IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 27, 2016, accepted July 15, 2016, date of publication August 25, 2016, date of current version September 28, 2016.

Digital Object Identifier 10.1109/ACCESS.2016.2593953

Discovering Patterns With Weak-Wildcard Gaps

CHAO-DONG TAN[1], FAN MIN[2], (Member, IEEE), MIN WANG[2], HENG-RU ZHANG[2], AND ZHI-HENG ZHANG[2]

[1]College of Petroleum Engineering, China University of Petroleum, Beijing 102249, China
[2]School of Computer Science, Southwest Petroleum University, Chengdu 610500, China

Corresponding author: F. Min (minfanphd@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61379089, in part by the Natural Science Foundation of the Department of Education of Sichuan Province under Grant 16ZA0060 and Grant 15ZB0056, and in part by the State Administration of Work Safety Project under Grant sichuan-0008-2016AQ.

ABSTRACT Time series analysis is an important data mining task in areas such as the stock market and petroleum industry. One interesting problem in knowledge discovery is the detection of previously unknown frequent patterns. With the existing types of patterns, some similar subsequences are overlooked or dissimilar ones are matched. In this paper, we define patterns with weak-wildcard gaps to represent subsequences with noise and shift, and design efficient algorithms to obtain frequent and strong patterns. First, we convert a numeric time series into a sequence according to the data fluctuation. Second, we define the pattern mining with weak-wildcard gaps problem, where a weak-wildcard matches any character in an alphabet subset. Third, we design an Apriori-like algorithm with an efficient pruning technique to obtain frequent and strong patterns. Experimental results show that our algorithm is efficient and can discover frequent and strong patterns.

INDEX TERMS Pattern discovery, sequences, time series analysis, weak-wildcard.

I. INTRODUCTION

Frequent pattern discovery from time-series data is important in fields such as the stock market and petroleum industry. Time series data can consist of data such as the daily closing price of stocks [1] or daily well production [2]. People often discover patterns [3] or motifs [4] based on similar subsequences. Specifically, frequent patterns are desired because they represent some events [5] or are important for commerce [6]. According to Lonardi and Patel [4], an efficient pattern discovery algorithm is useful. It can be used as a tool for summarizing and visualizing massive time-series databases. It also can be used as a subroutine in other data mining tasks such as classification, clustering, and association-rule mining. One key issue is how to define a pattern to represent similar subsequences.

A number of different definitions of a pattern have been proposed for numeric and symbolic data [7]. Popular approaches to numeric data are based on distance measures such as Euclidian distance [4]. Two subsequences are supposed to match if their distance is smaller than a user-specified threshold [4]. For instance, the subsequences illustrated in Figs. 1 (a)–(c) match each other within a reasonable threshold. The advantage of these approaches is that they discover the original subsequence. However, they essentially do not provide the generalized patterns of the subsequences. Moreover, two subsequences with different lengths cannot be compared.

Approaches using symbolic data have been more fruitful [8]–[10] in recent years. Some data such as gene sequences [8], [9] and text [11] are intrinsically symbolic. Numeric data can also be encoded as symbolic data [4], [12]. For example, the subsequences illustrated in Figs. 1 (a)–(c) can all be coded as ''BbCBD''. Consequently, the plain pattern P_1 = BbCBD matches all of them. Coding ignores minor differences in the subsequences, and the data are represented at a coarser granularity. In granular computing [13]–[16], the patterns are more visible [13], [17], [18] at an appropriate granularity. To deal with data with noise [19] or shift, the concept of wildcard gaps has been proposed. A wildcard is a special symbol ϕ that matches any character, while a wildcard gap is a sequence of wildcards [8], [20], [21]. The latter has the form $[N, M]$, where N and M stand for the lower and upper length bounds, respectively. Pattern P_2 = B[0,1]b[0,1]C[0,1]B[0,1]D matches all nine subsequences in Figure 1. We observe that subsequences in Figs. 1 (d)–(f) are similar to those in Figs. 1 (a)–(c). Unfortunately, the subsequences in Figs. 1 (h)–(j) are quite dissimilar to the rest.

In this paper, we define a new type of pattern and design an efficient algorithm to obtain frequent ones. First, we code

