Frequent pattern discovery with tri-partition alphabets

Fan Min[a], Zhi-Heng Zhang[b,a], Wen-Jie Zhai[a], Rong-Ping Shen[a]

[a]*School of Computer Science, Southwest Petroleum University, Chengdu 610500, China*
[b]*School of Science, Southwest Petroleum University, Chengdu 610500, China*

Abstract

The concept of patterns is the basis of sequence analysis. There are various definitions of patterns in Biological data, text and time series analysis. Inspired by the protein tri-partition and three-way decision (3WD), we propose here a frequent pattern discovery algorithm for a new type of pattern by dividing the alphabet into strong/medium/weak parts. The pattern is more general and flexible than existing ones and is therefore more interesting in applications. Experiments are undertaken on data in various fields including protein sequences mining, stock price time series analysis, and faked Chinese text keywords mining. The results show that tri-patterns are more meaningful and desirable than the existing four types. This study enriches the semantics of sequential pattern discovery and the application fields of 3WD.

Keywords: Pattern discovery; Sequence; Tri-pattern; Tri-wildcard.

1. Introduction

Pattern discovery has been widely studied in machine learning, such as image recognition [16] and text analysis [45]. A sequential pattern [7] is a (totally or partially ordered) subsequence of a transactional dataset [2], a symbolic sequence [39], a numeric time series [35], etc. Sequential pattern discovery (also called pattern mining) [2, 50] refers to discovering all frequent sequential patterns from these data. For transactional datasets such as retailing/market-basket and planning [15], a pattern is a sequence of events, where each event is represented by an itemset. This kind of pattern has the ability to extend both horizontally and vertically, and therefore producing general association rules. For symbolic sequences such as protein [33] and text [32], a pattern is a subsequence of characters, also called a string. It is simple however meaningful to represent codons [31] or keywords. For numeric time series such as stock price [34] and petroleum production [35], where the trend of fluctuation is essential to stock holders and the petroleum cooperation, a pattern is a subsequence of real values.

*Corresponding author. Tel.: +86 135 4068 5200.
Email: minfanphd@163.com.

One interesting research direction of symbolic sequential pattern discovery involves the generalization of patterns. A plain pattern is a subsequence that should be exactly matched. A wildcard [4], which is also called a motif [23, 51], matches any character in the alphabet. Consequently, a wildcard gap [4] matches any subsequence within the length constraint. In this way, a pattern with wildcard gaps is able to handle the noise or shift [53]. To enrich the semantics of the pattern, the alphabet is divided into the weak and strong parts. A weak-wildcard gap [43] matches a subsequence of weak characters. Consequently, a pattern with weak-wildcard gaps [43] not only ignores weak characters, but also maintains strong ones.

In this paper, we introduce a more general and flexible type called tri-pattern inspired by 3WD [56]. The alphabet is partitioned into three parts Γ, Λ, and Ω corresponding to strong, medium, and weak characters, respectively. In this situation, a tri-wildcard gap (N, M) matches any sequence of medium/weak characters with length between N and M. A tri-pattern is a sequence of strong/medium characters containing periodic tri-wildcard gaps, where periodic indicates that the gaps between any two adjacent characters are identical. The idea of tri-partition is widely adopted in applications. Biologists partition the human amino acid alphabet into essential, conditional essential, and nonessential amino acids [1]. Petroleum expert partition the oil production fluctuation into significant, insignificant, and minor change. Tri-partition of situations and actions [21, 27, 54] also have attracted much research interests in data mining, especially 3WD [9, 14, 40]. To the best of our knowledge, however, tri-partition of the alphabet has not been considered in the pattern discovery society.

Tri-patterns are more general than the four existing types. Let tri-patterns be Type I, plain patterns be Type II, patterns with periodic wildcard gaps [4] be Type III, patterns with periodic weak-wildcard gaps [43] be Type IV, and strong patterns with periodic weak-wildcard gaps [43] be Type V. Types II through V are the special cases of Type I on for $\Lambda = \Omega = \emptyset$, $\Lambda = \Sigma$, $\Omega = \emptyset$, and $\Lambda = \emptyset$, respectively. Similar to existing frequent pattern discovery problems, a new problem for Type I is defined. With the Apriori property of the new problem, an Apriori algorithm is designed for efficient pattern discovery and tree pruning.

We compare these five types on three application domains to reveal the universality of the new pattern. The first is the human protein sequence where essential, conditional essential, and nonessential amino acids correspond to strong, medium, and weak characters, respectively. Twenty sequences are concatenated to mine frequent patterns. These patterns and their popularity in the original sequences are analyzed. Tri-patterns are the most meaningful, while other types especially Type II suffer from too many weak characters. For the pattern popularity, tri-patterns have the best minimal and the second best average performance.

The second is oil well daily production time series. A coding table is designed to convert the numeric time series into a nominal sequence. Significant, insignificant, and minor changes correspond to strong, medium, and weak characters, respectively. The data of an oil well for two years is adopted to mine frequent patterns, which are finally matched in the original time series. Compared to

distance-based approaches that analyze numeric time series directly, the coding and mining approach handles minor difference easier. Observation on the original time series shows that tri-patterns match some similar subsequences with minor differences. In contrast, four existing types either excludes some similar subsequences, or includes dissimilar ones.

The third is faked Chinese text. Notional words, function words, and special characters correspond to strong, medium, and weak characters, respectively. Four sets of text are collected in news, novel, history, and law. The background is that some text creators distribute advertisements of illegal stuff such as faked money or invoice. They may insert some characters into the text, which makes the text still readable but hard for automated analysis. Our purpose is to extract keywords from the faked text, which are also keywords in the original text. This problem is closely relevant to text fuzzy matching and keywords extraction. With tri-patterns, the algorithm obtains the best accuracy in mining the keywords from the faked text.

The remainder of this paper is organized as follows. Section 3 defines the new type of patterns, analyzes the relationships among five types, and presents the new pattern discovery problem. Section 4 demonstrates the Apriori property and proposes the new algorithm. Section 5 explains experimental settings and results on three kinds of real-world data. Finally, concluding remarks are discussed in Section 6.

2. Related work

In this section, we present some related work concerning 3WD and sequential pattern discovery.

2.1. Three-way decision

In rough set theory [36], according to the equivalence relation, the universe is partitioned into three disjoint regions. The positive (negative, boundary) region is the set of objects that are definitely (definitely not, possibly) members of the target set. However, this strict division often hinders its application in practice. Probabilistic rough set models such as decision-theoretical rough sets (DTRS) [52, 58] and variable precision rough sets [65] are introduced for this issue. Among them, DTRS is a sound theory based on Bayesian decision making [11]. The required parameters are systematically determined based on costs of various decisions [55, 58].

3WD has made a big step forward through making itself a methodology of divide and conquer [57] rather than a concrete technique. With this methodology, one task is to construct a trisection or a tri-partition of the universal set. The other is to act upon objects in one or more regions by developing appropriate strategies. 3WD is a class of effective ways and heuristics commonly used in human problem solving and information processing.

Recently, various theories are inspired by 3WD. Three-way formal concept analysis [38] is constructed with the three-way operators and their inverse.

Three-way cognition computing [21] focuses on concept learning via multi-granularity from the viewpoint of cognition. Three-way fuzzy sets [6] constructs a three-way, three-valued, or three-region approximation with a pair of thresholds on the fuzzy membership function. Three-way decisions space [13] unifies decision measurement, decision conditions and evaluation functions. Sequential three-way decisions [20] is an iteration process that eventually leads to two-way decisions. There are also some generalized three-way decision models [22, 24].

There are also various 3WD applications. Three-way recommender system is applied to both classification [61] and regression [62] of user ratings. Three-way active learning [29, 48] achieves a tradeoff between teacher cost for acquiring class labels and misclassification costs. Three-way clustering [59] introduces a new strategy for overlapping clustering based on the DTRS model. Three-way spam filtering [63] reduces the email misclassification costs. Three-way face recognition [19] develops a sequential strategy to address the imbalanced misclassification cost and insufficient high quality facial image information.

2.2. Sequential pattern discovery

The sequence data can be usually categorized according to the elementary data type, namely numeric [18], symbol [10] and image [46]. Most of time series such as weather [18] and stock [41] are numerical. However, symbolic sequence data usually contains text [37] and biological data [10]. Image sequence data are commonly collected from traffic [46] and medical videos [64].

The definition of the pattern varies according to the practical problem. For numeric time series, a pattern is a segment that reveals certain trends [47] or periodicity [26]. The key issue of pattern matching is the similarity model. For symbolic sequences, a pattern is a subsequence representing keywords [28] or associations [3]. Wildcards [4] and fuzzy matching are often used to find more candidates. For videos, a pattern is part of the images representing vehicles, pedestrians and obstacles for traffic [42], or viscera and mass [44] for medical image sequence.

Pattern discovery algorithms are designed for different types of data and patterns. Due to the exponential nature of respective problems, efficiency is the main concern of these algorithms. For trend and periodicity analysis, various regression analysis methods [49] proposed. For symbolic sequences pattern mining, Apriori [2] and SPADE [60] algorithms are often employed for efficient pruning. For image processing, artificial neural network [25] and random walk [17] are often employed to handle complex situations. It is worthy noted that numeric time series are usually converted into symbolic ones [5, 43] to facilitate pattern mining.

3. Problem statement

In this section, we first propose a general type of pattern which will be called tri-patterns. Second we discuss the frequency of tri-pattern. Third we analyze its relationships with four existing types. Finally we define the problem

of frequent tri-pattern discovery. Table 1 lists notations used throughout the paper.

