, 2, .., k), then t is called a sub-sequence of s, and $< y_1, \dots, y_{k-1}>$ is called the *Prefix* of t about s, represented as *Prefix*(t); y_k is called the *Postfix* of t, represented as *Postfix*(t); Thus, a sub-sequence t of a DNA sequence s can be through a postfix-connection operation ∞ to generate: t = *Prefix*(t) ∞ *Postfix*(t).

Definition 2 (Association Matrix). Given a DNA sequence $s = < x_1, x_2, \dots, x_L>$, an *Association Matrix* for it is defined as $(p_{i,j})$, where: each row element is related to the length-fixing strings of {A , G, C, T}, and if the fixing length is k, it is called a k-level association matrix. Also, it always has 4 column element, related to Letter A, G, C or T; each matrix element $p_{i,j}$ is an Integer, which represents the appearing number of the sub-sequence $i \infty j$ in the DNA sequence.

As a novel and important data structure, the Association Matrix can provide an efficient information abstract from scanning the original long DNA sequence, and will further support pattern mining from the DNA sequences.

Example 1. Considering the DNA sequence s = <ATGTCGTGATTGCATTACTACT>, its 1-level association matrix is shown in Fig. 1.



	A	T	C	G
A	0	3	2	0
T	2	2	1	3
C	1	2	0	1
G	1	2	1	0

Fig. 1. The 1-level Association Matrix for the DNA sequence in Example 1.

Definition 3 (Key Segment). Given an association matrix $(p_{i,j})$ on a DNA sequence. Set a minimum association threshold be *Min-Ass,* when $p_{i,j}$ >= *Min-Ass*, then $i \infty j$ is thought as a Key Segment of s.

Indeed, mining key segments from DNA sequences is our main target in this paper. By making use of the association matrix structure in Definition 2, it is easy to evaluate the occurring frequency of any sub-string in an original DNA sequence, and so key segments in a DNA sequence can be found out.

Example 2. For Fig. 1, if *Min-Ass* = 2, then its 2-length key segments can be obtained by scanning the 1-level association matrix: <AT>, <AC>, <TA>, <TT>, <TG>, <CT>, and <GT>.

Definition 4 (Maximum Key Segment). Given a DNA sequence *s*. A key segment is called a Maximum Key Segment only when it is not contained by any other key segments of *s*.

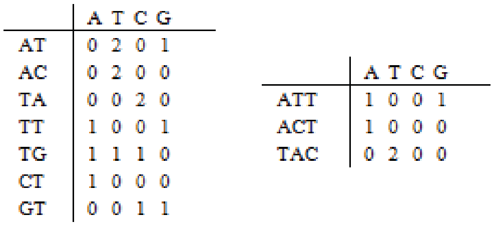
4 Algorithms

Large-size DNA samples can derive from a multitude of diverse organisms, and a key problem is to functionally search out key sub-sequences that is often much shorter but appearance-frequent. Such short sequences are called key segments in this paper. In this section, we first design the algorithm for discovering key segments from a single DNA sequence. Then, through analyzing the key problems of concurrently mining multiple DNA sequences, discuss the related mining methods.

4.1 Mining key segments from a DNA sequence

Naturally, an iteration procedure is necessary to find out key short sequences from a long sequence. As an instance, Example 2 has generated 2-length key segment set in the DNA sequence, so we can continue to do iterations to this dataset, in order to discover other key segments with longer sizes.

Example 3. For DNA sequence *s* in Example 1, we have gotten its 2-length key segment set: {<AT>, <AC>, <TA>, <TT>, <TG>, <CT>, <GT>}, so its 2-level association matrix can be further constructed shown as Fig. 2 (a). Going a step further, its 3-length key segment set is: {<ATT>, <ACT>, <TAC>}, and so its 3-level association matrix can be written as Fig. 2 (b). Scanning the 3-level association matrix, its one 4-length key segment <TACT> is found. Duo to its 4-level association matrix degenerates into a vector (1,0,0,0), so no 5-length key segment is found and the iteration is stopped.



	A	T	C	G
AT	0	2	0	1
AC	0	2	0	0
TA	0	0	2	0
TT	1	0	0	1
TG	1	1	1	0
CT	1	0	0	0
GT	0	0	1	1

(a)

	A	T	C	G
ATT	1	0	0	1
ACT	1	0	0	0
TAC	0	2	0	0

(b)

Fig. 2. The Association Matrixes of 2-level and 3-level for the DNA sequence in Example 1

Obviously, the Association Matrix structure is simple, but it can be efficient for finding key segments from a DNA sequence. Therefore, based on association matrix as an in-memory structure, we can organize related association information scanning from a long DNA sequence to the main memory, and so important segments can be efficiently searched out. Based on the step-by-step iteration idea, we can design the effective algorithm to mine a DNA sequence. Algorithm 1 provides the pseudo code for mining key segments from a single long DNA sequence.

