INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, e-ISSN 1841-9844, 14(4), 574-589, August 2019.

# Frequent Patterns Algorithm of Biological Sequences based on Pattern Prefix-tree

L.Y. Xue, X.K. Zhang, F. Xie, S. Liu, P. Lin

**Linyan Xue#, Xiaoke Zhang#, Fei Xie, Shuang Liu**
College of Quality and Technical Supervision,
Hebei University,
Baoding 071002, China
lineysnow@163.com; lianlianfushi@126.com; xiefei-1214@163.com; liushuang99@hotmail.com
#Joint first authors. These authors contributed equally to this work.

**Peng Lin***
College of Management,
Hebei University,
Baoding 071002, China
*Corresponding author: linpeng1982@139.com

**Abstract:** In the application of bioinformatics, the existing algorithms cannot be directly and efficiently implement sequence pattern mining. Two fast and efficient biological sequence pattern mining algorithms for biological single sequence and multiple sequences are proposed in this paper. The concept of the basic pattern is proposed, and on the basis of mining frequent basic patterns, the frequent pattern is excavated by constructing prefix trees for frequent basic patterns. The proposed algorithms implement rapid mining of frequent patterns of biological sequences based on pattern prefix trees. In experiment the family sequence data in the pfam protein database is used to verify the performance of the proposed algorithm. The prediction results confirm that the proposed algorithms can't only obtain the mining results with effective biological significance, but also improve the running time efficiency of the biological sequence pattern mining.
**Keywords:** Bioinformatics, frequent patterns, biological sequence, pattern prefix-tree, data mining.

## 1 Introduction

Bioinformatics is a new comprehensive cross discipline involving biology, mathematics, physics, informatics and computer science. It plays a vital role in the development of life science, and becomes the frontier of life science research. The core issue in bioinformatics is genome informatics which includes the obtaining, processing, storing, assigning and explaining of the genome information. By using computers and network as tools, based on the mathematical theory, methods and technology, genome informatics studies the biopolymers include the sequences, structures and functions of DNA and protein. The key issue in genome informatics is to understand the meaning of the order in nucleotide sequences, namely, to understand the exact locations of the genes in the chromosome and the functions of the DNA segments. These are very vital in the research of disease gene of people, the function of gene and the designing of pharmacy. To achieve those goals, pattern mining in biological data is the critical techniques [3, 7, 9].

Biological individuals have their particularity in the process of evolution, because a part of the sequence or region in the organism will affect the survival of the entire organism. Therefore, these sequence patterns will remain well-conserved in the evolutionary process. One of the vital contents of current biological sequence data analyzing is to mine proper biological sequence patterns. So far, researchers have proposed a large number of biological sequence pattern mining algorithms. These algorithms mainly find the following patterns in biological sequences:

(1) Repeat patterns in a biological sequence. For example, some continuous repeating patterns appeared in the DNA sequence called Tandem Repeats (TRS). It has been proved that these TRS play a key role in the evolution of genes. In the evolution of living individuals, using repeated sequences can help the formation of new genes [13]. Conversely, the generation of some human diseases may be caused by mutations in repeated sequences, such as DiGeorge syndrome, Williams syndrome, musculoskeletal muscular dystrophy [15], *etc.* However, we still do not fully understand these TRS and their biological functions. Therefore, finding and studying the repeat sequences in DNA sequences are very important for the establishment of repeat sequence databases and the unknown functional identification of biological sequences. In general, the above problems are included in the frequent pattern mining of biological single sequence.

(2) Conservative patterns in multiple biological sequence sets. Some sequence regions in the process of genetic variation in organisms will affect the survival of organisms, so these sequences will be highly conserved during the entire evolution of organisms. For example, most or even all of the sequences from the same family sequence set (such as the protein family) often contain conserved sequence pattern regions that play a vital role in the structure and function of protein [2]. Important functional sequences (transcription factor binding sites), which would be generally located in upstream region of co-expressed gene sequence, tend to be more conserved and can regulate the expression of genes [17]. These examples can all be seen as mining their conservative patterns in multiple biological sequences.

(3) Sequence patterns with different frequencies appear in sequences in multiple biological sequence sets. A repeating sequence is called a copy. In the human population, there are often differences in the number of repeats (the number of copies) of the repeat sequence, known as repeat copy number polymorphisms. This feature is widely used in genetic diversity analysis, individual identification, genetic diagnosis, genetic mapping and other applied research. For example, Saghai et al. conducted an experiment to verify the polymorphism through the RFLP linkage map [16]. In the experiment, they compared and analyzed the conserved regions covering 24% of rice gene sequences and 17 conserved regions covering 31% of barley gene sequences. They found that 72% of single copy genes in barley were similarly expressed in rice in a single copy. Similarly, about 60% of rice single-copy sequences are found in barley. This result shows that the difference between the grass crops is caused by the difference in repeat sequences, not due to structural differences in the genes. Excavating the repetitive patterns in the multi-sequences of organisms and their frequency of occurrence in each sequence has guiding significance in genetic recognition and so on [18]. This type of problem can be incorporated into the study of frequent patterns mining of biological multi-sequences.

## 2 Related works

At present, there are two main kinds of computational methods for studying biological sequence pattern discovery. Each kind of algorithm has a different search strategy. One kind of algorithm uses a heuristic search strategy and it is an approximate algorithm. In the artificial intelligence field, most machine learning methods use heuristic algorithms. In sequence pattern mining, such methods mainly include Gibbs sampling algorithm, EM method [11], MEME algorithm [4] and so on. This kind of algorithm is usually an iterative process, and it gets better solutions through iterations. In the algorithm, some approximate description of the sequence pattern information is needed to determine the quality of a certain measurement standard. In the iteration, the solution is optimized according to the criterion, and it is judged whether the iterative termination condition is satisfied. The advantage of this kind of algorithm is that the computational complexity is low and it is suitable for searching for a longer sequence pattern. However, the disadvantage is that the solution it obtains may be a local optimal solution, and it may not be able to obtain a global optimal solution. However, a large number of application practices have proved that such approximate solutions obtained by machine learning algorithms can be used to solve practical application problems.