Table 1: Notations

Notation	Meaning
Σ	An alphabet
$T = (\Gamma, \Lambda, \Omega)$	The strategy of tri-partition
Γ	The set of strong characters
Λ	The set of medium characters
Ω	The set of weak characters
S	A sequence
k	The length of S
$sub(S, i, j)$	A sub-sequence of S from position i to j
N	The gap lower bound
M	The gap upper bound
ψ	A tri-wildcard for P
P	A tri-pattern
m	The length of P
$w(N, M) = (N, M)$	The tri-wildcard gap for P
$sub(P, I)$	A sub-pattern of P on I
$\mathcal{P}$	The set of all tri-patterns
$\mathcal{P}_i$	The set of all tri-patterns with length i
$I = \langle i_1, i_2, \cdots, i_m \rangle$	A position sequence
$sup(P, S)$	The number of I which P actually matches S
$ofs(P, S)$	The number of I which P probably matches S
$fr(P, S)$	The frequency of P, which is $\frac{sup(P,S)}{ofs(P,S)}$
ρ	The user-specified frequency threshold

3.1. Tri-patterns

The starting point of sequential pattern discovery is an alphabet, which is fundamental in both natural and formal languages.

Definition 1. An **alphabet** Σ is a non-empty finite set, whose elements are called **characters**.

According to the Σ, a sequence can be defined as a series of ordered characters.

Definition 2. Any $S = s_1 s_2 \dots s_k$ where $s_i \in \Sigma$ for any $1 \leq i \leq k$ is a **sequence**, which is also called a string. The length of the sequence is $|S| = k$.

For example, $\Sigma = \{$o, a, b, c$\}$ is an alphabet, $S =$ aboacbb is a sequence on the alphabet, and $|S| = 7$.

One contribution of this work is to partition the alphabet into three disjoint subsets, which are enriched with certain semantics respectively.

Definition 3. $T = (\Gamma, \Lambda, \Omega)$ where $\Sigma = \Gamma \cup \Lambda \cup \Omega$ and $\Gamma \cap \Lambda = \Gamma \cap \Omega = \Lambda \cap \Omega = \emptyset$ is a tri-partition of Σ. Γ is the set of **strong characters**, Λ is the set of **medium characters**, and Ω is the set of **weak characters**.

This partition strategy $T = (\Gamma, \Lambda, \Omega)$ is used in two ways. One is the definition of the tri-wildcard.

Definition 4. A tri-wildcard ψ is a symbol that matches any character in $\Lambda \cup \Omega$. A **tri-wildcard gap** $w(N, M)$ is a sequence of tri-wildcards with minimal size N and maximal size M.

Example 1. *Let $\Sigma = \{o,\ a,\ b,\ c\}$, $\Gamma = \{c\}$, $\Lambda = \{a,\ b\}$, and $\Omega = \{o\}$. ψ matches a, b, o except c. The gap with size 2 ($M - N + 1 = 2$) matches a, ab, aa, etc, but not ca, oaa, etc. Gap $w(N, M)$ is also denoted by (N, M) in a pattern.*

The other is the definition of tri-pattern.

Definition 5. A **tri-pattern** (Type I pattern) is a sequence of strong/medium characters and periodic tri-wildcard gaps that begins and ends with tri-wildcards matching medium or weak characters. It has the following form:

$$P = p_1(N, M)p_2(N, M) \dots (N, M)p_m, \tag{1}$$

where m is the length of the pattern, $p_i \in \Gamma \cup \Lambda$ for any $1 \leq i \leq m$.

Moreover, the tri-wildcard gap (N, M) is the position constraint for each p_i, $i \in [2, m]$. Namely, for each pair $p_i(N, M)p_{i+1}$, $1 \leq i \leq m$, the nearest distance is N, and the furthest is M. When N and M is known, the pattern is also denoted as $P = p_1 p_2 \dots p_m$ for brevity.

Example 2. *Let Σ, Γ, Λ and Ω be the same as Example 1. P_1 = a(0,1)c(0,1)b = acb is a valid pattern. P_2 = a(0,1)o(0,1)b = aob is an invalid pattern because $p_2 = o \notin \Gamma \cup \Lambda$.*

Figure 1 shows the three different actions for respective kinds of characters defined by Definition 3. The strong elements of Γ are only used to form the characters p_i of tri-pattern. The weak characters of Ω are only used to match the tri-wildcard ψ. The medium characters of Λ not only can be used to form the characters p_i of tri-pattern, but also can be used to match the tri-wildcard.

3.2. Frequency

In order to obtain the frequency of tri-pattern P, we first adopt the definition in [30] as follows.

Definition 6. [30] Given a sequence S with length k, gap lower bound N and upper bound M, an offset sequence $I = \langle i_1, i_2, \dots, i_m \rangle$ is a **position sequence** iff

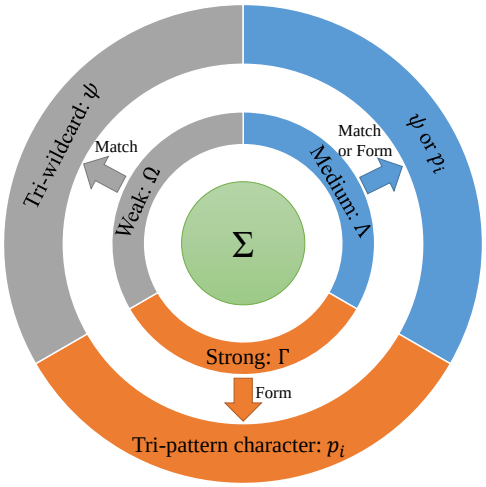


Figure 1: Tri-partition strategies and actions.

1) $1 \leq i_1 \leq k$; and
2) $N \leq i_{j+1} - i_j - 1 \leq M$ for all $1 \leq j \leq m-1$.

Consequently, the following proposition holds.

Proposition 1. *[30] Let $S = s_1 s_2 \dots s_k$ and $P = p_1(N,M)p_2(N,M)\dots(N,M)p_m$, the number of all position sequences is*

$$ofs(P,S) = k(M-N+1)^{m-1}. \tag{2}$$

The key issue is how the tri-pattern matches the sequence.

Definition 7. Let $S = s_1 s_2 \dots s_k$ be a sequence and $P = p_1(N,M)p_2(N,M)\dots(N,M)p_m$ be a tri-pattern. If there exists a position sequence $I = \langle i_1, i_2, \dots, i_m \rangle$ so that

1) $1 \leq i_1 < i_2 < \dots < i_m \leq k$;
2) $s_{i_j} = p_j$ for all $1 \leq j \leq m$;
3) $N \leq i_{j+1} - i_j - 1 \leq M$ for all $1 \leq j \leq m-1$; and
4) $\forall i_j < c < i_{j+1}$ where $1 \leq j \leq m-1$, $s_c \in \Lambda \cup \Omega$,

I is called an **occurrence** of P in S.

We also say that P matches S at I. The number of occurrences/matches of P in S will be denoted by $sup(P,S,\Gamma,\Lambda,\Omega)$. It is also called the support of P in S.

Finally, we define the frequency of a tri-pattern in a sequence.

Definition 8. Let $S = s_1 s_2 \dots s_k$ be a sequence and $P = p_1(N,M)p_2(N,M)\dots(N,M)p_m$ be a tri-pattern. The frequency of P in S is

$$fr(P,S,\Gamma,\Lambda,\Omega) = \frac{sup(P,S,\Gamma,\Lambda,\Omega)}{ofs(P,S)}. \tag{3}$$

When Γ, Λ, and Ω are known, $fr(P, S, \Gamma, \Lambda, \Omega)$ and $sup(P, S, \Gamma, \Lambda, \Omega)$ are abbreviated as $fr(P, S)$ and $sup(P, S)$, respectively. Moreover, we have the following example.

Example 3. *Let* $\Sigma = \{o, a, b, c\}$, $\Gamma = \{c\}$, $\Lambda = \{a, b\}$, $\Omega = \{o\}$, $S = acbcbacaocbbba$, $N = 0$, $M = 1$, *and* $P_1 = acbb$.

$I_1 = \langle 6, 7, 11, 12\rangle$ *is not a position sequence because the gap between positions 7 and 11 is 3, which is greater than* M. $I_2 = \langle 1, 2, 3, 5\rangle$ *is not an occurrence of* P_1 *in* S *because* $s_4 = c \notin \Lambda \cup \Omega$. $I_3 = \langle 8, 10, 11, 12\rangle$, $I_4 = \langle 8, 10, 11, 13\rangle$ *and* $I_5 = \langle 8, 10, 12, 13\rangle$ *are three occurrences.*

$fr(P_1, S) = \frac{3}{14\times 2^3} = \frac{3}{112} = 0.0268$.

3.3. Relationships with existing patterns

There are various definitions of existing patterns. We list them and discuss their relationships with the tri-pattern (Type I).

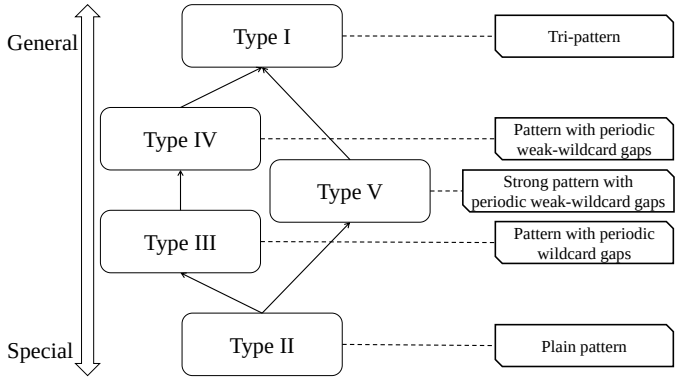


Figure 2: Relationships among different pattern types.