Algorithm 1. Mining key segments from a DNA sequence.

```
INPUT: DNA sequence s; minimum association Min-Ass.
OUTPUT: s' key segments KS.
begin
k←1; m←4; row-set←{A,T,G,C};
WHEN row-set is not null DO
    generate the s' k-level Association Matrix: $(p_{i,j})_{m*4}$;
    row-set←{};
    FOR i=1 TO m
        FOR j=1 TO 4
            IF pi,j >= Min-Ass THEN insert i∞j into row-set;
    add all elements of row-set into KS;
    updating m with the size of row-set;   k++;
ENDDO
Return KS.
end.
```

Example 4. For DNA sequence *s* in Example 1, applying Algorithm 1, all key segments can be obtained: {<AT>, <AC>, <TA>, <TT>, <TG>, <CT>, <GT>, <ATT>, <ACT>, <TAC>, <TACT>}; and its maximum key segment set is: {<TG>, <GT>, <ATT>, <TACT>}.

4.2 Mining key segments from multiple DNA sequences

To understand the common gene characters in multiple DNA sequences, it is necessary to together mining them. However, the job can be more difficult. This is because they can have different lengths as well as they can be very long. For solving this problem, we use sliding window technique, which can make multiple length-varied DNA sequences becoming more formal mining objects.

Based on sliding window, we can transfer a super long DNA sequence into a series of shorter sequences. Such short sequences are more formal and length-fixing, and so it is possible to process them in limited main memory more efficiently than to directly do these super long sequences.

Definition 5 (DNA Short Sequence). Given an original DNA long sequence s = $<x_1, x_2, \dots, x_L>$ and the size of the sliding window K, if L is much larger than K, the set of short sequences for s can be built up by using sliding window technique. That is, s is transformed into the set of short sequences with the fixing length K: $\{s_i\}_{L-K+1}$, where each s_i =< x_i, x_{i+1}, ... , x_{i+K-1}> is called the ith window sequence.

Definition 6 (Association Matrix of Short Sequence Set). Given the short sequence set $sSet$={s^1, s^2, … , s^n}. For each short sequence s^k (k=1, 2, ..., n), its association matrix ($p_{i,j}$) can be obtained by :

$$(\sum_{k=1}^{n} p_{i,j}^{k}) \quad (1)$$

Where $p_{i,j}^{\ k}$ is the appearing number of the string $i\infty j$ in s^k (described as the above Definition 2).

Example 5. Set the size of the sliding window be 10. Supposed the following three DNA sequences:

(1) <ATGTCGATTGCAAGCTGCG>;
(2) <AGTCGATGCATGATCG> ;
(3) <CGTCACTGATATG>.

Then, using sliding window technique, we can get the set of short sequences in every window from these three long sequences:

(1) The 1st window: {<ATGTCGATTG>, <AGTCGATGCA>, <CGTCACTGAT>};

(2) The 2nd window: <TGTCGATTGC>, <GTCGATGCAT>, <GTCACTGATA>};

(3) The 3rd window: {<GTCGATTGCA>, <TCGATGCATG>, <TCACTGATAT>};

(4) The 4th window: {<TCGATTGCAA>, <CGATGCATGA>, <CACTGATATG>};

(5) The 5th window: {<CGATTGCAAG>, <GATGCATGAT>, <ACTGATATGX>};

(6) The 6th window: {<GATTGCAAGC>, <ATGCATGATC>, <CTGATATGXX>};

(7) The 7th window: {<ATTGCAAGCT>, <TGCATGATCG>, <TGATATGXXX>};

(8) The 8th window: {<TTGCAAGCTG>, <GCATGATCGX>, <GATATGXXXX>};

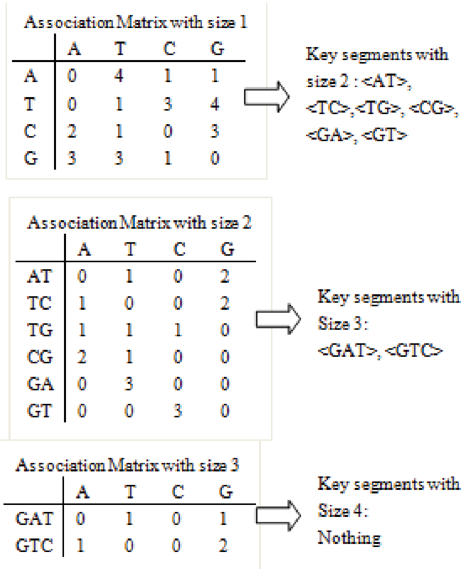
(9) The 9th window: {<TGCAAGCTGC>, <CATGATCGXX>, <ATATGXXXXX>};

(10) The 10th window: {<GCAAGCTGCG>, <ATGATCGXXX>, <TGXXXXXX>}.