Since the frequent pattern mining was defined by Agrawal and Srikant in 1995 [5], related research has become an important field of data mining and has received extensive attention from researchers. They proposed many algorithms, some of which are suitable for efficient mining of large-scale sequential patterns, such as SPADE, FreeSpan, Prefixspan, GSP, Apriori and so on. However, biological sequences are different from sequence data such as transaction sequences. When applying the above algorithm directly to biological sequence data, there are many difficulties and problems to be solved. For example, the meaning of fuzzy matching between patterns is difficult to understand, the mining results cannot meet the needs of biological research, some pruning strategies or data structures cannot be effectively applied to biological sequence data, and algorithm for data volume scalability. It is necessary for biological sequences to design frequent pattern detection algorithms. The existing algorithms mainly have the following categories:

(1) Tandem Repeats mining algorithm. Tandem repeats are a special type of sequence pattern, which is a subsequence that is arranged end-to-end in a DNA sequence, has repeating units in a string, and has a frequency exceeding a certain threshold. Tandem Repeats include Perfect Tandem Repeats (PTR), Longest Pattern Repeats (LPR), Approximate Tandem Repeats (ATR) and so on. Jiang et al. in [8] proposed the definition of the LPR for the exact search for new repeats, and designed an LPR search algorithm based on the subsequent array structure. KurtZ et al. [10] gave REPuter algorithm that depended on the data structure of the suffix tree. The repeated sequences are mined by pairwise alignment of subsequences. However, these algorithms are still difficult to find for frequently occurring repeats in DNA sequences [1, 6]. In addition, Won et al. proposed a mosaic silhouette algorithm to identify repeated sequences [19].

(2) Mining sequence models with conditional restrictions. In practical applications, we often want to mine sequence models with conditional constraints based on the characteristics of biological sequences and the needs of application problems. Such constraints usually include the need to mine patterns of biological sequences that may contain intervals of any length, to allow fuzzy matches between sequence patterns, and so on. The existing sequential pattern mining algorithm with some constraints does not consider the biological sequence's various constraint features. Liao et al. proposed a two-stage algorithm that can mine sequence patterns containing arbitrary length intervals [12]. The algorithm is divided into two phases: the first phase searches for all short frequent patterns, and these is no gap in these patterns; the second phase generates long patterns containing these short frequent patterns. The algorithm can use the short frequent mode information to reduce the search time of the spaced global mode. Although the algorithm can mine more sequence patterns that are more biologically meaningful, it takes more time to generate long patterns.

(3) Frequent subtree mining. With the continuous expansion of data mining applications, frequent patterns of mining objects are constantly diversified. There are sequences, transaction itemsets, and complex data structures such as graphs and trees to adapt to web mining, biological structure data mining semi-structured document mining. Therefore frequent subgraph mining and frequent subtree mining have become important research fields in frequent pattern mining. After several years of research, there have been some methods for mining frequent patterns such as trees and graphs. In the research of frequent subtree mining algorithms, the identification of tree isomorphism and the matching of tree patterns are the two key issues to be solved. In the frequent subtree mining algorithm, it is determined whether the tree in the database has a subtree that is isomorphic to a certain known tree. Let the tree in the database be $T$ and the known tree be $P$, $/T/\geq/P/$. If there is a one-to-one correspondence between the vertices of $P$ to $T$, and the edges of the corresponding vertices have the same relationship, $T$ is a subtree of $P$.

In the tree pattern matching problem of biological data, the trees $T$ and $P$ are ordered trees, and each vertex and edge have markers. At this point, the isomorphism of the subtree requires that the corresponding vertices have the same mark [14].

## 3 Single sequence frequent pattern mining algorithm

### 3.1 Basic mode

First of all, it is stipulated that DNA sequences always consist of four characters *A, C, G, T* and *\$* as the terminator. On this basis, the relevant definitions of patterns are given.

**Definition 1.** For alphabet $\Sigma = \{A, C, G, T\}$, pattern $P =< p_1, p_2, \ldots, p_n >$ and pattern $P' =< p'_1, p'_2, \ldots, p'_n >$ are given, approximation degree is defined as follows:

$$\text{Approximation\_degree}\left(P, P'\right) = \frac{|E|}{\text{length}(P)}$$

where $E = \{j | p'_j = p_j, j = 1, 2, \ldots, n\}$, it represents the set of same characters between two patterns. The approximation degree takes into account not only the Hamming distance between patterns, but also the influence of pattern length. Assuming that the Hamming distances between patterns are the same, the pattern lengths are 2 and 10, respectively, it is obvious that if the pattern length is 10, it means that the two patterns are more similar. When the degree approximation is 1, it means that the two patterns match perfectly.

**Definition 2.** For alphabet $\Sigma = \{A, C, G, T\}$, pattern $P =< p_1, p_2, \ldots, p_n >$ and pattern $P' =< p'_1, p'_2, \ldots, p'_n >$ are given, Approximation_match$(P, P')$ represents whether pattern $P$ and pattern $P'$ satisfy the minimum approximation specified by the user.

**Definition 3.** (Frequent approximation patterns). Given sequence $S$, pattern $P =< p_1, p_2, \ldots, p_n >$ is a frequent approximation pattern in sequence $S$ if and only if $\sum_{E=1}$ Approximation_match$(P, P') \geq m$, where

(1) $P_i$ is a substring of $S$,

(2) $|P_i| = |P|$,

(3) $m$ represents the minimum frequent threshold specified by the user,

(4) For $P_i = S[a \ldots b]$, $P_j = S[c \ldots d]$, where $i \neq j$, if Approximation_match$(P, P_i)$=1 and Approximation_match$(P, P_i) = 1$, then it must be $b < c$ or $d < a$.

**Definition 4.** (Support of frequent approximation patterns). Given sequence $S$, pattern $P =< p_1, p_2, \ldots, p_n >$ is a frequent approximation pattern in sequence $S$, then we call

$$\text{support}(P) = \frac{\sum_{i=1}^{n} \text{Approximation\_match}\left(P, P_i\right)}{[\text{length}(S)/\,\text{length}(P)]}$$

as support of frequent approximation patterns $P$ on sequence $S$. The higher the support degree, the higher the frequency of pattern occurrence in the sequence. When the support degree is 1, the pattern is a frequent and accurate pattern in the sequence.