Plain patterns (Type II pattern) are the simplest type of patterns. They do not enable any gap between adjacent characters. Naturally, a plain pattern only permits accurate matching.

Proposition 2. *Type II pattern is a special case of Type I pattern where* $N = M = 0$.

Wildcard was introduced to obtain certain flexibility.

Definition 9. [4] A wildcard ϕ is a special symbol that matches any character in Σ. A **wildcard gap** $g(N, M)$ is a sequence of wildcards of minimal size N and maximal size M.

The concept of pattern with periodic wildcard gaps (Type III pattern) is similar to the Type I pattern. The only difference is that a Type III pattern employs wildcard gaps instead of weak-wildcard gaps. It does not need a partition of Σ since the wildcard matches any character.

Table 2: Four special cases of tri-pattern (Type I)

Type	Property	Statement
II	$N = M = 0$	No gap
III	$\Gamma = \Omega = \emptyset$	ψ matches any character
IV	$\Omega = \emptyset$	ψ matches any character in Λ
V	$\Lambda = \emptyset$	Characters matched by ψ cannot be included in P

Proposition 3. *Type III pattern is a special case of Type I pattern where* $\Lambda = \Sigma$.

Naturally, in this case $\Gamma = \Omega = \emptyset$.

The weak-wildcard gap was introduced in [43] to control the flexibility of a pattern. However the concept is slightly different from this work. To define it, the alphabet is divided into two disjoint subsets, i.e., the set of strong characters Γ and the set of medium characters Λ. In this case, a tri-wildcard φ is a special symbol that matches any character in $\Lambda \subseteq \Sigma$. In other words, φ does not match any character in Γ. If we replace wildcard gaps of Type II patterns with weak-wildcard gaps based on φ, we obtain Type IV patterns. They are called pattern with periodic weak-wildcard gaps in [43].

Proposition 4. *Type IV pattern is a special case of Type I pattern where* $\Omega = \emptyset$.

Sometimes Σ is divided into two disjoint subsets, i.e., the set of strong characters Γ and the set of weak characters Ω. Strong characters can be used to form patterns, while weak characters cannot. The tri-wildcard φ matches any character in Ω. In this case we have the concept of strong patterns with periodic tri-wildcard gaps (Type V patterns) [43].

Proposition 5. *Type V pattern is a special case of Type I pattern where* $\Lambda = \emptyset$.

The relationships among all five types of patterns are illustrated in Figure 2 and listed in Table 2.

For completeness, one may discuss other three types where $\Gamma = \emptyset$, $\Gamma = \Lambda = \emptyset$, and $\Lambda = \Omega = \emptyset$. In fact, the case $\Gamma = \Lambda = \emptyset$ is meaningless since no pattern can be formed. The case $\Lambda = \Omega = \emptyset$ is equivalent to $N = M = 0$ since no wildcard exists. Hence we will not discuss them further.

3.4. Proposed problem

Now we propose the pattern discovery problem of tri-pattern.

Problem 1. *Frequent tri-pattern discovery.*
***Input**: Σ, Γ, Λ, Ω, S, ρ, N, M.*
***Output**: $\mathcal{P} = \mathcal{P}(\Sigma, \Gamma, \Lambda, \Omega, S, \rho, N, M)$.*
***Constraint**: any $P \in \mathcal{P}$, $fr(P, S) \geq \rho$.*

Σ, Γ, Λ, Ω, S, N, M and P have been defined in Definitions 1-5. ρ is the frequent threshold of tri-pattern P. Naturally, $\mathcal{P}$ varies with different parameters. When Λ changes, we have the following property.

Property 1. *Let* $\Lambda_1 \subset \Lambda_2 \subseteq \Sigma$,

$$\begin{aligned} &\mathcal{P}(\Sigma, \Sigma - \Lambda_1 - \Omega, \Lambda_1, \Omega, S, \rho, N, M) \\ &\subseteq \mathcal{P}(\Sigma, \Sigma - \Lambda_2 - \Omega, \Lambda_2, \Omega, S, \rho, N, M). \end{aligned} \tag{4}$$

Proof. Let $P = p_1 p_2 \dots p_m \in \mathcal{P}(\Sigma, \Sigma - \Lambda_1 - \Omega,\ \Lambda_1, \Omega, S, \rho, N, M)$. Suppose further that $I = \langle i_1, i_2, \dots, i_m \rangle$ is an occurrence of P on S given the tri-partition $T_1 = (\Sigma - \Lambda_1 - \Omega, \Lambda_1, \Omega)$.

Condition (1) According to Definition 5, $\forall 1 \leq i \leq m$, $p_i \in (\Sigma - \Lambda_1 - \Omega) \cup \Lambda_1$ $= \Sigma - \Omega = (\Sigma - \Lambda_2 - \Omega) \cup \Lambda_2$.

Condition (2) Since $\Lambda_1 \subset \Lambda_2$, $\forall i_j < c < i_{j+1}$ where $1 \leq j \leq m-1$, $s_c \in \Lambda_1 \cup \Omega \subseteq \Lambda_2 \cup \Omega$.

Combining conditions (1)-(2) and Definition 7, I is also an occurrence of P on S given the tri-partition $T_2 = (\Sigma - \Lambda_2 - \Omega, \Lambda_2, \Omega)$. Hence $sup(P, S, \Sigma - \Lambda_2 - \Omega, \Lambda_2, \Omega) \geq sup(P, S, \Sigma - \Lambda_1 - \Omega, \Lambda_1, \Omega)$. Because $ofs(P, S)$ is independent from the alphabet partition, we have $fr(P, S, \Sigma - \Lambda_2 - \Omega, \Lambda_2, \Omega) \geq fr(P, S, \Sigma - \Lambda_1 - \Omega, \Lambda_1, \Omega) \geq \rho$. Consequently $P \in \mathcal{P}(\Sigma, \Sigma - \Lambda_2 - \Omega, \Lambda_2, \Omega, S, \rho, N, M)$. Therefore, Equation (4) holds.

Property 1 shows that if we set more strong characters as medium characters, the number of patterns never decrease. As an extreme case, we have the following corollary:

Corollary 1.

$$\begin{aligned} &\mathcal{P}(\Sigma, \Sigma - \Omega, \emptyset, \Omega, S, \rho, N, M) \\ &\subseteq \mathcal{P}(\Sigma, \Sigma - \Lambda - \Omega, \Lambda, \Omega, S, \rho, N, M). \end{aligned} \tag{5}$$

According to Proposition 5, Type V pattern is a special case of Type I patterns where $\Lambda = \emptyset$. Corollary 1 shows that there are more Type I patterns than Type V ones.

Now we look at how the number of patterns changes w.r.t. the change of Ω.

Property 2. *Let* $\Omega_1 \subset \Omega_2 \subseteq \Sigma$,

$$\begin{aligned} &\mathcal{P}(\Sigma, \Gamma, \Sigma - \Gamma - \Omega_1, \Omega_1, S, \rho, N, M) \\ &\supseteq \mathcal{P}(\Sigma, \Gamma, \Sigma - \Gamma - \Omega_2, \Omega_2, S, \rho, N, M). \end{aligned} \tag{6}$$

Proof. Let $P = p_1 p_2 \dots p_m \in \mathcal{P}(\Sigma, \Gamma, \Sigma - \Gamma - \Omega_2, \Omega_2, S, \rho, N, M)$. Suppose further that $I = \langle i_1, i_2, \dots, i_m \rangle$ is an occurrence of P on S given the tri-partition $T_2 = (\Gamma, \Sigma - \Gamma - \Omega_2, \Omega_2)$.

Condition (1) According to Definition 5, $\forall 1 \leq i \leq m$, $p_i \in \Gamma \cup (\Sigma - \Gamma - \Omega_2)$ $= \Sigma - \Omega_2 \subset \Sigma - \Omega_1 = \Gamma \cup (\Sigma - \Gamma - \Omega_1)$.

Condition (2) $\forall i_j < c < i_{j+1}$ where $1 \leq j \leq m-1$, $s_c \in (\Sigma - \Gamma - \Omega_2) \cup \Omega_2 =$ $\Sigma - \Gamma = (\Sigma - \Gamma - \Omega_1) \cup \Omega_1$.

Combining conditions (1)-(2) and Definition 7, I is also an occurrence of P on S given the tri-partition $T_1 = (\Gamma, \Sigma - \Gamma - \Omega_1, \Omega_1)$. Hence $sup(P, S, \Gamma, \Sigma - \Gamma - \Omega_1, \Omega_1) \geq sup(P, S, \Gamma, \Sigma - \Gamma - \Omega_2, \Omega_2)$. Because $ofs(P, S)$ is independent from the alphabet partition, we have $fr(P, S, \Gamma, \Sigma - \Gamma - \Omega_1, \Omega_1) \geq fr(P, S, \Gamma, \Sigma - \Gamma - \Omega_2, \Omega_2)$. Consequently $P \in \mathcal{P}(\Sigma, \Gamma, \Sigma - \Gamma - \Omega_1, \Omega_1, S, \rho, N, M)$. Therefore, Equation (6) holds.