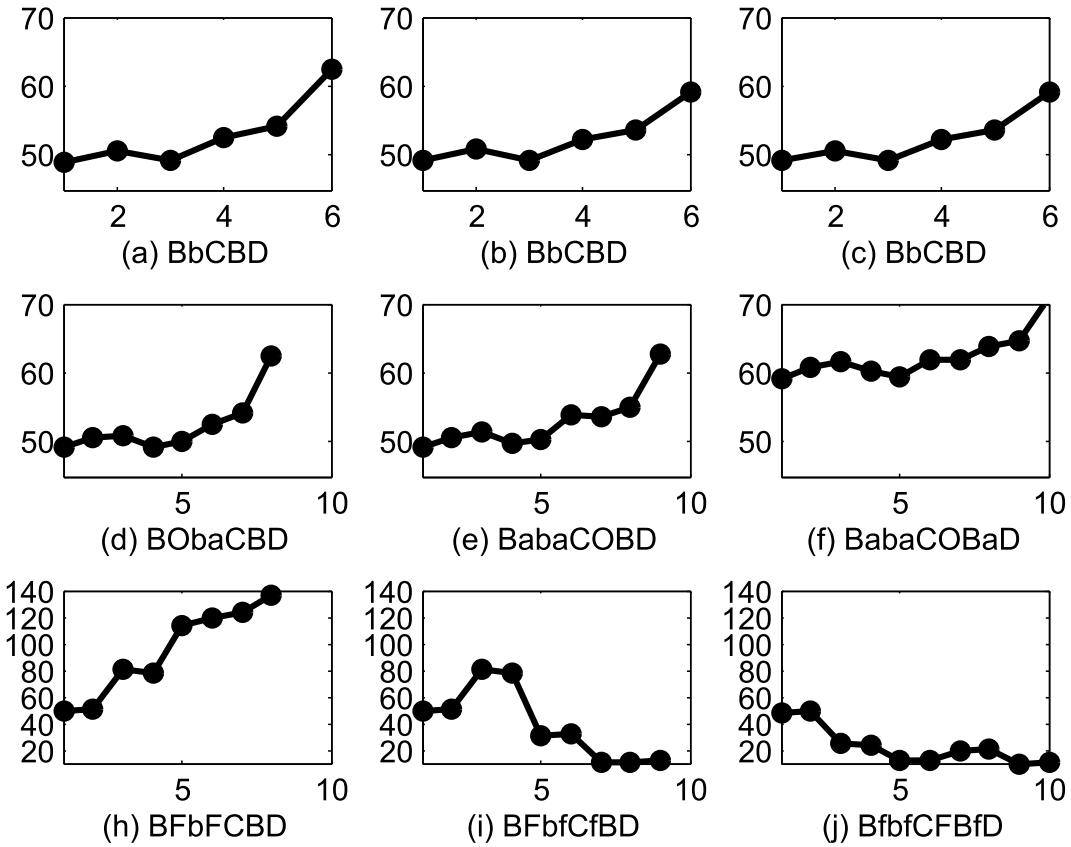

FIGURE 1. Nine subsequences: (a)–(c) are very similar, (a)–(f) are similar, and (h)–(j) are dissimilar.

the time-series to obtain a symbolic sequence according to the fluctuation in the data. This coding technique is similar to existing ones, but has some crucial differences [12]. The conversion is based on the exponential change of the data. In this way, minor to major fluctuations are coded by a small alphabet.

Second, we define patterns with weak-wildcard gaps along with a frequent pattern mining problem. In our data conversion approach, some characters represent minor changes while others represent major ones. Thus, we introduce the concepts of weak and strong characters. A weak-wildcard matches any weak character, while a weak-wildcard gap matches a sequence of weak characters and has the form (N, M). The new type of patterns is more general than two existing ones. In some cases, we require further that only strong characters can be included in a pattern. For example, P_3 = B(0,1)b(0,1)C(0,1)B(0,1)D matches the subsequences in Figs. 1 (a)–(f), but not the other three. This pattern is more appropriate than P_1 and P_2 in our scenario. Hence, this type of pattern is semantically richer than the other two because minor changes can be ignored if necessary.

Third, we propose an Apriori-like algorithm for the new problem and prove that the Apriori property holds for the new type of patterns. We design a pruning technique to accelerate the mining algorithm. To discover only strong patterns, we need a slight change in the code.

We conducted experiments on a number of real-world time series. Results show that 1) our algorithms are efficient, especially with the use of our pruning technique, exponentially reducing the running time of the system; 2) discovered patterns matches subsequences with different lengths in the original time-series; and 3) compared with patterns with wildcard gaps, our pattern excludes dissimilar subsequences.

The remainder of this paper is organized as follows. In Section II, we introduce the numeric data pattern discovery and review two sequence pattern mining problems. We present our data conversion method and define our new sequence pattern mining problem in Section III. In Section IV, we present the pattern mining algorithm and three subroutines. We provide experimental results in Section V. Finally, we present our concluding remarks in Section VI.

II. RELATED WORK

In this section, we focus on three concepts: numerical data pattern discovery, symbolic sequence pattern discovery, and symbolic sequence pattern discovery with wildcard gaps.

A. NUMERICAL DATA PATTERN DISCOVERY

Time series data mining has been widely used in different applications. Examples of numerical time series include stock prices [22], oil well production [23], product sales

volume [24], and temperature [25]. A time series is a sequence of real numbers, where each number represents a value for an attribute of interest at a particular time instant.

Definition 1: [26] A time series T is a sequence of time-ordered data.

$$T = \{t_i | i = 1, \ldots, k+1\}, \tag{1}$$

where t represents time, $k+1$ is the number of observations made during that period, and t_i is the value measured at time instant i.

Given a subsequence P of time series T, a similarity measure such as Euclidian distance, and a similarity threshold D, efficient algorithms have been developed [27]–[29] to locate all subsequences similar to P in T. This problem is often referred to as pattern matching [30] or subsequence searching.

Lonardi and Patel [4] argued that finding previously unknown, frequent patterns (also call motifs) are more interesting. This task is referred to as pattern discovery [11] or pattern mining [31]. Lonardi and Patel [4] essentially employed two techniques to find frequent patterns. One is dimension reduction, which reduces the number of data points.

$$\bar{t}_i = \frac{w}{k} \sum_{j=\frac{k}{w}(i-1)+1}^{\frac{k}{w}i} t_j. \tag{2}$$

Simply stated, to reduce the time series from k dimensions to w dimensions, the data is divided into w equal-sized parts. In this way, the number of data points is decreased. For example, we may be interested in the daily runoff of a river, instead of hourly values.

The other is discretization. First, we obtain the piecewise aggregate approximation (PAA) of a time series using Equation (2). Second, we need to determine the breakpoints by looking them up in a statistical table. Finally, all PAA coefficients that are below the lowest breakpoint are mapped to symbol a. All coefficients higher than or equal to the smallest breakpoint but lower than the second smallest breakpoint are mapped to symbol b, and so on. In fact, this technique is quite common in areas such as communication.

Many scholars have carried out a number of related studies on the pattern discovery problem. Zhong *et al* [11] presented an effective pattern discovery technique to find relevant and interesting information. Tanaka *et al* [32] proposed a motif discovery algorithm to extract a motif based on the Minimum Description Length (MDL) principle. Mueen *et al* [33] presented a tractable exact algorithm for finding time series motifs. Tang and Liao [31] introduced a k-motif-based algorithm that can discover original motifs with different lengths.

Sometimes the data are converted into symbols. However, respective approaches [4], [31] are still based on a similarity measure and distance threshold.