**Lemma 5.** *0≤Approximation_degree*$(P, P')$*≤1*

*It is obvious that Approximation_degree*$(P, P')$*≥0, according to Definition 1, for pattern* $P =< p_1, p_2, \ldots, p_n >$ *and pattern* $P' =< p'_1, p'_2, \ldots, p'_n >$, $I = \{i | p'_i = p_i, i = 1, 2, \ldots, n\}$, *it is obvious that* $|I| \leq \text{length}(P)$, *so Approximation_degree*$(P, P')$*≤1.*

**Lemma 6.** *0≤sup(P)≤1*

*It is obvious that sup(P)≥0, according to definition of frequent approximation patterns, for $P_i = S[a \dots b]$, $P_j = S[c \dots d]$, if Approximation_match($P, P_i$)=1, and Approximation_match ($P, P_j$)=1, then it must be $b < c$ or $d < a$, so sup(P)≤1.*

$P$ is a frequent approximation pattern in sequence $S$, if there are m patterns $P_1$, $P_2$, ..., $P_m$ in sequence $S$, where $P_1 = S[a_1, a_1']$, $P_2 = S[a_2, a_2']$, $\cdots$, $P_m = S[a_m, a_m']$ $(a_1 < a_2 < \cdots < a_m)$, they are all satisfied Approximation_match($P, P_i$)=1, in order to maximize the number of non-overlapping patterns in $S$ sequence, the first non-overlapping pattern must be $P_1$.

According to definition of frequent approximation patterns, $|P_i| = |P|$. let $|P| = l$, then $P_1 = S[a_1, a_1+l-1]$, $P_2 = S[a_2, a_2+l-1]$, $\cdots$, $P_m = S[a_m, a_m+l-1]$. If the first non-overlapping pattern is not $S[a_1, a_1 + l - 1]$, it is the $i$-th $S[a_i, a_i + l - 1]$, where $a_i > a_1$. If $a_{i+1} - a_i \geq l$, then the second non-overlapping pattern is $S[a_i + 1, a_{i+1} + l - 1]$. If $a_{i+1} - a_i < l$, then $a_{i+2}$ and $a_i$ are compared, if $a_{i+1} - a_i < l$, then $a_{i+3}$ and $a_i$ are compared, until $a_{k+1} - a_i \geq l$. So the second non-overlapping pattern is $S[a_k, a_k + l - 1]$. And so on, the $n$-th non-overlapping pattern is obtained.

Obviously, the second non-overlapping pattern will not overlap with the first non-overlapping pattern. Because $a_i > a_1$, it doesn't overlap with pattern $S[a_1, a_1 + l - 1]$. The method can be used to clip candidate patterns and quickly find frequent approximate patterns satisfying conditions.

## 3.2 Basic mode table

After getting all the basic patterns of a sequence $S$, a basic pattern table of $S$ can be constructed. For ease of searching, it needs to sort all the basic patterns of $S$ by the lexicographic order of the characters in $|\Sigma|$.

For example, for basic pattern of $S$="yxzxxyzyxy", after sorting, it can get the basic schema list of $S$, it is shown in Table 1. In Table 1, each item is in the form of (Num, $S_m$, loc), where: Num represents the number of item; $S_m$ represents the basic pattern; loc is starting position of the basic pattern $S_m$ in $S$.

Table 1: Basic mode table for $S$

| **Num** | $S_m$ | **loc** |
|---|---|---|
| 1 | x | 3 |
| 2 | xy | 7 |
| 3 | xyzy | 6 |
| 4 | xz | 3 |
| 5 | y | 9 |
| 6 | yx | 7 |
| 7 | yxzxx | 2 |
| 8 | yz | 4 |
| 9 | zxxy | 1 |
| 10 | zyxy | 6 |

Of course, there may be duplicates in the basic pattern, such as: $S'$="xxyxxyxyxy", the basic pattern "x","yx" all appear twice, "x" appears 4 times. In this case, the above basic model table can be improved and designed as shown in Table 2.

In Table 2, each item is still in the form of (Num, $S_m$, input, loc), but where loc is the set of starting positions of the basic pattern $S_m$ in $S$.

Table 2: Basic mode table for $S'$

| **Num** | $S_m$ | **loc** |
|---|---|---|
| 1 | x | 1 3 |
| 2 | xy | 1 3 6 8 |
| 3 | y | 9 |
| 4 | yx | 5 7 |
| 5 | yxx | 3 |

## 3.3 Construct basic frequent pattern prefix tree

**Definition 7.** For a basic pattern table, the basic pattern prefix tree which the basic pattern table corresponds to is a rooted tree, whose path from root to each leaf node represents a basic pattern. Every side of path represents a substring, which represented by edges of the same node do not have the same prefix. A basic pattern is achieved through sequentially arranging substrings represented by edges on the path in the basic pattern prefix tree.

For the basic model Table 1, the basic pattern prefix tree shown as Figure 1 could be constructed by using the above recursive algorithm.

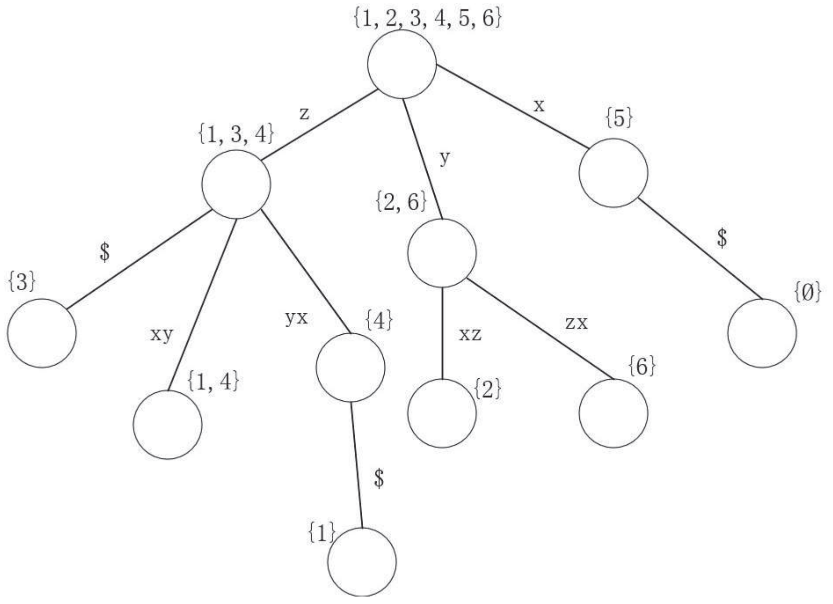

Figure 1: The basic pattern prefix tree corresponding to $S$

In the tree, each node has a set of starting points $\{l_1, l_2, \cdots, l_k\}$. Its meaning is: $x$ is the substring represented by the edge of the node connected to its father node, then $\{l_1, l_2, \cdots, l_k\}$ represents the substring $x$ in the initial position collection of $S$. For root node, $\{l_1, l_2, \cdots, l_k\}$ represents the set of initial position of the empty string in $S$, that is, the collection $\{1, 2, \cdots, |S|\}$ for all positions in $S$.