Note that: there is Letter X excepting {A, T, C, G} in the above window data, which is not any meaning but just fills the vacancy positions.

For a fixed window, the investigated data can organized into a set of short sequences from multiple DNA sequences by Definition 5. Also, when scanning the short sequence set in a window, its association matrix can be built according to Definition 6. Thus, its key segments in this window can be found out according to the idea of Algorithm 1.

For example, suppose *Min-Ass* =3, as far as the first window data in Example 5 is concerned, we can deal with it in a loop way with increasing the sizes of strings in the window. Fig. 3 shows related processing detail. Through iteratively computing in the first window, the key segment set in this window is found: {<AT>,<TC>,<TG>, <CG>, <GA>, <GT>, <GAT>,<GTC>}; and the maximum key segment set is {<TG>, <CG>, <GAT>,<GTC>}.



Association Matrix with size 1

	A	T	C	G
A	0	4	1	1
T	0	1	3	4
C	2	1	0	3
G	3	3	1	0

Key segments with size 2 : <AT>, <TC>,<TG>, <CG>, <GA>, <GT>

Association Matrix with size 2

	A	T	C	G
AT	0	1	0	2
TC	1	0	0	2
TG	1	1	1	0
CG	2	1	0	0
GA	0	3	0	0
GT	0	0	3	0

Key segments with Size 3: <GAT>, <GTC>

Association Matrix with size 3

	A	T	C	G
GAT	0	1	0	1
GTC	1	0	0	2

Key segments with Size 4: Nothing

Fig. 3. The process for getting key segments from a window of multiple sequences (for Example 5, the first window, *Min-Ass* =3).

Algorithm 2 gives the pseudo code for mining key segments from multiple long DNA sequences. Algorithm 2 will deal with each window data in a loop way. And for each window, it need do three main things: (1) generating window data of short sequences; (2) finding key segments from the window data, whose processing phases include making and using the association matrix to store and search, so its basic ideal has been introduced in the above Algorithm 1; (3) Uniting the found key segments in all windows to get global key segments.

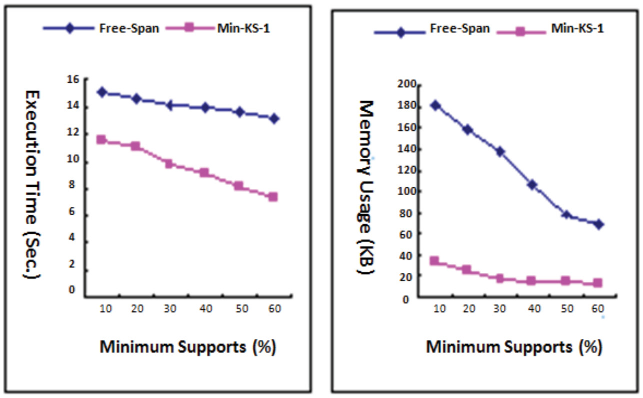
Algorithm 2. Mining Key Segments from Multiple DNA Sequences.

```
INPUT: DNA sequences $s^1$, $s^2$, $s^3$, …, $s^n$;  Sliding window size $K$;
       Minimum association Min-Ass.
OUTPUT: Key segments KS.
begin
k=1;
REPEAT
    clear sSet;
    generate the short sequence set of the kth window sSet;
    generate the key segment set $KS_k$ from sSet by Definition 6 and Min-Ass;
    insert $KS_k$ into KS;
    k++; move to the next window of the DNA sequences
UNTIL all data is processed
Return KS;
end.
```

5 Evaluations of experiments

To evaluate the proposed algorithms, the above Algorithm 1 and 2, denoted as Min-KS-1 and Min-KS-*n*. We compared these algorithms with the popular sequential pattern mining algorithm Free-Span [9], and conducted several experiments on an 800MHz CPU with 2GB main memory.