B. SYMBOLIC SEQUENCE PATTERN DISCOVERY

Examples of symbolic time series include ultrasonic data [34], mechanical systems anomaly detection data [35], and chaotic systems parameter estimation [36]. The respective research has been quite fruitful in recent years. In this context, we are often not interested in the distance between subsequences.

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

Definition 2: An alphabet Σ is a non-empty finite set, whose elements are called characters.

A symbolic sequence is a series of ordered characters.

Definition 3: Any $S = s_1 s_2 \ldots s_k$, where $s_i \in \Sigma$ for any $1 \leq i \leq k$ is a symbolic sequence, also called a string.

For example, Σ = {a, b} is an alphabet, S = ababb is a symbolic sequence of the alphabet, and $|S| = 5$.

To analyze a symbolic sequence, people often focus on small parts of the sequence called patterns. Essentially, a pattern $P = p_1 p_2 \ldots p_m$ is also a symbolic sequence. For example, in natural language processing, each word can be viewed as a pattern. To automatically extract keywords from a paper, a common approach is to count the number of occurrences of each word or phrase.

Definition 4: Given sequence $S = s_1 s_2 \ldots s_k$ and pattern $P = p_1 p_2 \ldots p_m$, P matches S at position i iff $\forall 1 \leq j \leq m$, $s_{i+j-1} = p_j$.

The number of matches, also called occurrences, of P in S is denoted as $sup(P, S)$.

Definition 5: The frequency of pattern P in sequence S is

$$f(P, S) = sup(P, S)/k. \tag{3}$$

This indicates the frequency with which P matches S at different positions. Given a frequency threshold ρ, P is frequent iff $f(P, S) \geq \rho$.

The following problem is formulated for finding frequent patterns.

Problem 1: **Frequent plain pattern mining**

Input: sequence S and support threshold ρ.

Output: frequent pattern set $\mathcal{P}$.

Example 1: Let $S = ababb$ and $P = ab$. P matches S at positions 1 and 3, $sup(P, S) = 2$, and $f(P, S) = 2/5 = 0.4$. Let $\rho = 0.3$, $\mathcal{P} = \{a, b, ab\}$.

C. SYMBOLIC SEQUENCE PATTERN DISCOVERY WITH WILDCARD GAPS

Patterns with wildcards have garnered substantial attention in real-world applications because of their flexibility. In business, managers can adjust marketing methods or strategies using frequency patterns with wildcards [37]. In biology, periodic patterns with wildcards are valued for having significant biological and medical utility [38], [39]. Li and Wang [40] found that some frequent periodic patterns can be used in feature selection.

Definition 6 [41]: A wildcard ϕ is a special symbol that matches any character in Σ. A gap $g[N, M]$ is a sequence of wildcards of minimal size N and maximal size M. $W = M - N + 1$ is the gap flexibility of $g[N, M]$.

$g[N, M]$ is also denoted by $[N, M]$ in a pattern.

Definition 7 [41]: A pattern is a sequence of characters and gaps that begins and ends with characters. A periodic pattern has the following form:

$$P = p_1[N, M]p_2[N, M] \ldots, [N, M]p_m, \tag{4}$$

where m is the length of the pattern.

The term periodic indicates that all gaps are identical, rendering the analysis simpler.

Example 2: A[1, 3]C[1, 3]C is a pattern with periodic wildcard gaps [1, 3].

Definition 8 [41]: If there exists a position sequence $1 \leq i_1 \leq i_2 \leq \cdots \leq i_m \leq k$ such that $s_{i_j} = p_j$ for all $1 \leq j \leq m$ and $N \leq i_j - i_{j-1} - 1 \leq M$ for all $1 \leq j \leq m-1$, then $\langle i_1, i_2, \ldots, i_m \rangle$ is called an occurrence of P in S with the $g[N, M]$ gap constraint.

Example 3: Suppose we are given sequence S = BBCCBCCDCDA and P = B[0, 1]C. The five occurrences of P are $\langle 1, 3 \rangle$, $\langle 2, 3 \rangle$, $\langle 2, 4 \rangle$, $\langle 5, 6 \rangle$, and $\langle 5, 7 \rangle$. In contrast, $\langle 1, 4 \rangle$ and $\langle 1, 6 \rangle$ are not occurrences of P because they do not satisfy the gap constraint.

Problem 2 can adequately consider the influence of noise by setting the wildcard gap constraint. Thus, we must redefine the support of the pattern.

Definition 9 [8]: The pattern support ratio, also called frequency, is

$$f(P, S) = sup(P, S)/ofs(P, S), \tag{5}$$

where

$$ofs(P, S) = kW^{m-1}, \tag{6}$$

is the number of offset sequences of P in S.

If $f(P, S)$ is greater than or equal to a pre-specified threshold ρ, pattern P is a frequent pattern; otherwise, P is an infrequent pattern.

Definition 10 [8]: If a sequence of indices $I = \langle i_1, \ldots, i_m \rangle$ is subject to $N \leq i_j - i_{j-1} - 1 \leq M$ for $2 \leq j \leq m$, I is an offset sequence of P with periodic wildcard gaps $[N, M]$ in S.

Definition 11 [8]: Given sequence S, pattern P, and offset sequence I, we say that P matches S with respect to I if $s_{i_j} = p_j$ for all $1 \leq j \leq k$.

Example 4: Suppose we are given sequence S = BBCCBCCDCDA and P = B[1, 3]C. Because $|S| = 11$ and $W = 3 - 1 + 1 = 3$, $ofs(P, S) = 33$. Then, $sup(P, S) = 6$ because there are six offset sequences (i.e.,$\langle 1, 3 \rangle$, $\langle 1, 4 \rangle$, $\langle 2, 4 \rangle$, $\langle 2, 6 \rangle$, $\langle 5, 7 \rangle$, and $\langle 5, 9 \rangle$) producing matches. Therefore, $f(P, S) = 6/33 = 0.182$.