By the above analysis and its definition, a novel recursive algorithm to construct a basic pattern prefix tree is designed. Starting from root node, the prefix subtree is constructed layer by layer for each vertex. The core is to apply the following node-extend$(S, d)$ extension algorithm to a node. Since we need to calculate displacements for the set of starting positions in this algorithm, we define the addition of sets and numbers for this purpose.

**Definition 8.** Given the set $t = (t_1, t_2, \cdots, t_k)$ and the number l, their addition result is a set: $t + l = \{t_1 + l, t_2 + l, \cdots, t_k + l\}$.

The node's extension algorithm node-extend$(S, d)$ establishes a prefix tree with $d$ as the root for collection $S$ of basic pattern string sets. The basic pattern table of the string $H$ is $S(H)$, then constructing the basic pattern prefix tree of $H$ is the following recursive called Node-extend$(S(H), root)$.

## 3.4 Pruning of the basic pattern prefix tree

Using the basic pattern prefix tree, basic frequent patterns can be mined. Firstly, we trim the basic pattern prefix tree as follows. If loc=$\{l_1, l_2, \cdots, l_k\}$ and loc is less than minloc_sup in a pattern (Num, $S_m$, loc), the node and the subtree rooted from the node can be subtracted. Because in a pattern, if its subpattern is not frequent, then according to the nature of Apriori algorithm, the pattern itself must not be frequent. If a basic pattern is not frequent, all patterns including it would be certainly not frequent. Deleting infrequent nodes and their subtrees in the basic pattern prefix tree will delete the infrequent subpatterns.

Given loc_sup is 2, after pruning the basic pattern prefix tree in Figure 1, the prefix tree which is shown as Figure 2 can be obtained.

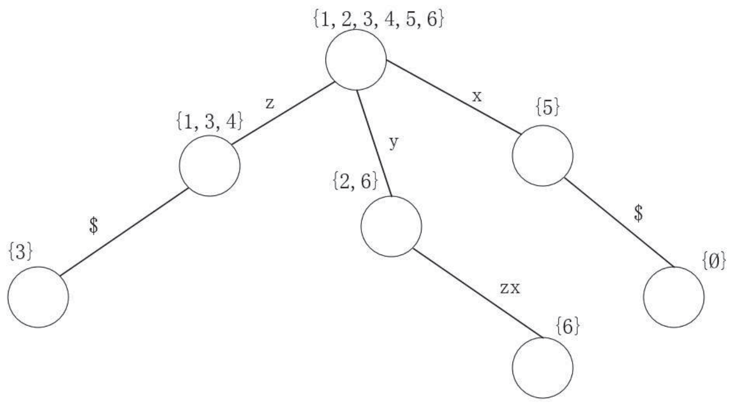

Figure 2: Prefix tree after pruning

From the prefix tree we can know that the original string. The frequent patterns in $S$="aabaababab" are: the frequency of "a" is four, the frequency of "b" is four, the frequency of "c" is two, the frequency of "ab" is two, and the frequency of "ba" is two.

## 3.5 General frequent pattern mining

The basic frequent mode is limited to that there is no letter in the mode that is the same as the start letter. The above basic pattern prefix tree can only mine the basic frequent mode, and cannot mine the general frequent mode. In order to enable it to mine general frequent patterns, we designed a general frequent pattern detection algorithm based on basic sub-pattern prefix tree. The algorithm is a recursive algorithm. From root node, every node is model-expanded layer-by-layer. The kernel of the algorithm will be to apply the expansion algorithm Span$(s, E_s, a)$ to a node. There is a pattern $s$, whose set of end positions in the string $S$ is $Es$. In the basic pattern prefix tree, the last node of the path of the pattern $s$ is $a$. The algorithm could detect all the frequent patterns prefixed by $s$.

In order to detect all frequent patterns in the string $S$, Span$(\Phi, E, root)$ can be called where $E = \{1, 2, \cdots, |S|\}$.In this way, the algorithm starts with an empty string, expands from tree to tree, and records the extended set of end positions at each layer. When extended to a leaf node, it will return to the root node and continue to expand until the extended substring is infrequent.

By synthesizing the above several steps, the single sequence pattern mining for biological data algorithm for single biological sequence mining can be obtained. The algorithm quickly detects all basic frequent patterns firstly, and then implements pattern growth on this basis to obtain all frequent patterns of the sequence.

The flow of the algorithm is defined as:

### Algorithm 1

*Step 1.* A candidate pattern $P$ is extracted from set $C$ in turn, if pattern $P$ is already in the approximate frequent pattern set $S$, the next candidate pattern is selected. Otherwise, the candidate pattern $P$ is compared with other element $C$ $[index]$ of set $C$.

*Step 2.* If Approximation_degree$(P, C[index])$ is greater than minimum approximation, then the number of repetitions of candidate pattern $P$ is increased by 1, $index = index + i$. candidate pattern $P$ continues to compare with element $C$ $[index]$ of set $C$. Otherwise $index = index + 1$, candidate pattern $P$ continues to compare with element $C$ $[index]$ of set $C$.

*Step 3.* Repeat step 2 until index is greater than or equal to the maximum subscript in set $C$, repetition times and support degree of candidate pattern $P$ are counted, if the repetition times and support degree of candidate pattern $P$ satisfy the specified minimum frequent threshold $m$ and the minimum support degree minsupport, then candidate pattern $P$ is a frequent approximation pattern and is added to the approximate frequent pattern set $S$.

*Step 4.* Repeat step 1 until every element in set $C$ is taken out.

## 4 Multiple sequence frequent pattern mining algorithm

Above algorithm for mining single sequence frequent patterns can be extended for mining multiple sequence frequent patterns.

**Definition 9.** Distribution Support was set a collection of biological sequences $D$ and subsequences $T$, the number of sequences containing subsequences $T$ in $D$ is called the distribution support of $T$. The distribution support degree of sub-sequence $T$ is the quantity of sequences containing subsequence $T$ in $D$, denoted as dis_sup$D(T)$.

**Definition 10.** Given biological sequence set $D$ and subsequence $T$, if the sub-sequence support distribution dis_sup$D(T)$ is greater than or equal to mindis_sup, $T$ is said to be a frequent pattern.

For understanding the process of constructing a multiple sequence basic model table and its prefix tree easier, we would give the following definition:

**Definition 11.** (Set vector) There are vectors $T = (T_1, T_2, \cdots, T_n)$ where each $T_i$ is a set, and $T$ is a set vector.