Property 2 shows that if we set more medium characters as weak characters, the number of patterns decreases. As an extreme case, we have the following corollary

Corollary 2.

$$\begin{aligned}&\mathcal{P}(\Sigma, \Gamma, \Sigma - \Gamma, \emptyset, S, \rho, N, M)\\ &\supseteq \mathcal{P}(\Sigma, \Gamma, \Sigma - \Gamma - \Omega, \Omega, S, \rho, N, M).\end{aligned} \tag{7}$$

According to Proposition 4, Type IV pattern is a special case of Type I patterns where $\Omega = \emptyset$. Corollary 2 shows that there are more Type IV patterns than Type I ones when Γ is fixed.

Both Properties 1 and 2 show that setting more characters to Λ results in the increase of the number of patterns. Combining Corollaries 1 and 2, we obtain the following corollary.

Corollary 3.

$$\begin{aligned}&\mathcal{P}(\Sigma, \emptyset, \Sigma, \emptyset, S, \rho, N, M)\\ &\supseteq \mathcal{P}(\Sigma, \Gamma, \Lambda, \Omega, S, \rho, N, M).\end{aligned} \tag{8}$$

According to Proposition 3, Type III pattern is a special case of Type I pattern where $\Gamma = \Omega = \emptyset$. Corollary 2 shows that there are more Type III patterns than Type I ones.

4. Algorithm design

In this section, we present the frequent **T**ri-**P**attern **M**ining (TPM) algorithm. Table 1 shows the meaning of all notations to improve the readability.

4.1. TPM main function

Algorithm 1 presents the main function of the TPM algorithm. The parameter list contains a sequence $S = s_1 s_2 \dots s_k$, tri-wildcard gap $G = (N, M)$, tri-partition strategy $T = (\Gamma, \Lambda, \Omega)$ and support threshold ρ. Because the TPM algorithm belongs to Apriori, it consists of two phases: frequent length one tri-patterns mining and frequent longer tri-patterns mining. The first phase is listed by Line 2 and implemented by Algorithm 2. The second phase is listed by Lines 3-14, where "getCandidates" is implemented in Algorithm 3 and "computeSupport" is implemented in Algorithm 4.

4.2. Pattern growth

Algorithm 3 generates $\mathcal{C}_i$ from $\mathcal{P}_{i-1}$, and $\mathcal{C}_i$ is a superset of $\mathcal{P}_i$. Pattern growth entails concatenating two patterns. Let $P = p_1 p_2 \dots p_m$ and $P' = p'_1 p'_2 \dots p'_{m'}$ be two tri-patterns with tri-wildcard gaps $G = (N, M)$. The concatenation is

$$P \oplus P' = p_1 p_2 \dots p_m p'_2 \dots p'_{m'}. \tag{9}$$

In Line 3, we always concatenate pattern P with another pattern $P' \in \mathcal{P}_1$. Lines 4-6 correspond to the pruning technique. Here, $sub(C, 2, |C|)$ is the subpattern of C without the last characters. C is infrequent if $sub(C, 2, |C|)$ is

Algorithm 1 TPM main function

Input: $S = s_1 s_2 \dots s_k$; (N, M); $T = (\Gamma, \Lambda, \Omega)$ and ρ.
Output: Tri-pattern set $\mathcal{P}$.
Constraint: $\forall P \in \mathcal{P}$, $fr(P, S) \geq \rho$.

1: $\mathcal{P} = \emptyset$;
2: $\mathcal{P} = \mathcal{P}_1 =$ frequent1TPMining(S, ρ);
3: **for** ($i = 2$; $\mathcal{P}_{i-1} \neq \emptyset$; i ++) **do**
4: $\mathcal{P}_i = \emptyset$;
5: $\mathcal{C}_i =$ getCandidates($\mathcal{P}_{i-1}$);
6: **for** (each $P \in \mathcal{C}_i$) **do**
7: $sup(P, S) = \sum_{j=1}^{k-|P|+1}$ computeSupport($sub(S, j, k), P$);
8: $fr(P, S) = \frac{sup(P,S)}{k(M-N+1)^{|P|-1}}$;
9: **if** ($fr(P, S) \geq \rho$) **then**
10: $\mathcal{P}_i = \mathcal{P}_i \cup \{P\}$;
11: **end if**
12: **end for**
13: $\mathcal{P} = \mathcal{P} \cup \mathcal{P}_i$;
14: **end for**
15: return $\mathcal{P}$;

Algorithm 2 Frequent length-1 tri-patterns mining

Input: S and ρ.
Output: $\mathcal{P}_1$.
Constraint: $\forall P \in \mathcal{P}_i$, $fr(P, S) \geq \rho$.
Method: frequent1TPMining.

1: $\mathcal{P}_1 = \emptyset$;
2: **for** (each $a \in \Gamma \cup \Lambda$) **do**
3: $P = a$;
4: **if** ($fr(P, S) \geq \rho$) **then**
5: $\mathcal{P}_1 = \mathcal{P}_1 \cup \{P\}$;
6: **end if**
7: **end for**
8: return $\mathcal{P}_1$;

infrequent. This technique facilitates the removal of some candidate patterns. This technique can reduce the runtime significantly, however, the algorithm still can obtain the correct output even without it.

4.3. Computing the pattern occurrences

Algorithm 3 Obtain candidate tri-patterns with length i

Input: $\mathcal{P}_{i-1}$ and $\mathcal{P}_1$.
Output: Candidate tri-patterns set $\mathcal{C}_i$.
Method: getCandidates.

```
1: for (each P ∈ P_{i-1}) do
2:    for (each P' ∈ P_1) do
3:       C = P ⊕ P';
4:       if (sub(C, 2, |C|) ∈ P_{i-1}) then
5:          C_i = C_i ∪ {C};
6:       end if
7:    end for
8: end for
9: return C_i;
```

Algorithm 4 Compute the support of a tri-pattern

Input: $S = s_1 s_2 \dots s_k$ and $P = p_1 p_2 \dots p_m$.
Output: occurrence count.
Method: computeSupport.

```
1: if (s_1 ≠ p_1) then
2:    return 0;//The first position does not match
3: end if
4: if (|P| equals 1) then
5:    return 1;//One occurrence
6: end if
   //Try to match the remaining part of P
7: for (i = N + 1; i ≤ M + 1; i ++) do
8:    if (i + 1 ≥ k) then
9:       break; //Exceeds the bound
10:   end if
11:   if (s_i ∉ Λ ∪ Ω) then
12:      break; //Cannot be viewed as a tri-wildcard
13:   end if
14:   count += computeSupport(sub(S, i + 1, k), sub(P, 2, m));
15: end for
16: return count;
```

Algorithm 4 computes the number of occurrences of a pattern starting at the first position. Note that the support of a pattern in the sequence is computed in Algorithm 1, where all positions are considered.

For Lines 1-3, the result is returned as 0 if the first element of tri-pattern does not match the current element of S. Lines 4-6 indicate that there is exactly one match if the length of P is one. Lines 7-15 attempt to skip some characters

matching tri-wildcards. Lines 8-10 ensure that the bound of the sequence is not exceeded. Lines 11-13 verify whether the character is weak or medium, or not. If the character is not a tri-wildcard and cannot be ignored, the matching progress will be stopped and failed. Line 14 is the core code and invokes the function recursively with a shorter subsequence and shorter subpattern.

We now analyze the time complexity of Algorithm 4. In the worst case, all possible position indices are checked. Therefore, let $W = M - N + 1$, the time complexity is

$$O(W^m), \tag{10}$$

which is exponential with respect to the length of P. According to Line 6 of Algorithm 1, the time complexity of computing $sup(P, S)$ is

$$O(kW^m). \tag{11}$$

4.4. A running example

Let S = CADEFCFDFDEFCADFCDDF, $\Sigma = \{$A, B, C, D, E, F$\}$, $\Gamma = \{$F$\}$, $\Lambda = \{$C, D, E$\}$, $\Omega = \{$A, B$\}$, gap $G = (0, 1)$, namely $N = 0$, $M = 1$ and $\rho = 0.05$. After the execution of Line 1, we have $\mathcal{P}_1 = \{$C, D, E, F$\}$. Next we have $\mathcal{C}_2 = \{$CC, CD, CE, CF, DC, DD, DE, DF, EC, ED, EE, EF, FC, FD, FE, FF$\}$ after the execution of Line 4, which is all of the possible combination that can be formed from $\mathcal{P}_1$. Then the support of each patterns in $\mathcal{C}_2$ is calculated.

Let P = CD, $I_1 = (1, 3)$ is an occurrence of P in S while $I_2 = (6, 8)$ is not. Because $s_7 \in \Gamma$, which means wildcard can not match this kind of characters. $I_3 = (13, 15)$, $I_4 = (17, 18)$ and $I_5 = (17, 19)$ are three occurrences of p in s. Then we have $fr(p, s) = \frac{4}{20 \times 2^1} = 0.1$. According to Line 7, $fr(P, 3) \geq \rho$, and CD will be included in $\mathcal{P}_2$. After the execution of Lines 5 through 10, we have $\mathcal{P}_2 = \{$CD, DF, FC, FD, FF$\}$. Similarly we have $\mathcal{P}_3$ = CDF and $\mathcal{P}_4 = \emptyset$. Finally we have the set of tri-patterns:
$\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3 = \{$C, D, E, F, CD, DF, FC, FD, FF, CDF$\}$.

Let $\Gamma = \{$F, C, D, E, A, B$\}$, $\Lambda = \emptyset$, $\Omega = \emptyset$, $N = 0$ and $M = 0$. We have the set of Type II patterns:
$\mathcal{P} = \{$A, C, D, F, AD, CA, DF, FC, FD, CAD$\}$.