Sequence pattern mining problems with wildcard gaps can be described as follows: given a sequence, pre-specified threshold, and variable gap length with wildcards between every two consecutive letters, the task is to obtain all frequent patterns with wildcard gaps.

Problem 2: **Sequence pattern mining with wildcard gaps.**

Input: sequence S, support threshold ρ, and wildcard gap $g[N, M]$.

Output: frequent pattern set $\mathcal{P}$ with periodic wildcard gaps.

III. PROPOSED PROBLEM

In this section, we define our new problem. First, we convert the time series into a sequence using an exponential coding function. Once we obtain the sequence, we can employ sequence mining techniques to process it. Second, we define a new pattern mining problem based on weak-wildcard gaps.

A. DATA CONVERSION

Our objective is to discover what we can learn from the data, how we can grasp the characteristics of original data, and how to support human experts in analyzing the data. According to industry experts, the fluctuation can be more interesting than the production. Hence, we propose a coding scheme to convert the time series into a sequence.

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

$$f_i = (t_{i+1} - t_i)/t_i, \tag{7}$$

where $1 \leq i \leq k+1$. We propose the exponential code table shown in Table 1. 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 $S = s_1 s_2 \ldots s_k$.

Example 5: Given a time series t = {1, 1.03, 1.07, 1.12, 1.17, 1.21, 1.26, 1.32, 1.44, 1.50, 1.66, 1.68}, the sequence is S = BBCCBCCDCDA.

The advantage of Table 1 is that it fully reflects the weak changes in the small signal. The small signal in the encoding interval is relatively small, while the large signal on the encoding interval is large.

TABLE 1. Exponential code table.

f_i	Code	f_i	Code
$(-1\%, 1\%)$	O		
$[1\%, 2\%)$	A	$(-2\%, -1\%]$	a
$[2\%, 4\%)$	B	$(-4\%, -2\%]$	b
$[4\%, 8\%)$	C	$(-8\%, -4\%]$	c
$[8\%, 16\%)$	D	$(-16\%, -8\%]$	d
$[16\%, 32\%)$	E	$(-32\%, -16\%]$	e
$[32\%, \text{inf})$	F	$[-100\%, -32\%]$	f

B. SEQUENCE PATTERN MINING WITH WEAK-WILDCARD GAPS

According to Table 1, some characters such as F and f correspond to major changes; hence, they are always important. Others, such as O and A, correspond to minor changes, and they can be ignored in the pattern matching process. We define the concepts of weak-wildcard and patterns with weak-wildcard gaps.

Definition 12: Let $\Omega \subseteq \Sigma$ be the set of **weak characters**. A **weak-wildcard** ψ is a symbol that matches any character in Ω. A **weak gap** $w(N, M)$ is a sequence of weak-wildcards of minimal size N and maximal size M.

Gap $w(N, M)$ is also denoted by (N, M) in a pattern. Accordingly, $\Sigma - \Omega$ is the set of **strong characters**.

With weak gaps, we can define a new type of pattern.

Definition 13: A periodic pattern with weak-wildcard gaps (**PW-pattern**) is a sequence of characters and weak gaps that begins and ends with weak characters. It has the following form

$$P = p_1(N, M)p_2(N, M)\dots(N, M)p_m, \tag{8}$$

where m is the length of the pattern.

In some applications such as oil well production, weak characters form the majority of the data. Most PW-patterns are formed by weak characters. Examples include OOAOO and OAOOO. These patterns are not quite meaningful. Therefore, we propose the following concept.

Definition 14: A periodic strong pattern with weak-wildcard gaps (**PSW-pattern**) $P = p_1(N, M)p_2(N, M)\dots(N, M)p_m$ is a special case of PW-patterns, where for any $1 \le i \le m$, $p_i \in \Sigma - \Omega$.

In other words, characters in the pattern should be *strong*.

Definition 15: 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 PW-pattern. If there exists a position sequence $1 < i_1 < i_2 < \cdots < i_m < k$ such that $s_{i_j} = p_j$ for all $1 \le j \le m$ and $N \le i_j - i_{j-1} - 1 \le M$ for all $1 \le j \le m-1$, $\forall i_{j-1} < c < i_j$, $s_c \in \Omega$, $I = \langle i_1, i_2, \dots, i_m\rangle$ is called an occurrence of P in S with the (N, M) gap constraint.

Example 6: Suppose we are given $S =$ BBCCBCCDCDA, $\Omega = \{O, A, B, a, b\}$, and $P_1 = B(0, 1)\,C(0, 1)C$. In this case, P_1 has three occurrences in S, and the position sequences are $\langle 1, 3, 4\rangle$, $\langle 2, 3, 4\rangle$ and $\langle 5, 6, 7\rangle$. In contrast, the pattern with wildcard gaps P_2 = B[0, 2]C[0, 2]C has two more matches in S, and its position sequences are $\langle 2, 4, 6\rangle$ and $\langle 5, 7, 9\rangle$.

Similar to Problems 1 and 2, we define the following problem.

Problem 3: **Pattern mining with weak-wildcard gaps (PW-pattern mining).**

Input: sequence S, support threshold ρ, set of weak characters Ω, and weak-wildcard gap (N, M).

Output: set of all frequent PW-patterns $\mathcal{P}$.

The reason why we do not consider the problem of PSW-pattern mining here is discussed in the next section.

According to the definition of weak-wildcards, we can use them to accurately express the level of noise in the signal. By introducing weak-wildcards, Problem 3 can more accurately express the degree of noise that we want to ignore. This allows an expert to better determine the frequent sequence patterns of higher practical value.

Plain patterns are also called type I patterns. We refer to patterns with periodic wildcard gaps as type II patterns, PW-patterns as type III patterns, and PSW-pattern as type IV patterns.

Remark 1: A type I pattern can be regarded as a special case of a type II pattern where the wildcard gaps are [0, 0]. A type II pattern can be regarded as a special case of a type III pattern for which $\Omega = \Sigma$.

Therefore, type III patterns are natural extensions of type I and II patterns and can be applied to a wider range of applications.