**Definition 12.** (Support function of the set vector) For the set vector $T$, its support function $F(T)$ is defined as the number of non-empty sets in $T$,

$$\mathrm{F(T)} = \left|\{\mathrm{T_i}|\mathrm{T_i} \neq \emptyset; 1 \leq \mathrm{i} \leq \mathrm{n}\}\right|. \tag{1}$$

**Definition 13.** (Operation of set vector) If there are set vectors $S = (S_1, S_2, \cdots, S_n)$ and $T = (T_1, T_2, \cdots, T_n)$, the intersection of the two vectors is defined as:

$$\mathrm{T} \cap \mathrm{S} = (\mathrm{T_1} \cap \mathrm{S_1}, \mathrm{T_2} \cap \mathrm{S_2}, \ldots, \mathrm{T_n} \cap \mathrm{S_n}). \tag{2}$$

**Definition 14.** (Addition of Set Vectors and Numbers) With the set vector $T = (T_1, T_2, \cdots, T_n)$ and the number l, there is

$$T + l = T_1 + l, T_2 + l, \cdots, T_n + l. \tag{3}$$

"+" represents the addition of sets and numbers as defined in Definition 3.

## 4.1 Multiple sequence basic pattern table

For the mining of multiple sequence frequent patterns, firstly we would intercept all the basic patterns in each sequence, and then sort the basic pattern tables of each sequence to obtain the basic pattern table of multiple sequences in the sequence set.

Suppose there are four sequences in the sequence set $D$. $S_1$="$xyzyxz$", $S_2$="$xzyzx$", $S_3$="$yzyxyz$", $S_4$="$xzyxyz$". Algorithms 1 is invoked for each sequence. The basic mode table for each sequence of $D$ is shown as Table 3.

Table 3: Basic patterns of multiple sequences

Basic pattern table of $S_1$

| **Num** | $S_m$ | **loc** |
|---|---|---|
| 1 | xyzy | 1 |
| 2 | xz | 4 |
| 3 | yxz | 3 |
| 4 | yz | 3 |
| 5 | z | 5 |
| 6 | zyx | 6 |

Basic pattern table of $S_2$

| **Num** | $S_m$ | **loc** |
|---|---|---|
| 1 | xy | 4 |
| 2 | xzyz | 1 |
| 3 | y | 5 |
| 4 | yzx | 2 |
| 5 | zxy | 3 |
| 6 | zy | 2 |

Basic pattern table of $S_3$

| **Num** | $S_m$ | **loc** |
|---|---|---|
| 1 | xyz | 3 |
| 2 | yx | 4 |
| 3 | yz | 1 4 |
| 4 | z | 5 |
| 5 | zyxy | 2 |

Basic pattern table of $S_4$

| **Num** | $S_m$ | **loc** |
|---|---|---|
| 1 | xyz | 3 |
| 2 | xzy | 1 |
| 3 | yx | 2 |
| 4 | yz | 4 |
| 5 | z | 5 |
| 6 | zyxy | 3 |

In order to mine multiple sequence frequent patterns, after we get the basic pattern tables of multiple sequences, it must integrate the tables to get a merged multiple sequence pattern table. For example, the basic pattern table of 4 sequences in Table 3 is integrated to obtain basic frequent pattern table of multiple sequence sets similar to Table 1, it is shown in Table 4.

Table 4: Basic pattern table after merging

| **Num** | $S_m$ | **loc_set** |
|---|---|---|
| 1 | x | {1,5},{1,5},{4},{4} |
| 2 | xy | {1},{5},{4},{4} |
| 3 | xyz | {1},∅,{3},{3} |
| 4 | xz | {4},{1},∅,{2} |
| 5 | xzy | ∅,{1},∅,{1} |
| 6 | y | {1,3},{2,5},{1,2,4},{2,4} |
| 7 | yx | {3},∅,{2},{3} |
| 8 | yz | {2},{2},{4},{4} |
| 9 | z | {2,5},{1,3},{1,4},{2,4} |
| 10 | zx | {2},{1},{2},{3} |
| 11 | zyx | {2},∅,{1},{1} |
| 12 | zyxy | ∅,∅,{2},{1} |

From Table 4 basic frequent patterns could be easily discovered. Let (Num,$S_m$,loc_set) be the term of a basic mode $S_m$, then the frequency of $S_m$ is the support function F(loc_set) of the set vector loc-set. For example, if dis_sup is set to 2, then according to Table 4 we can easily tap out the basic frequent patterns: the frequency of "x" is four, the frequency of "xy" is four, the frequency of "xyz" is three, the frequency of "xz" is three, the frequency of "xzy" is two; the frequency of "y" is four, the frequency of "yx" is three, the frequency of "yz" is four; the frequency of "z" is four, the frequency of "zy" is four, the frequency of "zyx" is three, the frequency of "zyxy" is one.

## 4.2 Multiple sequence basic frequent pattern prefix tree

Assuming that the basic pattern table of the multiple sequence set $D$ is $D(H)$, then constructing the basic pattern prefix tree of $D$ can invoke the node's extension algorithm. The algorithm is a recursive algorithm that can be invoked with node-extend $(D(H), root)$. In contrast to the mining of single sequence frequent patterns, the "+" in the "($P$',loc+$|x_i|$)" of the algorithm represents the set vector and number defined in the Definition of 12.

Since the basic patterns in Table 4 are all frequent patterns, the following recursive algorithm is used to construct the following multiple sequence basic frequent pattern prefix tree.

Assume that there are $k$ sequences $1, 2, \cdots, k$ in $D$. In the tree, each side of $x$ represents a character or substring; $x$ is connected to the node in the direction of the leaf is $b$, there is a set vector $T^{(b)} = (T_1^{(b)}, T_2^{(b)}, \ldots, T_k^{(b)})$ on $b$ node represents the mode prefixed with $x$ in each sequence in $D$. $T_i^b = \{l_1, l_2, \ldots, l_{ki}\}$ is the collection of initial positions of $x$ in sequence $S_i$. A basic frequent pattern in Table 4 can be obtained by sequentially arranging the characters represented by the nodes on the path from root to every leaf node.