Let $\Gamma = \emptyset$, $\Lambda = \{$F, C, D, E, A, B$\}$, $\Omega = \emptyset$, $N = 0$ and $M = 1$. We have the set of Type III patterns:
$\mathcal{P} = \{$A, B, C, D, E, F, AD, CA, CD, DD, DF, FC, FD, FF, CDF, FCD$\}$.

Let $\Gamma = \{$F$\}$, $\Lambda = \{$C, D, E, A, B$\}$, $\Omega = \emptyset$, $N = 0$ and $M = 1$. We have the set of Type IV patterns:
$\mathcal{P} = \{$A, B, C, D, E, F, AD, CA, CD, DF, FC, FD, FF, CDF$\}$.

Let $\Gamma = \{$F$\}$, $\Lambda = \emptyset$, $\Omega = \{$C, D, E, A, B$\}$, $N = 0$ and $M = 1$. We have the set of Type V patterns:
$\mathcal{P} = \{$F, FF$\}$.

5. Experiments

We undertake experiments on three types of data, namely protein sequence, numerical time series, and Chinese text. We will show that it is natural to

define tri-patterns in these different domains. The source code and all datasets are available at http://www.fansmale.com/software.html.

5.1. Protein sequence

Protein and gene sequences are important data types for pattern discovery with wildcards (see, e.g., [51, 8]). On the one hand, some patterns appear in the sequence many times, hence they are called frequent patterns. On the other hand, in many cases pattern shift [50, 30] causes little influence to the function of the protein or gene (see, e.g., [12, 39]). Here we do not use gene sequences for our experimentation since they contain only four characters, namely A, G, C, and T. It is hard to select some as the medium or weak characters. However, it is useful to analyze amino acid sequence data.

5.1.1. Data and parameter settings

There are 22 major types of amino acid for human being, although Pyrrolysine is often not considered. Biologists partition them into three categories. It is interesting that our idea coincides with theirs.

1) Essential amino acids. These include Histidine (H), Isoleucine (I), Leucine (L), Lysine (K), Methionine (M), Phenylalanine (F), Threonine (T), Tryptophan (W), and Valine (V).
2) Conditional essential amino acids. These include Arginine (R), Cysteine (C), Glutamine (Q), Glycine (G), Proline (P), Serine (S), Tyrosine (Y), and Asparagine (N).
3) Nonessential amino acid. These include Alanine (A), Aspartic acid (D), Glutamic acid (E), Selenocysteine (U), and Pyrrolysine (O).

Let $\Gamma = \{$H, I, L, K, M, F, T, W, V$\}$, $\Lambda = \{$R, C, Q, G, P, S, Y, N$\}$ and $\Omega = \{$A, D, E, U, O, X$\}$. X is an additional character appears in some sequences.

Table 3 lists a total of 20 protein sequences for our experimentation. They are downloaded from the "http://www.ncbi.nlm.nih.gov". Their lengths range from 316 to 2,577.

The parameters are set as follows. For Type I patterns, $N = 0$, $M = faker?$. The lower bound 0 corresponds to the situation where there is no gap at all. This happens quite frequently. The upper bound is (faker?) since pattern shift is often insignificant.

5.1.2. Evaluation measure

We undertake the following steps to evaluate the effectiveness of our algorithms. Let $\mathcal{S} = \{S_1, \ldots, S_n\}$ be a set of the sequences. In our experimentation $n = 20$. We concatenate all sequences to be a whole, that is, $\mathcal{S}' = S_1 + S_2 + \cdots + S_n$.

For each type of patterns, we mine frequent patterns in $\mathcal{S}'$. The set with length l is denoted by $\mathcal{P}_l$. The popularity of any $P \in \mathcal{P}_l$ is given by

$$pop(P, \mathcal{S}) = \frac{|\{S_i \in \mathcal{S} | sup(P, S_i) \geq 1\}|}{n}. \tag{12}$$

Table 3: Protein sequences

Serial No.	Name	Length
AAD21789.2	X1	786
3E9K_A	Kynureninase-3-Hydroxyhippuric	465
AAX88925.1	X2	785
ABA02873.1	cotransporter KCC3a-S3	1,141
AAY24118.1	X3	847
NP_002878.2	arginine–tRNA ligase	660
NP_001161832.1	X-linked isoform e precursor	1,339
NP_116755.1	Y-linked isoform c precursor	1,340
NP_116751.1	X-linked isoform d precursor	1,337
XP_016870347.1	focadhesin isoform X4	1,766
AAY24208.1	X4	2,577
1XCR_A	Longer Splice Variant Of Ptd012	316
NP_001161833	X-linked isoform f precursor	1,065
AAS00361.1	X5	788
AAZ23558.1	cytochrome P450	503
ACM69034.1	Toll-like receptor 5	858
XP_005269704.1	protein RRP5 homolog isoform X1	1,872
AAX88936.1	X6	1,170
NP_001265868.1	elastin isoform m precursor	786
AAF88103.1	zinc finger protein 226	803

5.1.3. Results

Table 4 lists top-10 patterns with length 3 for each type. Here we observe that

1) Type II patterns ($N = M = 0$) are not quite meaningful. Many of them contains only nonessential amino acid, e.g., AAA, EEE, and EEA. Some of them contain conditional essential amino acids, e.g. KKK.
2) Type IV ($\Omega = \emptyset$) patterns are the most similar to Type I. 8 out of 10 frequent Type I patterns are also frequent Type IV pattern. This phenomenon copes with their relationship as indicated in Figure 2.

Note that such explanation is rather simple. We should consult biologists for the usefulness of these patterns.

Table 5 lists the popularity of all types of top-10 patterns. Since overwhelming patterns are undesirable to human experts, we deliberately calibrate ρ such that 40 to 60 patterns are presented. For Type II patterns, the frequency threshold is set to 4ρ since there offset is 1 while others are 4. We observe that Type II have the lowest popularity. This is due to the fact that it is the most special type. Type III pattern has the biggest average popularity, and Type I has the biggest minimal and maximal popularity.

Table 4: Top-10 length-3 frequent protein patterns

Type I	Type II	Type III	Type IV	Type V
DEE	SSS	KAL	DEE	LLV
LEE	AAA	ALV	LGL	KIL
LLE	KKK	LLE	LST	VLL
LAA	EEE	LSI	LEE	LLL
LLA	EEA	LAG	LLE	LVV
KIL	REK	LAA	LAA	LLK
VLL	VPG	LLL	LLA	KLL
DLL	PGV	PLV	KIL	KLK
LLV	EKP	SSL	VLL	TLL
LDE	GVG	LLA	DLL	LIL

Table 5: Frequent protein patterns popularity ($l = 3$, $\rho = 3.4e-4$)

Type	ρ	$\lvert\mathcal{P}_3\rvert$	Popularity		
			Average	Minimal	Maximal
I	$3.4e-4$	54	0.6287	**0.4000**	**0.8999**
II	$8.5e-4$	54	0.3861	0.1000	0.7500
III	$6.7e-4$	55	**0.7299**	0.3499	**0.8999**
IV	$4.4e-4$	53	0.5990	0.2000	**0.8999**
V	$2.2e-4$	50	0.5969	0.3499	0.8000

5.2. Time series

In this section, we employ an oil well production time series for our experimentation. A code table is presented for converting time series into sequences. The tri-partition of the alphabet is also presented. We obtain patterns of different types. These patterns are evaluated by human rather than a predefined measure.

5.2.1. Data and parameter settings

Figure 3 illustrates the daily production of an oil well for two years, hence there are 730 data points.

Given a time series $T = \{t_i | i = 1, \ldots, k+1\}$, the fluctuation from time i to $i+1$ is given by

$$f_i = \frac{t_{i+1} - t_i}{t_i}, \tag{13}$$

where $1 \leq i \leq k+1$.

To model the fluctuation of oil production, we propose the code table as shown in Table 6. Here, the alphabet is Σ = {A, B, C, D, E, F, O, a, b, c, d, e, f}. Using f_i and the code table, we can convert T into sequence $S = s_1 s_2 \ldots s_k$. A part of the sequence of the time series in Figure 3 is AcCCAcECeffBeeEedEEEeeDEoBAcdfD.

Based on the Σ, we empirically let Γ = {D, d, E, e, F, f}, Λ = {B, C, b, c} and Ω = {o, a, A}. This is because bigger fluctuation is considered more

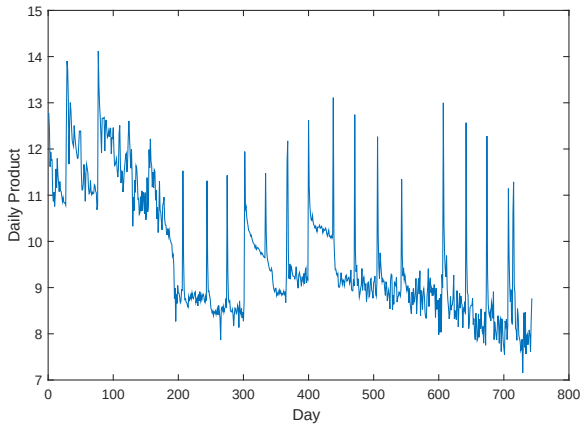


Figure 3: The oil production time series.

Table 6: The code table

f_i	Code	f_i	Code
$(-1\%, 1\%)$	O		
$[1\%, 2\%)$	A	$(-2\%, -1\%]$	a
$[2\%, 3\%)$	B	$(-3\%, -2\%]$	b
$[3\%, 4\%)$	C	$(-4\%, -3\%]$	c
$[4\%, 6\%)$	D	$(-6\%, -4\%]$	d
$[6\%, 10\%)$	E	$(-10\%, -6\%]$	e
$[10\%, +\infty)$	F	$(-\infty, -10\%]$	f

important. In order to discuss the characteristic of Type I to Type V patterns, we first obtain frequent patterns for each type of pattern. Second, for Type I to Type V patterns, we choose a pattern with support not less than 3. Finally, we present the figure of the position matched by the pattern.