The concepts of sub-pattern and super-pattern are discussed in [8]. For type III patterns, a sub-pattern is constructed as follows.

Definition 16: Given $P = p_1(N, M)\ p_2(N, M)\ \dots\ (N, M)p_m$ and $1 \le i \le j \le m$, the sub-pattern of P starting at i and ending at j is given by

$$sub(P, i, j) = p_i(N, M)p_{i+1}(N, M)\dots(N, M)p_j. \tag{9}$$

Conversely, P is a super-pattern of $sub(P, i, j)$.

For a sequence, we define a subsequence as follows.

Definition 17: The sub-sequence of $S = s_1 s_2 \dots s_k$ starting at i and ending at j is given by

$$sub(S, i, j) = s_i s_{i+1} \dots s_j. \tag{10}$$

Conversely, S is a super-sequence of $sub(S, i, j)$.

The Apriori property is essential for frequent item-set mining problems [42] as well as frequent pattern mining problems [8]. We therefore examine this property for Problem 3. Similar to [8], the property is described as follows.

Theorem 18: Let P' be a sub-pattern of P,

$$f^*(P, S) \le f^*(P', S). \tag{11}$$

Proof: We only need to consider the condition where P' contains one less character than P. Formally, let $P' = p_1 g(N, M)p_2 \cdots g(N, M)p_{l-1}$ and $P = p_1 g(N, M)p_2 \cdots g(N, M)p_{l-1}g(N, M)p_l$. According to Equation (6),

$$ofs^*(P, S) = ofs^*(P', S)W. \tag{12}$$

Each match of p' corresponds to at most W matches of P, i.e.,

$$N(P, S) \le N^*(P', S)W. \tag{13}$$

Therefore,

$$f^*(P, S) = \frac{N(P, S)}{ofs^*(P, S)} \le \frac{N(P', S)}{ofs^*(P', S)} = f^*(P', S). \tag{14}$$

■

IV. ALGORITHM DESIGN

In this section, we present the algorithm for frequent PW-pattern mining. This algorithm is called the PWM algorithm. Because we only discuss our newly defined type of pattern, we also denote $P = p_1 p_2 \dots p_m$ for simplicity when N and M are indicated.

A. PWM ALGORITHM

Algorithm 1 is the main function of the PWM algorithm. Weak character set Ω and the weak-wildcard gap minimal size N and maximal size M are viewed as constants. They are not included in the parameter list.

Line 1 constructs the set of frequent 1-patterns. The corresponding function is listed in Algorithm 2. This set is

TABLE 2. Notations.

Notation	Meaning
T	A time series
S	A sequence
P	A pattern
$\mathcal{P}$	A frequent pattern set
k	The length of the sequence S
m	The length of the pattern P
Σ	A non-empty finite set {f, e, d, c, b, a, O, A, B, C, D, E, F}
Ω	A set of weak characters
$sup(P, S)$	The number of matches (P in S)
$f(P, S)$	The frequency of pattern P in S
ρ	A frequency threshold
t	The strong pattern filtering threshold
ϕ	A wildcard
ψ	A weak-wildcard
N	The (weak)-wildcard gap minimal size
M	The (weak)-wildcard gap maximal size
$g[N, M]$, or $g[N, M]$	A wildcard gap
$w(N, M)$, or (N, M)	A weak-wildcard gap
$sub(P, i, j)$	The sub-pattern of P starting at index i and ending at j
$sub(S, i, j)$	The sub-sequence of S starting at index i and ending at j
$ofs(P, S)$	The number of offset sequences of P in S
$pop(P, \mathcal{S})$	The popularity of pattern P in a sequence set $\mathcal{S}$
$I = \langle i_1, i_2, \cdots, i_m \rangle$	A sequence of indices

Algorithm 1 PWM

Input: sequence $S = s_1 s_2 \ldots s_k$ and support threshold ρ.
Output: frequent PW-pattern set $\mathcal{P}$.
Constraint: $\forall P \in \mathcal{P}, f(P, S) \geq \rho$.

1: $\mathcal{P}_1$ = findFrequent1Patterns(S, ρ);
2: $\mathcal{P}_2$ = constructLength2Patterns(S, ρ);
3: **for** ($i = 3$; $\mathcal{P}_{i-1}$! = $\emptyset$; i ++) **do**
4: $\quad \mathcal{P}_i = \emptyset$;
5: $\quad \mathcal{C}_i$ = aprioriGenerate($\mathcal{P}_{i-1}$);
6: $\quad$ **for** (each $P \in \mathcal{C}_i$) **do**
7: $\quad\quad sup(P, S) = \sum_{j=1}^{k-|P|+1}$ computeSupport($sub(S, j, k)$, P);
8: $\quad\quad f(P, S) = \frac{sup(P,S)}{k(M-N+1)^{|P|-1}}$;
9: $\quad\quad$ **if** ($f(P, S) \geq \rho$) **then**
10: $\quad\quad\quad \mathcal{P}_i = \mathcal{P}_i \cup \{P\}$;
11: $\quad\quad$ **end if**
12: $\quad$ **end for**
13: **end for**
14: return $\mathcal{P} = \cup_{j=1}^{i-1} \mathcal{P}_j$;

Algorithm 2 findFrequent1Patterns

Input: sequence S and support threshold ρ.
Output: frequent 1-patterns $\mathcal{P}_1$.

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

denoted by $\mathcal{P}_1$. We generate the $\mathcal{P}_2$ in order to improve the effectiveness of the algorithm.

Patterns with longer lengths are then mined in the `for` loop, which continues until no frequent pattern of the current size exists.

Line 5 generates candidate patterns with length i based on frequent patterns with length $i - 1$. The corresponding function is listed in Algorithm 3. This set is denoted by C_i.

Line 7 computes the support of each candidate pattern, which is the sum of the occurrences of P starting from different positions in S. The corresponding function is listed in Algorithm 4. Note that this step is more time consuming than pattern generation.