By synthesizing above several steps, multiple sequence pattern detection for biological data algorithm for multiple biological sequence frequent pattern mining can be obtained. The algorithm quickly detects all basic frequent patterns firstly, then implements pattern growth on this basis to obtain all frequent patterns of sequence set. Algorithm's framework is as follows:

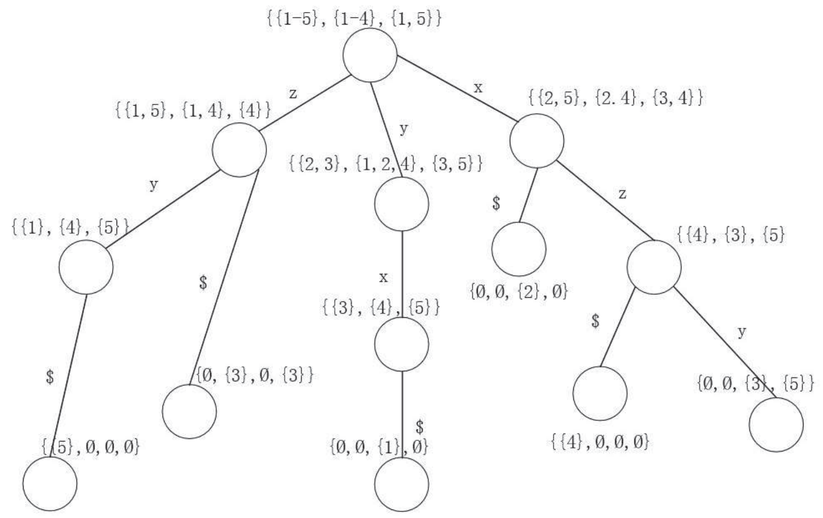

Figure 3: The basic frequent pattern prefix tree corresponding to $D$

### Algorithm 2

```
Multiple sequence pattern mining for biological data algorithm
Input: Biological sequence $S$, minimum local support mindis_sup;
Output: All frequent pattern sets Freq_set;
Begin
1. For each sequence $S_i$ in $S$ do
2. Intercept($S$); /* The intercept algorithm is used to get all the basic pattern string sets $H$ of sequence $S$.*/
3. Sort($S_{mi}, S''_{mi}$); /* Using the algorithm, the basic schema table $S_i(H)$ in order of lexicographic order can be gotten.*/
4. End For
5. Merge the basic schema tables of multiple sequences and sort them.
6. The basic pattern frequency table of multiple sequence sets is constructed and all basic frequent pattern sets $H$ are extracted;
7. Node-extend ($S(H), root$) is recursively called to construct the basic frequent pattern prefix tree $T$;
8. MFreq-Mining($S, root$);
End.
```

## 5 Experimental results and analysis

In order to verify effectiveness of the algorithm, two sets of experiments were compared. First set of experiments compares the traditional algorithm with the proposed two fast and efficient biological sequence pattern detection algorithms. It is mainly used for verifying that these algorithms change with the supportability threshold, which has little impact. The second set of experiments compares the traditional algorithm with the proposed algorithms to verify that the proposed algorithms in this paper have better elapsed time efficiency under the same supportability threshold.

Under the same biological sequence set, with the increasing of support threshold, running time changes of Prior, BioPM, and the single sequence frequent pattern mining algorithm are shown in Figure 4.

From Figure 4, it can be seen that the elapsed time of each of the three algorithms will gradually decrease with increasing of the support threshold. However, overall elapsed time of the single sequence frequent pattern mining algorithm changes steadily, and it is obviously smaller than the other two algorithms. Especially when the threshold is small, the Apriori algorithm will generate a large number of candidate modes and interference modes during the mining process, which inevitably leads to a high complexity of the space-time of the algorithm. Similarly, the BioPM algorithm needs to construct a projection database frequently during the mining process, and there are also a large number of short-term generations, which greatly affects the efficiency of algorithm. The single sequence frequent pattern detection algorithm starts from length of basic pattern. It prunes basic pattern prefix tree in the process, avoiding the generation of a great quantity of short biological patterns and candidate patterns to speed up the operation speed and improve the efficiency.

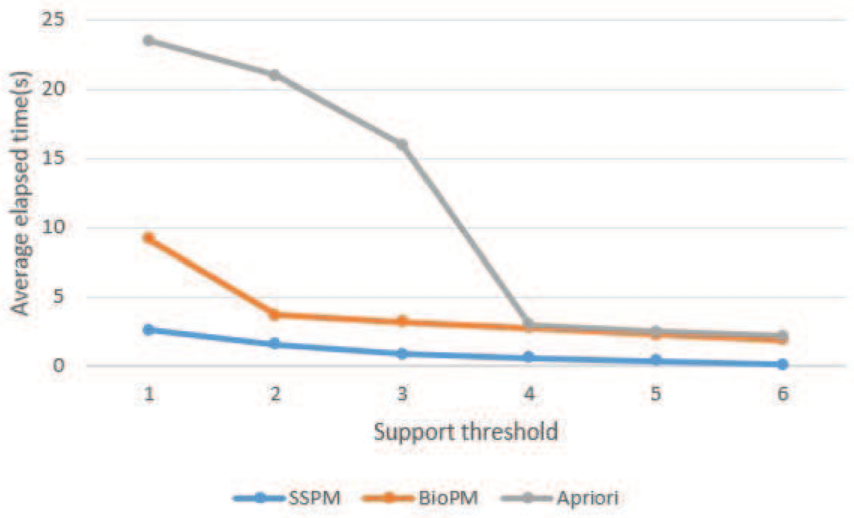

Figure 4: The relation between support threshold and average elapsed time

The following experiment is used to verify that the single sequence frequent pattern mining algorithm has better excavation efficiency under the same support degree threshold. The experimental data were from 10 families in the Pfam protein database. The same or similar part of the length is selected as the test set, and set a 15% support threshold for them. From the 100-sequences, the sequence was gradually increased. The patterns were mined using the BioPM and the single sequence frequent pattern mining algorithm respectively, and the overall time of pattern mining of all sequences was obtained. As the number of sequences increases, the overall elapsed time trends of the two algorithms are shown in Figure 5.

From Figure 5, it can be seen that under certain fixed support threshold conditions, the running time of both the BioPM algorithm and single sequence frequent pattern mining algorithm will increase as the quantity of sequences increases. However, running time of the single sequence frequent pattern mining algorithm is much smaller than that of the BioPM algorithm, and the trend of change is always stable. The reason is that the BioPM algorithm needs to construct a projection database frequently during the mining process and also a large number of short-term generations causes the complexity of the algorithm's space-time and increase the efficiency. However, the single sequence frequent pattern mining algorithm is to start frequent pattern mining from the basic pattern length, which avoids the elapsed time overhead of generating a great quantity of short patterns.