5.2.2. Results

Figure 4 depicts the time series matched by the patterns. Here we observe that

1) A Type I pattern matches some similar subsequences with minor differences.
2) A Type II pattern only matches very similar subsequences.
3) A Type III pattern may match some very dissimilar subsequences.
4) A Type IV pattern may also match some very dissimilar subsequences.
5) A Type V pattern may also ignore some important change, however it is better than Types III and IV.

5.3. Faked Chinese text

Frequent patterns in the text are often viewed as keywords. Keywords extraction is an interesting direction with many efficient algorithms. The keywords

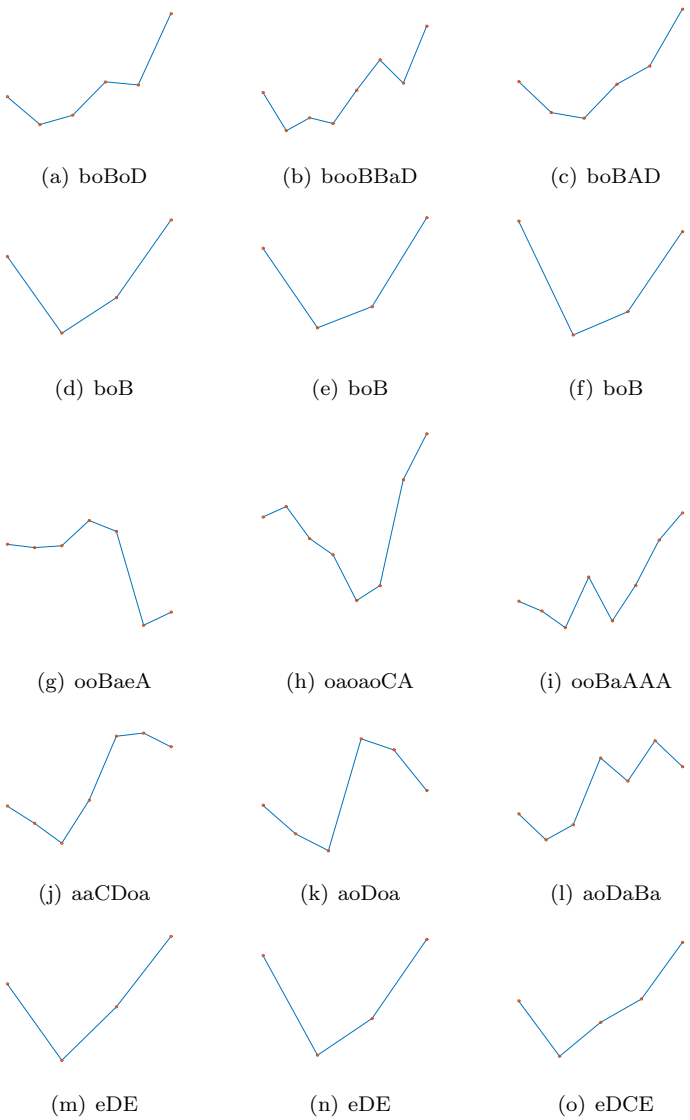


Figure 4: Patterns matched for five types, (a-c) match bBD of Type I, (d-f) match boB of Type II, (g-i) match oaA of Type III, (j-l) match aDa of Type IV, and (m-o) match eDE of Type V.

of texts are often presented in an intuitive way on web pages.

We discuss pattern discovery on faked Chinese text. Some text creators may want to distribute advertisements on illegal stuff such as faked money or invoice. They know that the keywords might be mined by a program and consequently, the text will be removed by email or web servers. Hence they

insert some characters into the text, so that the text is still readable for human. Examples include "代a开**发1票" (faked invoice)and "代@刻*公++章" (faked official seal). Since different characters are inserted in different positions, a plain text mining algorithms will fail to discover these keywords/patterns.

Our question is: Can our algorithm discover the same keywords when the text is deliberately faked? For this purpose, we set $\Lambda = \{$的, 地, 了, $\dots\}$. In some situations they are structural auxiliary words. For example, 的 distinguishes "mine" from "I." In others they express certain meaning. For example, 的 also stands for "target." All other Chinese characters belongs to Γ. Latin characters, numbers, and mathematical operators are included in Ω. That is, $\Omega = \{0, 1, \dots, 9, +, -, *, \dots, a, b, \dots z\}$. We will randomly insert a small proportion of weak characters and an even smaller proportion of medium characters into the text. According to Table 2, all five types of patterns can be mined using our algorithm.

5.3.1. Data and parameter settings

Table 7: Data for text mining

Category	Quantity	Length
News	5	1k
Novel	5	6k-100k
History	5	6k-8k
Law	10	2k-6k

As listed in Table 7, a total of 25 text files are downloaded from the Internet. The filed of content includes news, novel, poem and history. The length ranges from 1k to 100k.

Different proportion of weak characters are inserted into the text. To control the number of parameters, we set the proportion between inserted medium characters and weak characters to be $\frac{1}{10}$.

5.3.2. Evaluation measure

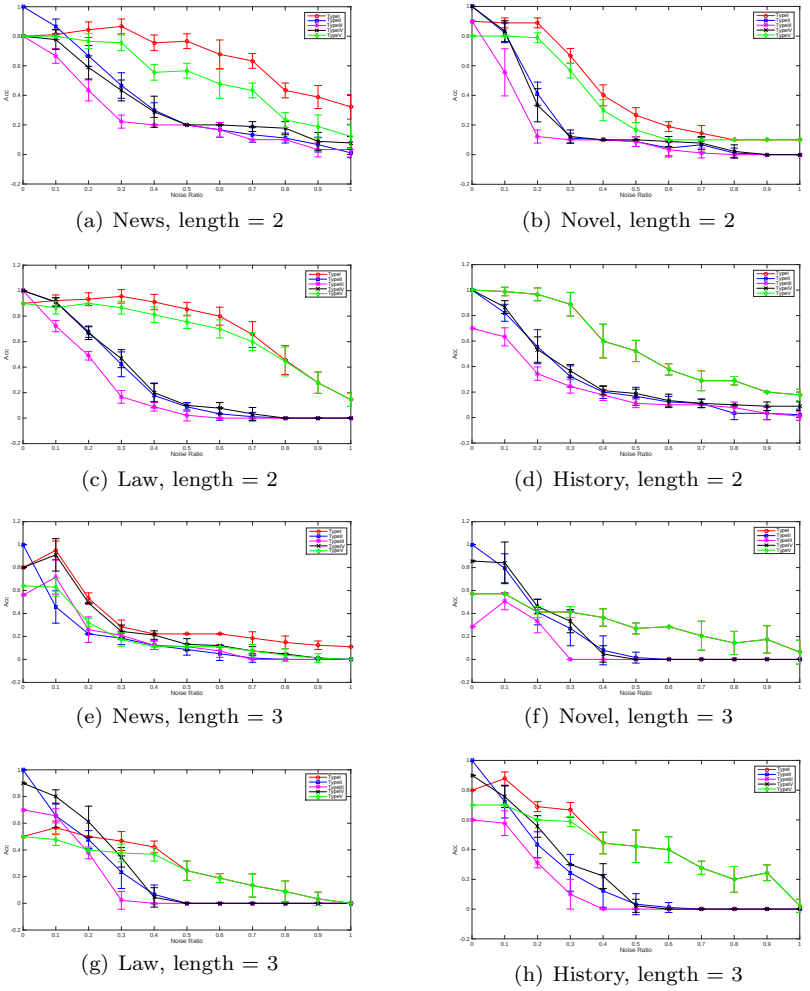
We undertake the following steps to evaluate the effectiveness of our algorithms. First, we obtain the top-k frequent plain patterns with length l from the original text. The set is denoted by $\mathcal{P}_l^o$. This is done through adjusting the parameter ρ through binary search. Second, we find out the top-k frequent weak patterns with length l from the faked text. The set is denoted by $\mathcal{P}_l^f$. The accuracy of the algorithm for length l is defined as

$$acc_l = \frac{|\mathcal{P}_l^o \cap \mathcal{P}_l^f|}{|\mathcal{P}_l^o|}. \tag{14}$$

To show the validity of the algorithm, frequent plain patterns on the faked text is also mined. In this way the accuracy of the plain pattern discovery algorithm is obtained.

Here we consider the pattern length while computing the accuracy. The reason is that a frequency that appropriate for one length would be quite inappropriate for another. To obtain a few length-4 patterns, we may set the threshold ρ to be quite low, and the number of length-2 patterns might be overwhelming. Taking them as a whole will make length-4 patterns quite invisible.

5.3.3. Results



(a) News, length = 2 (b) Novel, length = 2

(c) Law, length = 2 (d) History, length = 2

(e) News, length = 3 (f) Novel, length = 3

(g) Law, length = 3 (h) History, length = 3

Figure 5: Accuracy using all five types of patterns with length 2 or 3.

Figure 5 presents the results of length 2 and 3 patterns, respectively. For each text, we generate 10 fake versions, and obtain the mining accuracy on these versions. Note that the noise ratio can be greater than 1. In this case there are more inserted text than the original, and the text is badly faked.