Line 8 computes the frequency of the candidate pattern.

Line 9 selects patterns whose frequency meets the threshold. The result set is denoted by $\mathcal{P}_i$.

Finally, Line 14 constructs and returns the set of all frequent patterns. This set is denoted by $\mathcal{P}$.

We use the following example to illustrate the process.

Example 7: Let S = DABCDBACDA, $\rho = 0.1$, $N = 0$, $M = 1$, and Ω = {O, A, a}.

After the execution of Line 1, we have $\mathcal{P}_1$ ={A, B, C, D}.

Next, we have $\mathcal{C}_2$ = {AA, AB, AC, AD, BA, BB, BC, BD, CA, CB, CC, CD, DA, DB, DC, DD} after the execution of Line 5. This shows all of the possible combinations that can be formed from $\mathcal{P}_1$.

For each pattern in $\mathcal{C}_2$, the support is calculated. For example, let P = BC, after executing Line 6, we have $f(P, S) = 2/20 = 0.1$. According to Line 9, $f(P, S) \geq \rho$, and BC will be included in $\mathcal{P}_2$. After the execution of Lines 6–12, we have $\mathcal{P}_2$ = {BC, CD, DA, DB}.

Algorithm 3 aprioriGenerate

Input: frequency sequence patterns $\mathcal{P}_{i-1}$ and $\mathcal{P}_2$.
Output: candidate sequence patterns $\mathcal{C}_i$.

1: **for** (each $P \in \mathcal{P}_{i-1}$) **do**
2: **for** (each $P' \in \mathcal{P}_2$) **do**
3: $C = P \oplus P'$;
4: **if** ($C == \varepsilon$) **then**
5: continue;
6: **end if**
//prune
7: **if** ($sub(C, 2, |C|) \notin \mathcal{P}_{i-1}$) **then**
8: continue;
9: **end if**
10: $\mathcal{C}_i = \mathcal{C}_i \cup \{C\}$;
11: **end for**
12: **end for**
13: return $\mathcal{C}_i$;

Similarly, we have $\mathcal{P}_3 = \{\text{BCD}, \text{DBC}\}$, $\mathcal{P}_4 = \{\text{DBCD}\}$ and $\mathcal{P}_5 = \emptyset$. Therefore, we do not need to compute $\mathcal{P}_5$, and the `for` loop terminates.

Finally, we have $\mathcal{P} = \{\text{A}, \text{B}, \text{C}, \text{D}, \text{BC}, \text{CD}, \text{DA}, \text{BCD}, \text{DBC}, \text{DBCD}\}$.

Because the algorithm is based on the Apriori property (Theorem 18), it is complete.

Property 1: The output of Algorithm 1 is complete.

Proof: Suppose that there exists a pattern $P = p_1p_2 \dots p_m$, $P \notin \mathcal{P}_m$, and $f(P, S) \geq \rho$. Then P should not be tested in Algorithm 1. According to Lines 2 through 11, $P' = P_{m-1} = p_1p_2 \dots p_{m-1}$ should not be tested and $P_{m-1} \notin \mathcal{P}_{m-1}$. For the same reason, $P_{m-2}, P_{m-3}, \dots, P_m$ are not tested in Algorithm 1. However, P_1 is a pattern with only one character. According Line 1, it must be tested. Therefore, the hypothesis does not hold, indicating that the algorithm is able to mine all frequent patterns. ■

B. PATTERN GROWTH ALGORITHM

Algorithm 3 generates $\mathcal{C}_i$ from $\mathcal{P}_{i-1}$. Note that $\mathcal{C}_i$ is a superset of $\mathcal{P}_i$. Pattern growth entails concatenating two patterns. Let $P = p_1p_2 \dots p_m$ and $P' = p'_1p'_2 \dots p'_{m'}$ be two patterns with weak-wildcard gaps (N, M). The concatenation operation is defined as

$$P \oplus P' = \begin{cases} p_1p_2 \dots p_mp'_2 \dots p'_{m'}, & \text{if } p'_1 = p_m; \\ \varepsilon, & \text{otherwise .} \end{cases} \quad (15)$$

Here, ε is the empty string. In other words, P and P' can be concatenated iff $p'_1 = p_m$.

In Line 3, we always concatenate pattern P with another pattern $P' \in \mathcal{P}_2$. In this way, the algorithm is complete. This is discussed in Property 1.

Lines 4 through 6 filter out empty patterns. According to Equation (15), empty patterns indicate that the respective patterns cannot be concatenated.

Algorithm 4 computeSupport

Input: sequence $S = s_1s_2 \dots s_k$ and pattern $P = p_1p_2 \dots p_m$.
Output: occurrence count.

1: **if** ($s_1 != p_1$) **then**
2: return 0;//The first position does not match
3: **end if**
4: **if** ($m == 1$) **then**
5: return 1;//One occurrence
6: **end if**
//Try to match the remaining part of P
7: **for** ($i = N + 1$; $i \leq M + 1$; i ++) **do**
8: **if** ($i + 1 \geq k$) **then**
9: break; //Exceeds the bound
10: **end if**
11: **if** ($s_i \notin \Omega$) **then**
12: break; //Cannot be viewed as a weak-wildcard
13: **end if**
14: count += computeSupport($sub(S, i + 1, k)$, $sub(P, 2, m)$);
15: **end for**
16: return count;

Lines 7 through 9 correspond to the pruning technique. Here, $sub(C, 2, |C|)$ is the sub-pattern of C without the last characters. According to Theorem 18, C is infrequent if $sub(C, 2, |C|)$ is infrequent. This technique facilitates the removal of some candidate patterns. In fact, the algorithm can obtain the correct output even without this technique. However, as discussed in Section V-A, it can reduce the runtime significantly.

C. COMPUTING THE PATTERN OCCURRENCES

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.