Because classic Apriori algorithm, BioPM algorithm, and multiple sequence frequent pattern

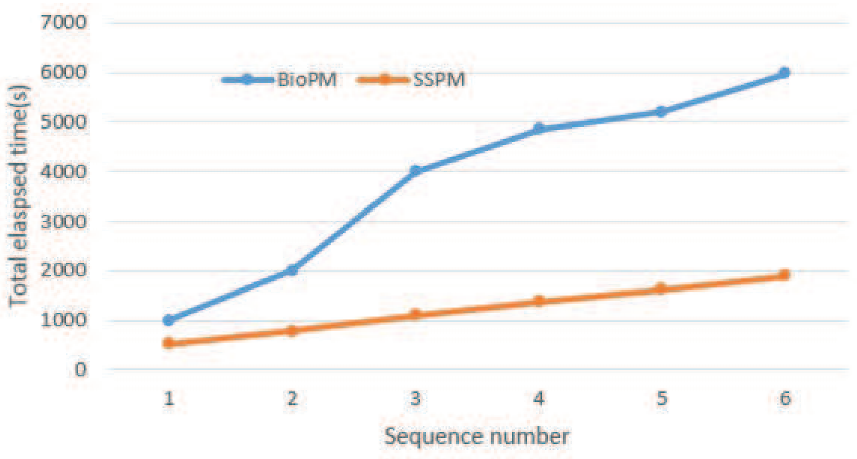

Figure 5: The relationship between the number of sequences and the time

detection algorithm are deterministic algorithm of multiple sequence frequent patterns, the set of frequent patterns mined for the same biological sequence collection and support threshold is exactly the same. In order to verify the performance of the multiple sequence frequent pattern mining algorithm, two sets of experiments were conducted and then compared based on the experimental results. In the first set of experiments, these algorithms are compared with the multiple sequence frequent pattern mining algorithm to verify that the multiple sequence frequent pattern mining algorithm is less affected by changes in the support threshold. The second set of experiments compares the BioPM algorithm, the MBioPM algorithm, and the multiple sequence frequent pattern mining algorithm. It is verified that under the same premise (with the same support threshold), the elapsed time efficiency of the multiple sequence frequent pattern mining algorithm is better, and it has superior excavation performance.

Through the group of experiments, it can be confirmed that the multiple sequence frequent pattern mining algorithm is affected less by the change of the support threshold set by the user. Experimental data were sampled from 3 families in the Pfam protein sequence database (G-alpha, Calici Coat, Glyco_hydro_19). A subset of the same or similar lengths (out of a total of 50) are selected as test sets to ensure that this algorithm is suitable for different types of sequence sets. Under each of the specific support conditions, 50 sequences of data were tested using four algorithms respectively. Under the same set of biological sequences, as the support threshold is increased, the elapsed time changes of the Prior, BioPM, MBioPM, and the multiple sequence frequent pattern mining algorithm are shown in Figure 6.

From Figure 6, it can be seen that the elapsed time of the four algorithms will gradually decrease as support threshold increases. However, the trend of overall elapsed time of the multiple sequence frequent pattern mining algorithm is relative stability, especially when the threshold becomes small, the elapsed time will be obviously less than the other three algorithms. The main reason for this result is that the Apriori algorithm will generate a great quantity of candidate modes and interference patterns during mining process, which will inevitably lead to a higher complexity of the space-time algorithm; the BioPM algorithm requires frequent construction of the projection database during the mining process, and there are also a large number of short-mode generations, which greatly affect the efficiency of the algorithm; the multiple sequence frequent pattern detction algorithm is started from length of basic model to avoid generating a great quantity of short biological models, through the merge operation can quickly basic frequent pattern detection and use basic frequent pattern prefix tree for pattern growth to improve

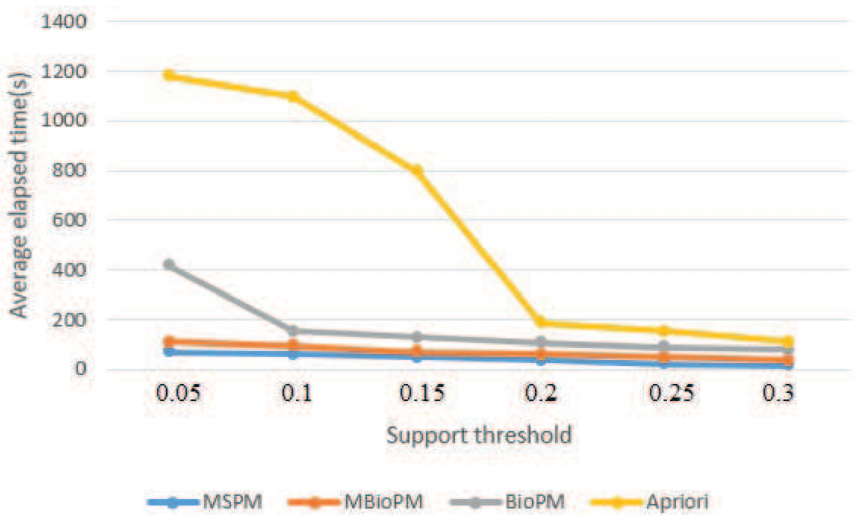

Figure 6: The relation between support threshold and average elapsed time

efficiency.

The set of experiments is used to verify that the multiple sequence frequent pattern mining algorithm has better excavation efficiency under the same support degree threshold. The experimental data were from 10 families in the Pfam protein database. The same or similar part of the length is selected as the test set, and set a 15% support threshold for them. From the 100-sequences, the sequence was gradually increased. The patterns were mined using the BioPM and the multiple sequence frequent pattern mining algorithm respectively, and the overall time of pattern mining of all sequences was obtained. As the number of sequences increases, the overall elapsed time trends of the two algorithms are shown in Figure 7.