Figures 5 (a)-(d) illustrate the accuracy of Type I to Type V patterns with length 2 on News, Novel, Law and History Chinese texts. It can be observed that the Type I performs the best, while Type III performs the worst. The accuracy of all types of patterns decreases with the increase of the noise ratio. In Figure 5 (d), both Type I and Type V the same best performance. This is because the most of middle elements in Λ are not frequent. Therefore, the obtained frequent tri-patterns (Type I) do not contain these middle elements. Hence, Type I patterns are the same as Type V patterns.

Figures 5 (e)-(h) illustrate the results with length 3 on these four Chinese texts. On the News text (Figure 5 (e)), Type I is always the best with the noise ratio ranging from 0.1 to 1.0. When there is no noise (noise ratio = 0.0), Type I and Type IV are the second best one following Type II. The accuracies of Type I and Type IV are the same as each other and very closet to 0.8. On the Novel text (Figure 5 (f)), Type I and Type V has the same accuracy as each other and perform the best with the noise ratio ranging from 0.3 to 1.0. When the noise ratio ranging from 0.0 to 0.3, the accuracy of Type I is less than those of Type II and Type IV. On the Law text (Figure 5 (g)), Type I performs the best when the noise ratio ranges from 0.3 to 1.0. Moreover, from 0.5 to 1.0, the accuracy of Type I is the same as Type V. On the History text (Figure 5 (h)), Type I performs the best when the noise ratio ranges from 0.1 to 1.0. When the noise ratio is 0.0, Type I performs worse than Type II and Type IV. However, its accuracy is up to 0.8.

Table 8 lists the accuracy of different types of patterns on all 25 texts when the noise ratio is 30% and the pattern length is 2. We can observe that the accuracy of Type I is always the highest. For all 25 texts, Type I obtains 8 highest accuracies no less than 0.9. The lowest one is 0.5, which is still the best one. Note that, the accuracy of Type V is the same as Type I on some texts (e.g., Law: nsfc.gov.cn). The reason is that most middle elements in Λ are infrequent.

The answer to our question is: when the text is slightly faked (e.g., less than 30% characters are inserted), most original keywords can be still mined with Type I patterns through our algorithms. We also observe that

1) Type I patterns are the most accurate among all five types.
2) Type I patterns are always more accurate than Type V patterns.
3) Type III patterns are most sensitive to noise. There are often plenty of frequent but meaningless patterns. The reason lies in that any character can form the pattern, while they can also be ignored.
4) From Figures 5, when the noise ratio is low, some Type IV patterns are causal, making Type IV patterns less accurate than Type II ones. However, with the increase of the noise ratio, Type IV patterns gradually outperforms Type II ones.

6. Conclusions

In this paper, we defined a frequent tri-pattern discovered from a tri-partition alphabet. Existing sequential patterns (Type II through V) are special cases

Table 8: Results summary of text mining

Type	Source	Accuracy				
		Type I	Type II	Type III	Type IV	Type V
News	news.sohu.com	**1.0000 ± 0.0000**	0.5111 ± 0.1050	0.2000 ± 0.0866	0.5777 ± 0.1092	**1.0000 ± 0.0000**
	finance.ifeng.com	**0.9111 ± 0.0333**	0.5777 ± 0.0833	0.3666 ± 0.1000	0.6333 ± 0.1500	**0.9111 ± 0.0333**
	news.china.com	**0.6555 ± 0.0527**	0.4111 ± 0.0600	0.2222 ± 0.0440	0.4333 ± 0.0499	0.5555 ± 0.0527
	chinanews.com	**0.9666 ± 0.0499**	0.5333 ± 0.1118	0.3000 ± 0.1118	0.5666 ± 0.099	**0.9666 ± 0.0499**
	dvdc100.com	**0.8777 ± 0.0440**	0.4888 ± 0.1054	0.2222 ± 0.0440	0.3888 ± 0.1054	0.7222 ± 0.0666
Novel	jjwxc.net	**0.7000 ± 0.0866**	0.1777 ± 0.0666	0.0999 ± 0.0001	0.1222 ± 0.0440	0.6000 ± 0.0866
	xs8.cn	**0.7000 ± 0.0001**	0.4333 ± 0.0001	0.3777 ± 0.0440	0.4111 ± 0.0333	**0.7000 ± 0.0001**
	xs8.cn	**0.8777 ± 0.0440**	0.3333 ± 0.0707	0.0666 ± 0.050	0.3333 ± 0.0500	0.7888 ± 0.0333
	xs8.cn	**0.6333 ± 0.0499**	0.2555 ± 0.0726	0.099 ± 0.0001	0.2777 ± 0.0440	**0.6333 ± 0.0499**
	news.ifeng.com	**0.5777 ± 0.0440**	0.2222 ± 0.0440	0.1999 ± 0.0001	0.1999 ± 0.0001	**0.5777 ± 0.0440**
Law	law-lib.com	**0.9000 ± 0.0001**	0.3333 ± 0.0500	0.1888 ± 0.0781	0.3555 ± 0.0527	0.7000 ± 0.0001
	nsfc.gov.cn	**0.9888 ± 0.0333**	0.7888 ± 0.0927	0.4777 ± 0.0666	0.8777 ± 0.0440	**0.9888 ± 0.0333**
	gov.cn	**0.9444 ± 0.0527**	0.4111 ± 0.0600	0.1555 ± 0.0527	0.4555 ± 0.0527	0.8555 ± 0.0527
	jincao.com	**0.8888 ± 0.0333**	0.3111 ± 0.0333	0.1666 ± 0.0500	0.3111 ± 0.0333	0.7999 ± 0.0001
	bj.xinhuanet.com	**0.9888 ± 0.0333**	0.5111 ± 0.0333	0.3888 ± 0.0600	0.5666 ± 0.0499	0.9000 ± 0.0010
History	news.ifeng.com	**0.5000 ± 0.0001**	0.2333 ± 0.0499	0.1888 ± 0.0333	0.2111 ± 0.0333	**0.5000 ± 0.0001**
	news.ifeng.com	**0.9222 ± 0.0666**	0.3888 ± 0.0781	0.2222 ± 0.0440	0.3999 ± 0.0500	**0.9222 ± 0.0666**
	71.cn	**0.7000 ± 0.0001**	0.2444 ± 0.1013	0.0999 ± 0.0001	0.2111 ± 0.0781	**0.7000 ± 0.0001**
	news.ifeng.com	**0.7000 ± 0.0001**	0.3333 ± 0.0500	0.1666 ± 0.0500	0.3000 ± 0.0500	**0.7000 ± 0.0001**
	news.ifeng.com	**1.0000 ± 0.0000**	0.3888 ± 0.0600	0.2444 ± 0.0527	0.3888 ± 0.0333	**1.0000 ± 0.0000**

of our tri-pattern. Hence the proposed TPM algorithm can also discover all other types by only changing the tri-partition strategy and/or the wildcard gap. Experiments on protein sequence, time series and Chinese texts show that:

1) Type I patterns provide a better tradeoff between flexibility and accuracy than the other four.
2) The proposed algorithm can obtain frequent tri-patterns effectively and reliably.

The following research topics deserve further investigation:

1) Pattern explanation and evaluation involving domain-specific experts. For some real applications such as oil-well production pattern discovery, the meaning of the pattern can be only evaluated by experts.
2) Application of the discovered patterns. These patterns can be used to not only reveal the situation or status of the sequence, but also cluster sequences or prediction.
3) Tri-pattern discovery from multiple time series, which is more complex, and more widely exists in reality.

We hope that this work opens a new door for the development of three-way decision.

Acknowledgements

This work is in part supported by National Science Foundation of China [grant numbers 61379089, 41604114]; and the Open Research Fund of Sichuan Key Laboratory for Nature Gas and Geology [grant number 2015trqdz04].

[1] Amino acid, https://en.wikipedia.org/wiki/Amino_acid (2017).

[2] R. Agrawal, R. Srikant, Mining sequential patterns, in: ICDE, 1995.

[3] M. Andradeab, P. Borkab, Automated extraction of information in molecular biology, Febs Letters 476 (1–2) (2000) 12–17.

[4] G. Chen, X.-D. Wu, X.-Q. Zhu, A. N. Arslan, Y. He, Efficient string matching with wildcards and length constraints, Knowledge & Information Systems 10 (4) (2006) 399–419.

[5] Chiu, Bill, Keogh, Eamonn, Lonardi, Stefano, Probabilistic discovery of time series motifs, in: Proc. Acm Sigkdd Int.conf.on Knowledge Discovery & Data Mining, 2003.

[6] X.-F. Deng, Y. Y. Yao, Decision-theoretic three-way approximations of fuzzy sets, Information Sciences 279 (2014) 702–715.

[7] T. G. Dietterich, R. S. Michalski, Discovering patterns in sequences of events, Artificial Intelligence 25 (2) (1985) 187–232.

[8] P. G. Ferreira, P. J. Azevedo, Protein sequence pattern mining with constraints, in: European Conference on Principles of Data Mining and Knowledge Discovery, 2005.

[9] C. Gao, Y. Y. Yao, Actionable strategies in three-way decisions, Knowledge-Based Systems (2017) 1–15.

[10] V. Gligorijević, N. Pržulj, Methods for biological data integration: perspectives and challenges, Journal of the Royal Society Interface 12 (112).

[11] S. Greco, R. Slowiński, Y. Y. Yao, Bayesian decision theory for dominance-based rough set approach 4481 (2007) 134–141.

[12] D. Guo, X.-G. Hu, F. Xie, X.-D. Wu, Pattern matching with wildcards and gap-length constraints based on a centrality-degree graph, Applied intelligence 39 (1) (2013) 57–74.

[13] B.-Q. Hu, Three-way decisions space and three-way decisions, Information Sciences 281 (281) (2014) 21–52.