Experiment 1. The first set of experiments was conducted on the synthetic data set: C256S64N4D100K, which have average length 256, whole volume 100K, and there are 4 sequences to simulate biology objects [21]. We conduct Min-KS-1 and Free-Span on the 4 sequences with different minimum Association-degrees (or Support-degree stated as Free-Span). Fig. 4 (a) and (b) show the results of execution time and memory space consumptions. Note that when setting a minimum association or support degree, Min-KS-1 and Free-Span are all executed on the 4 sequences, and the time consumption in Fig. 4 (a) is the average of time-spends on the 4 sequences, but the memory consumption is the maximum spending on the 4 sequences.

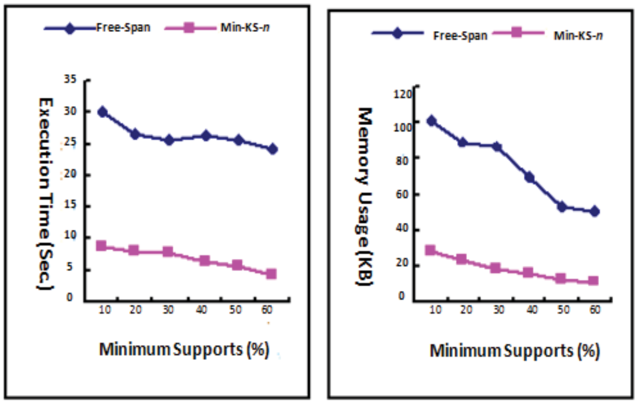


(a) (b)

Fig. 4. Time and space consumptions of Min-KS-1, and Free-Span on dataset C256S64N4D100K.

Result analysis of Experiment 1: From Fig. 4 (a), with increasing minimum Association degrees, both Min-KS-1 and Free-Span can make time down, but Min-KS-1 has much less time consumption than Free-Span on every minimum support degrees. Fig. 4 (b) shows that in comparison with Free-Span, main memory consumptions of Min-KS-1 are much less on every minimum degrees and more stable with varying minimum support degrees. A main factor contributed to these results is introducing Association matrix structure, which can be more compact than other data structure such that used in Free-Span.

Experiment 2. The second set of experiments was conducted on real life DNA sequences [21], which was organized into 125 DNA sequences (from size 2000 to 3000). We choose 5 sequences and use sliding window technique (Size 50) to evaluate the performance of Min-KS-*n*. Comparing algorithm is still Free-Span. Of course, Free-Span also has to be executed once for every window dada as well as Min-KS-*n* do. Fig. 5 (a) and (b) show the results of execution time and memory space consumptions of Min-KS-*n* and Free-Span, aiming to evaluate their scalability for mining key segments from multiple DNA sequences, with different minimum support degrees.



(a) (b)

Fig. 5. Execution time and memory space for Min-KS-*n* and Free-Span on real life DNA sequences.

Result analysis of Experiment 2: Fig. 5 shows, from left to right, with increasing minimum support degrees, executing time and using space consumptions of the two algorithms are declining, but our method are much better than Free-span.

6 Conclusions

DNA sequences are a different type of explosion of search space from classic transaction sequences, and so traditional data mining techniques for business transactional data are not more efficient for them. This paper has studied the challenges and methods for mining key segments from long DNA sequences.

We have presented a methodology to systematically mine DNA sequences. First, the structure Association Matrix aims at processing long sequences that have small letter set. Then, two main algorithms are presented to make this preliminary idea programmable. Algorithm 1 is to mine a single long DNA sequence, which introduces the basic procedure to discover key segments in a DNA sequence. By using the sliding window technique, Algorithm 2 implements finding key segments from multiple DNA sequences. In addition, experiments on both synthetic and real life data sets also demonstrated the proposed methods have less time costs and smaller space usages than some existing algorithms.

We are working on more experiments and graph layout algorithms in this research field. We will also investigate more biological sequences and do research to them.

7 Acknowledgment

First of all, we would like to extend my sincere gratitude to my colleagues including Dr. Wang and Dr. Liu, for her useful suggestions to this paper.

Second, high tribute shall be paid to two postgraduate students, Ms. J.Wang and Ms. S. Xie, for they help the author to implement programming of the algorithms and do related experiments of this paper.

Finally, I am also deeply indebted to the NSFC (National Science Foundation of China), for this work was supported in part by the NSFC 61273293. Also, this work is partly supported by the CUFE discipline construction.

8 References

[1] M. Hirosawa, Y. Totoki, M. Hoshida and M. Ishikawa, "Comprehensive study on iterative algorithms of multiple sequence alignment)," J. Comput. Applic. Biosci., Vol.11, pp. 13-18, 1995.