In Lines 1 through 3, the result is set to zero if the pattern does not match the first position of S. Lines 4 through 6 indicate that there is exactly one match if the length of P is one. Lines 7 through 15 attempt to skip some characters matching weak-wildcards. Lines 8 through 10 ensure that the bound of the sequence is not exceeded. Lines 11 through 13 verify if the character is weak.

Line 14 is the core code and invokes the function recursively with a shorter sequence and shorter pattern.

Example 8: Let $S = \text{BBCBCCA}$, $N = 1$, $M = 2$, $\Omega = \{\text{A}, \text{B}\}$, $P_1 = \text{CBC}$, and $P_2 = \text{BCC}$. Denote Algorithm 4 as $cr(P, S)$.

We have $cr(P_1, S) = 0$ because $s_1 = \text{B}$ while $p_1 = \text{C}$.

Tracking through the algorithm gives us

$cr(P_2, S) = cr(\text{BCC}, \text{BBCBCCA}) = cr(\text{CC}, \text{CBCCA}) + cr(\text{CC}, \text{BCCA})$

$= cr(\text{C}, \text{CCA}) + cr(\text{C}, \text{CA}) + 0 = 1 + 0 = 1$.

We now analyze the time complexity of Algorithm 4. In the worst case, all possible position indices are checked.

Therefore, the time complexity is

$$O(W^m), \tag{16}$$

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

$$O(kW^m). \tag{17}$$

D. FREQUENT PSW-PATTERN MINING

As discussed in Definition 14, we would sometimes like to represent patterns with only strong characters. There are two approaches to the mining of this type of pattern. If frequent patterns have already been mined, we can filter out these strong patterns. Otherwise, we mine them directly from the sequence. We only need to replace $a \in \Sigma$ with $a \in \Sigma - \Omega$ in Algorithm 2.

Example 9: Let S = DBCDBACD, $\rho = 0.1$, $N = 0$, $M = 1$, and $\Omega = \{O, A, a\}$.

The set of PSW-patterns is $\mathcal{P} = \{B, C, D, BC, CD, DB, BCD, DBC, DBCD\}$.

V. EXPERIMENTAL RESULTS

In this section, we first demonstrate the efficiency of the pruning technique. Next, we present a frequent pattern and reproduce it accurately in the original time series. Third, we prove that we can effectively solve the variable-length problem in [4] through a set of comparative experiments.

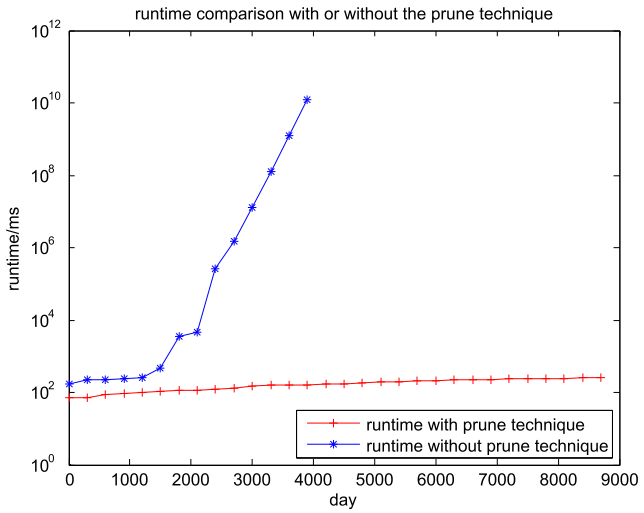

FIGURE 2. Runtime comparison with and without the pruning technique.

A. EFFICIENCY OF THE PRUNING TECHNIQUE

Fig. 2 compares the runtime with and without our pruning technique. For the algorithm without the pruning technique, the runtime does not increase significantly when the sequence length is no more than 2,000. In this situation, the runtime is essentially determined by the program overhead. However, the runtime increases exponentially as *day* increases beyond 2,000. This is because too many candidate patterns are generated, and the time complexity of the pattern support computation is exponential with respect to the pattern length, as shown in Equation (17). In contrast, with the pruning technique, the runtime increases only marginally, even for sequences with $day > 10,000$. This indicates that many candidate patterns are successfully pruned. Thus, the pruning technique is effective and necessary.

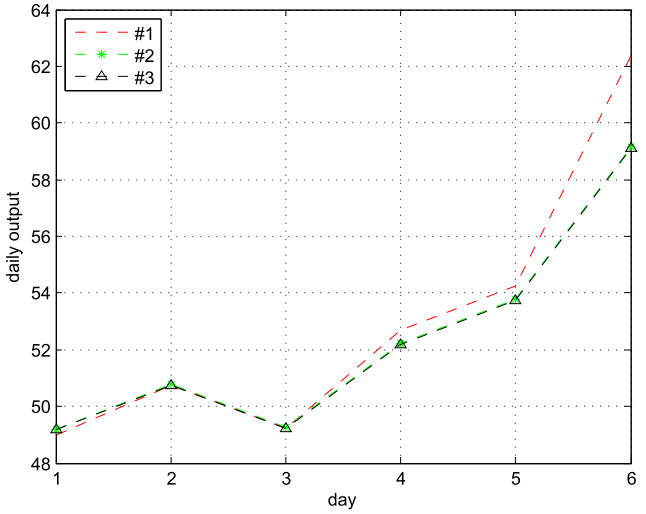

FIGURE 3. Pattern discovery based on the Apriori property.

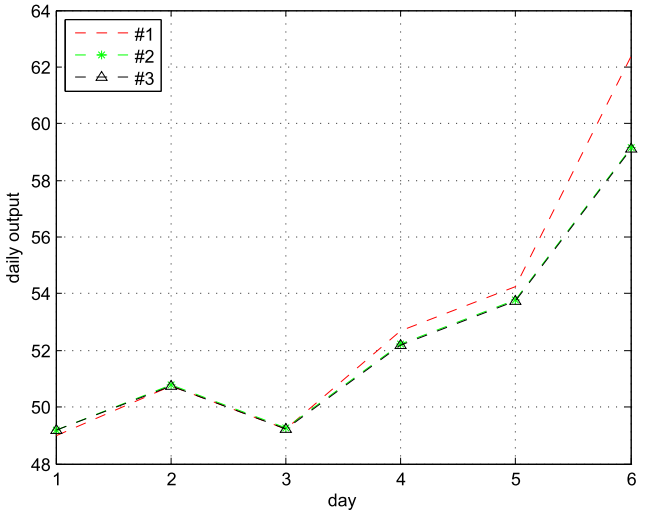

FIGURE 4. Pattern discovery based on Euclidean distance.