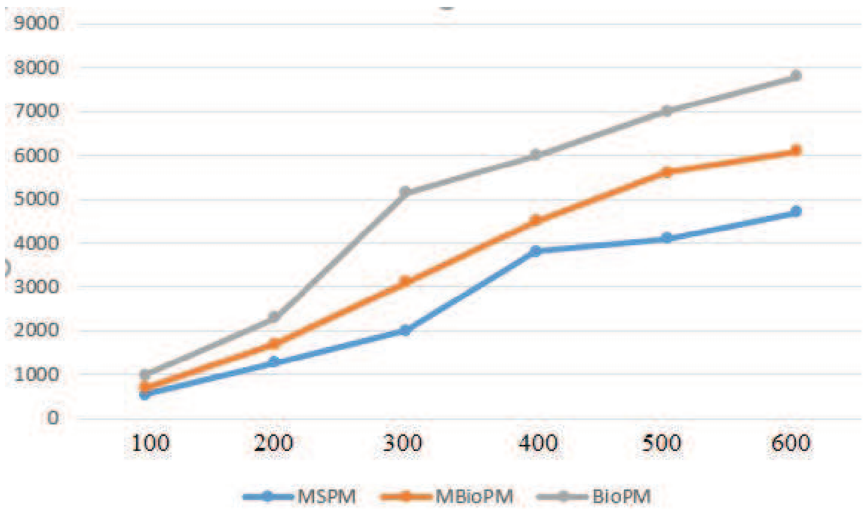

Figure 7: The relationship between the number of sequences and the time

From Figure 7, under the conditions of a certain fixed support threshold, the elapsed time of the three algorithms for comparison will increase without exception with the increase of the number of sequences. However, it can be noted that the elapsed time of the multiple sequence frequent pattern mining algorithm is more stable, especially when the model is longer, the overall time is significantly less than the other two algorithms. It is due to the fact that the BioPM algorithm needs frequent construction of the projection database during the mining process, and there are also a large number of short-term generations that cause the complexity of the algorithm's space-time and increase the efficiency. For the MBioPM algorithm, since each time a frequent pattern of $k$-class length is mined, the existing pattern needs to be compared with the buffer pattern one by one. At the same time, the buffer area needs to be cleared and opened repeatedly during the pattern growth. These will definitely reduce the speed of the algorithm. The multiple sequence frequent pattern mining algorithm starts frequent pattern mining from the basic pattern, avoids the runtime overhead of generating a great quantity of short patterns, and at the same time uses basic frequent pattern prefix trees for pattern growth after fast mining to obtain basic frequent pattern string sets to avoid "interference patterns". The generation of the mining efficiency of the algorithm has been greatly improved.

## 6 Conclusion

According to the characteristics of biological sequence patterns and mining requirements, fast and effective biological single sequence and multiple sequence pattern mining algorithms are proposed in this paper, respectively. First of all, the concept of the basic pattern of the sequence is defined. The algorithm can connect the basic patterns of the sequence to obtain the basic pattern table of the sequence. For multiple sequences, the basic pattern table of multiple sequences can be obtained through the merge operation. Based on the basic model, overall basic frequent patterns could be easily detected. In order to mine general frequent patterns, a basic frequent pattern prefix tree is constructed based on a single or multiple sequence basic frequent pattern table, and the general frequent pattern mining is quickly implemented using the set vector operation, which improves the mining efficiency. At the same time, the use of pruning technology avoids producing a great quantity of unnecessary short patterns. The analysis results of the experiment show that the two proposed algorithms significantly improve the mining efficiency of the biological sequence model, especially can get better mining results in the case of a small support threshold.

## Funding

This research was supported by the Natural Science Foundation of Hebei Province (H2019201378).

## Bibliography

[1] Apostolico, A.; Preparata, F. P. (1983). Optimal off-line detection of repetitions in a string, *Theoretical Computer Science*, 22(3), 297-315, 1983.

[2] Ben-Hur, A.; Brutlag, D. (2003). Remote homology detection: A motif based approach, *Bioinformatics*, 19 Suppl 1(1), 26-33, 2003.

[3] Benkaddour, M. K., Bounoua, A. (2017). Feature extraction and classification using deep convolutional neural networks, PCA and SVC for face recognition, *Traitement du Signal*, 34(1-2), 77-91, 2017.

[4] Cardon, L. R.; Stormo, G. D. (1992). Expectation maximization algorithm for identifying protein-binding sites with variable lengths from unaligned DNA fragments, *Journal of Molecular Biology*, 223(1), 159-170, 1992.

[5] Chen, L.; Liu, W. (2013). Frequent patterns mining in multiple biological sequences, *Computers in Biology & Medicine*, 43(10), 1444-1451, 2013.

[6] Crochemore, M.; Ilie, L. (2008). Maximal repetitions in strings, *Journal of Computer & System Sciences*, 74(5), 796-807, 2008.

[7] Dong, T. (2017). Assessment of data reliability of wireless sensor network for bioinformatics, *International Journal Bioautomation*, 21(1), 241-250, 2017.

[8] Jiang, Q.; Li, S.; Guo, S. (2011). A new model for finding approximate tandem repeats in DNA sequences, *Journal of Software*, 6(3), 386-394, 2011.

[9] Karmaker, S.; Ruhi, F. Y.; Mallick U. K. (2018). Mathematical analysis of a model on guava for biological pest control, *Mathematical Modelling of Engineering Problems*, 5(4), 427-440, 2018.

[10] Kurtz, S.; Choudhuri, J. V.; Ohlebusch, E.; Schleiermacher, C.; Stoye, J.; Giegerich, R. (2001). REPuter: the manifold applications of repeat analysis on a genomic scale, *Nucleic Acids Research*, 29(22), 4633-4642, 2001.

[11] Lawrence, C. E.; Reilly, A. A. (1990). An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences, *Proteins Structure Function & Bioinformatics*, 7(1), 41-51, 1990.

[12] Liao, C. C.; Chen, M. S. (2014). DFSP: A Depth-First SPelling algorithm for sequential pattern mining of biological sequences, *Knowledge* & *Information Systems*, 38(3), 623-639, 2014.

[13] Makalowski, W. (2003). Not junk after all, *Science*, 300(5623), 1246-1247, 2003.

[14] Pasquier, N.; Bastide, Y.; Taouil, R.; Lakhal, L. (1999). Efficient mining of association rules using closed itemset lattices, *Information Systems*, 24(1), 25-46, 1999.

[15] Rubin, E. M.; Lucas, S.; Richardson, P. (2004). Finishing the euchromatic sequence of the human genome, *Nature*, 431(7011), 931-945, 2004.

[16] Saqhai Maroof, M. A.; Yang, G. P.; Biyashev, R. M.; Maughan, P. J.; Zhang, Q. (1996). Analysis of the barley and rice genomes by comparative RFLP linkage mapping, *Theoretical & Applied Genetics*, 92(5), 541-551, 1996.

[17] Shapiro, J. A.; Von, S. R. (2005). Why repetitive DNA is essential to genome function, *Biological Reviews*, 80(2), 227-250, 2005.

[18] Stansfield, W.; King, R.; Mulligan, P. (2006). A dictionary of genetics, *Yale Journal of Biology & Medicine*, 75(4), 236-245, 2006.

[19] Won, J. I.; Park, S.; Yoon, J. H.; Kim, S. W. (2006). An efficient approach for sequence matching in large DNA databases, *Journal of Information Science*, 32(1), 88-104, 2006.