[2] P. Papapetrou, G. Benson and G. Kollios, "Mining poly-regions in DNA," Intl J. Data Min Bioinform. Vol. 6, Issue 4, pp. 406-418, 2012.

[3] D. Bell and J. Guan, "Data mining for Motifs in DNA sequences," Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing: Lecture Notes in Computer Science, Vol. 2639, pp. 507-514, 2003.

[4] Z. Liu, D. Jiao and X. Sun, "Classifying genomic sequences by sequence feature analysis," Genomics Proteomics Bioinformatics, Vol. 3, Issue 4, pp. 201-205, 2005.

[5] P. Stegmaier, A. Kel, E. Wingenderand J. Borlak, "A discriminative approach for unsupervised clustering of DNA sequence motifs," PLoS Comput Bio, Vol. 9, Issue 3, e1002958-e1002958 , 2013.

[6] N. Habib, T. Kaplan, H, Margalit and N. Friedman, "A novel Bayesian DNA motif comparison method for clustering and retrieval," PLoS Comput Biol, Vol. 4, Issue 2, e1000010-e1000010, 2008.

[7] R. Agrawal and R. Srikant, "Mining sequential patterns," in 1995 Proc. Intl. Conf. on Data Engineering, IEEE Computer Society Press (Taipei, Taiwan), pp.3~14.

[8] R. Srikant and R. Agrawal, "Mining sequential patterns: generalizations and performance improvements," in 1996 Proc. Intl Conf. on Extending Database Technology (EDBT), Springer-Verlag Press, pp.3–17.

[9] J. Han and J. Pei, "Free-span: Frequent pattern-projected sequential pattern mining," in 2000 Proc. Intl. Conf. Knowledge Discovery and Data Mining, ACM Press (New York) , pp.355-359.

[10] J. Mohammed, "SPADE: an efficient algorithm for mining frequent sequences," J. Machine Learning, Vol. 42, Issue 1, pp. 31-60, 2001.

[11] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal and M. Hsu, "Mining sequential patterns by pattern-growth: the PrefixSpan approach," IEEE Transactions on Knowledge and Data Engineering, Vol. 16, Issue 11, pp. 1241-1440, 2004.

[12] W. Peng and Z. Liao, "Mining sequential patterns across multiple sequence databases," Data & Knowledge Engineering, Vol. 68, Issue 10, pp. 1014–1033 , 2009.

[13] C. Liu, L. Chen, Z. Liu and V. Tseng, "Effective peak alignment for mass spectrometry data analysis using two-phase clustering approach," Intl J. Data Mining and Bioinformatics, Vol. 9, Issue 1, pp. 52-66, 2014.

[14] E.Arjas, H.Mannila, M.Salmenkivi, R. Suramo and H. Toivonen. BASS: Bayesian analyzer of event sequences, in Proc. of COMPSTAT'96, A. Prat (ed.) (Barcelona, Spain, 1996), pp.199-204.

[15] H. Mannila, H. Toivonen and I. Verkamo, "Discovery of frequent episodes in event sequences," J. Data Mining and Knowledge Discovery, Vol. 1, Issue 3, pp. 259-289, 1997.

[16] H. Mannila. and M. Salmenkivi, "Finding simple intensity descriptions from event sequence data," in 2000 Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, ACM Press, 2000 (New York), pp.341-346.

[17] E. Keogh, S. Chu, D. Hart and M. Pazzani, "An online algorithm for segmenting time series," in 2001 Proc. IEEE Intl Conf. on Data Mining, IEEE Computer Society Press (San Jose, California, USA,), pp.289-296.

[18] P. Fournier-Viger, X. Wu, V. Tsen and R. Nkambou, "Mining sequential rules common to several sequences with the window size constraint," Lecture Notes in Computer Science, vol. 7310, pp. 299-304, 2012.

[19] G. Lee, Y. Chen and K. Hung, "FPTree: Mining sequential patterns efficiently in multiple data streams environment," J. Information Science and Engineering, Vol. 29, Issue 6, pp. 1151-1169, 2013.

[20] Y. Wu, L. Wang, J. Ren, W. Ding and X. Wu, "Mining sequential patterns with periodic wildcard gaps," Appl. Intell. , Vol. 41, Issue 1, pp. 99-116, 2014.

[21] K. Wang, Y. Xu and J. Yu, "Scalable sequential pattern mining for biological sequences," in 2004 Conf. t Intl Conf. on Information and Knowledge Management, ACM Press (Washington, DC, USA), pp.178-187.