[14] B. Huang, C.-X. Guo, Y.-X. Zhuang, H.-X. Li, X.-Z. Zhou, Intuitionistic fuzzy miltigranulation rough sets, Information Sciences 277 (2014) 299–320.

[15] C.-K. Huang, Mining the change of customer behavior in fuzzy time-interval sequential patterns, Applied Soft Computing 12 (3) (2012) 1068–1086.

[16] A. Khotanzad, Y. H. Hong, Invariant image recognition by zernike moments, IEEE Transactions on Pattern Analysis & Machine Intelligence 12 (5) (1990) 489–497.

[17] H. Kim, Y. Kim, J. Y. Sim, C. S. Kim, Spatiotemporal saliency detection for video sequences based on random walk with restart, IEEE Transactions on Image Processing 24 (8) (2015) 2552–2564.

[18] D. Li, W. Yan, W.-Y. Li, Z.-Y. Ren, A two-tier wind power time series model considering day-to-day weather transition and intraday wind power fluctuations, IEEE Transactions on Power Systems 31 (6) (2016) 4330–4339.

[19] H.-X. Li, L.-B. Zhang, B. Huang, X.-Z. Zhou, Sequential three-way decision and granulation for cost-sensitive face recognition, Knowledge-Based Systems 91 (C) (2016) 241–251.

[20] H.-X. Li, X.-Z. Zhou, B. Huang, D. Liu, Cost-sensitive three-way decision: A sequential strategy, in: International Conference on Rough Sets and Knowledge Technology, 2013.

[21] J.-H. Li, C.-C. Huang, J.-J. Qi, Y.-H. Qian, W.-Q. Liu, Three-way cognitive concept learning via multi-granularity, Information Sciences 378 (2017) 244–263.

[22] X.-N. Li, H.-J. Yi, Y.-H. She, B.-Z. Sun, Generalized three-way decision models based on subset evaluation, International Journal of Approximate Reasoning 83 (C) (2017) 142–159.

[23] J. Lin, E. Keogh, S. Lonardi, P. Patel, Finding motifs in time series, in: 2nd Workshop on Temporal Data Mining, 2002.

[24] D. Liu, D. Liang, C.-C. Wang, A novel three-way decision model based on incomplete information system, Knowledge-Based Systems 91 (2016) 32–45.

[25] S. C. B. Lo, H. P. Chan, J. S. Lin, H. Li, M. T. Freedman, S. K. Mun, Artificial convolution neural network for medical image pattern recognition, Neural Networks 8 (7–8) (1995) 1201–1214.

[26] I. Lopes, H. G. Silva, Looking for granulation and periodicity imprints in the sunspot time series, Astrophysical Journal 804 (2).

[27] D.-Q. Miao, G.-Y. Wang, Q. Liu, T.-Y. Lin, Y. Y. Yao, Granular computing: past, present and future prospects, Science Press, Beijing, 2007.

[28] R. Mihalcea, P. Tarau, Textrank: Bringing order into texts, Unt Scholarly Works (2004) 404–411.

[29] F. Min, F.-L. Liu, L.-Y. Wen, Z.-H. Zhang, Tri-partition cost-sensitive active learning through knn, Soft Computing (10) (2017) 1–16.

[30] F. Min, Y.-X. Wu, X.-D. Wu, The apriori property of sequence pattern mining with wildcard gaps, International Journal of Functional Informatics and Personalised Medicine 4 (1) (2012) 15–31.

[31] C. H. Mooney, J. F. Roddick, Sequential pattern mining–approaches and algorithms, ACM Computing Surveys (CSUR) 45 (2) (2013) 1–39.

[32] A. Mueen, E. J. Keogh, Q. Zhu, S. Cash, M. B. Westover, Exact discovery of time series motifs, in: SDM, 2009.

[33] S. B. Needleman, C. D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, Journal of molecular biology 48 (3) (1970) 443–453.

[34] A. Ng, A. W.-C. Fu, Mining frequent episodes for relating financial events and stock trends, in: Advances in Knowledge Discovery and Data Mining, Springer, 2003, pp. 27–39.

[35] O. Olominu, A. A. Sulaimon, et al., Application of time series analysis to predict reservoir production performance, in: SPE Nigeria Annual International Conference and Exhibition, Society of Petroleum Engineers, 2014.

[36] Z. Pawlak, Rough sets, International Journal of Computer & Information Sciences 11 (5) (1982) 341–356.

[37] S. Pletscherfrankild, A. Pallejá, K. Tsafou, J. X. Binder, L. J. Jensen, Diseases: Text mining and data integration of disease-gene associations, Methods 74 (2015) 83–89.

[38] B.-B. Sang, Y.-T. Guo, D.-R. Shi, W.-H. Xu, Decision-theoretic rough set model of multi-source decision systems, International Journal of Machine Learning & Cybernetics (1) (2017) 1–14.

[39] R. She, F. Chen, K. Wang, M. Ester, J. L. Gardy, F. S. L. Brinkman, Frequent-subsequence-based prediction of outer membrane proteins, in: ACM SIGKDD, 2003.

[40] Y.-H. She, X.-L. He, H.-X. Shi, Y.-H. Qian, A multiple-valued logic approach for multigranulation rough set model, International Journal of Approximate Reasoning 82 (2017) 270–284.

[41] W.-B. Shi, P.-J. Shang, The multiscale analysis between stock market time series 26.

[42] A. Suppes, F. Suhling, M. Hötter, Robust obstacle detection from stereoscopic image sequences using kalman filtering, Pattern Recognition 2191 (2001) 385–391.

[43] C.-D. Tan, F. Min, M. Wang, H.-R. Zhang, Z.-H. Zhang, Discovering patterns with weak-wildcard gaps, IEEE Access 4 (2016) 4922–4932.

[44] J.-S. Tang, R. Rangayyan, J.-H. Yao, Y.-Y. Yang, Digital image processing and pattern recognition techniques for the detection of cancer, Pattern Recognition 42 (6) (2009) 1015–1016.

[45] Y. R. Tausczik, J. W. Pennebaker, The psychological meaning of words: Liwc and computerized text analysis methods, Journal of Language and Social Psychology 29 (1) (2010) 24–54.

[46] M. A. P. Taylor, Understanding traffic systems: Data analysis and presentation, 2nd edition (hardback) - routledge, Transportation Research Part A-Policy and Practice 32 (8) (2000) 644–645.

[47] M. Trottini, M. Isabel, V. Aguiar, S. B. Palazón, On the use of running trends as summary statistics for univariate time series and time series association, Journal of Climate 28 (19) (2015) 7489–7502.

[48] M. Wang, F. Min, Z.-H. Zhang, Y.-X. Wu, Active learning through density clustering, Expert Systems with Applications 85 (2017) 305–317.

[49] M. M. Wiest, K. J. Lee, J. B. Carlin, Statistics for clinicians: An introduction to logistic regression, Journal of Paediatrics and Child Health 51 (7) (2015) 670–673.

[50] X.-D. Wu, J.-P. Qiang, F. Xie, Pattern matching with flexible wildcards, Journal of Computer Science and Technology 29 (5) (2014) 740–750.

[51] Y.-X. Wu, S. Fu, H. Jiang, X.-D. Wu, Strict approximate pattern matching with general gaps, Applied Intelligence 42 (3) (2015) 566–580.

[52] W.-H. Xu, Y.-T. Guo, Generalized multigranulation double-quantitative decision-theoretic rough set, Knowledge-Based Systems 105 (2016) 190–205.

[53] J. Yang, W. Wang, P. S. Yu, J.-W. Han, Mining long sequential patterns in a noisy environment, in: 2002 ACM SIGMOD, 2002.

[54] J.-T. Yao, A. V. Vasilakos, W. Pedrycz, Granular computing: perspectives and challenges, IEEE Transactions on Cybernetics 43 (6) (2013) 1977–1989.

[55] Y. Y. Yao, Three-way decision: An interpretation of rules in rough set theory, in: International Conference on Rough Sets and Knowledge Technology, 2009.

[56] Y. Y. Yao, An outline of a theory of three-way decisions, in: International Conference on Rough Sets and Current Trends in Computing, 2012.

[57] Y. Y. Yao, Three-way decisions and cognitive computing, Cognitive Computation 8 (4) (2016) 543–554.

[58] Y. Y. Yao, S. K. Wong, A decision theoretic framework for approximating concepts, International Journal of Man-Machine Studies 37 (6) (1992) 793–809.

[59] H. Yu, Y. Wang, Three-way decisions method for overlapping clustering, in: International Conference on Rough Sets and Current Trends in Computing, 2012.

[60] M. J. Zaki, Spade: An efficient algorithm for mining frequent sequences, Machine Learning 42 (1–2) (2001) 31–60.

[61] H.-R. Zhang, F. Min, Three-way recommender systems based on random forests, Knowledge-Based Systems 91 (C) (2016) 275–286.

[62] H.-R. Zhang, F. Min, B. Shi, Regression-based three-way recommendation, Information Sciences 378 (2017) 444–461.

[63] B. Zhou, Y. Y. Yao, J.-G. Luo, Cost-sensitive three-way email spam filtering, Journal of Intelligent Information Systems 42 (1) (2014) 19–45.

[64] X.-Q. Zhu, W. G. Aref, J.-P. Fan, A. C. Catlin, Medical video mining for efficient database indexing, management and access, in: International Conference on Data Engineering, Proceedings, 2003.

[65] W. Ziarko, Variable precision rough set model, Journal of Computer & System Sciences 46 (1) (1993) 39–59.