B. PATTERN DISCOVERY

We adopt two methods for discovering the effective patterns using the same data sets. First, we use the PWM algorithm proposed in this paper for pattern mining, which can discover many meaningful patterns. Fig. 3 lists the three appearances of the pattern BbCBD in the original time series. Second, we use the EMMA algorithm [4] to discover frequent patterns based on the Euclidean distance. Three occurrences of pattern BbCBD in the original time series are shown in Fig. 4. From the figure, it is clear that both methods can discover

meaningful patterns from the time series. Moreover, they can identify all appearances accurately in the original time series.

C. DISCOVERING FREQUENT PATTERNS OF DIFFERENT LENGTHS

In Fig. 4, it appears that we can only discover frequent patterns of equal length using the method in [4]. However, time delays are ubiquitous in the real environment, leading to a certain signal shift. Thus, there is a large number of variable-length patterns in practice.

As seen in Fig. 5, we can solve the variable-length problem by the introduction of wildcards. We discover the patterns from the same motif of lengths 6, 8, 9, and 10. However, a serious problem is that one may find completely dissimilar patterns, such as patterns 7, 8 and 9 in Fig. 5. This is intolerable in practical applications.

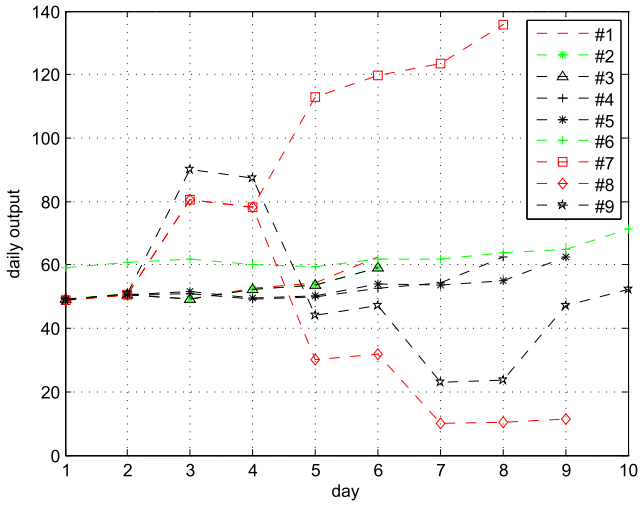

FIGURE 5. Frequent patterns with wildcard gaps.

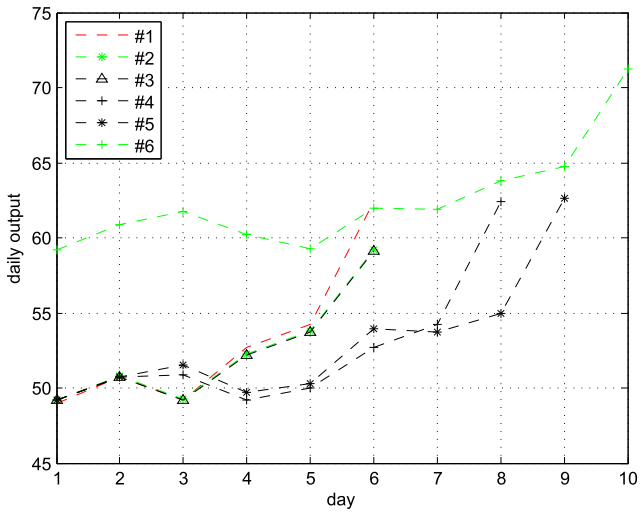

FIGURE 6. Frequent patterns with weak-wildcard gaps.

In Fig. 6, dissimilar patterns were successfully ignored, and pattern 6 was discovered. Pattern 6 has a different data starting point, but it is similar to patterns 1, 2, 3, 4 and 5. Clearly, they all come from the same motif. Thus, we can solve the variable-length problem with high similarity.

TABLE 3. Some PSW-patterns from oil production data.

Rank	Pattern	Curves
1	eBbCBD	
2	cDdbcD	
3	DdDccD	
4	DeBbCB	
5	DBbCBD	
6	DeDBbC	

Table 3 lists some PSW-patterns from a petroleum production time series. Here, we let $W = \{O, a, A\}$, that is, only fluctuations smaller than 2% are viewed as ignorable. The length of all these patterns is 5. We observe that they have different shapes, which might be explained by experts as meaningful.

VI. CONCLUSIONS

In this paper, we proposed a new approach for analyzing time-series data using a sequence pattern mining algorithm. Specifically, we designed a code table to convert the time series into a sequence. We defined the concept of weak-wildcard and PW-patterns, and proposed the PWM algorithm to find frequent PW-patterns and PSW-patterns. Experiments on a petroleum production data set show that the new algorithm can find similar patterns while filtering out dissimilar ones. Moreover, the pruning technique is very effective.

From the viewpoint of algorithms and applications, the following research topics merit further investigation:

1) The code table determines the granularity [13]–[16] of the data. Hence, it is essential for applications. In this paper, we use an exponential code table, but other types of code tables are also feasible. The code table can be adjusted considering user feedback.
2) The incorporation of human expert feedback in this work is quite basic. Further feedback could facilitate pattern evaluation and parameter adjustment. New scenarios and their corresponding algorithms are required to enhance the usability of the system.

3) Each pattern can be viewed as a feature of a well. With these features, we can study the characteristics of each well or calculate the similarity between wells. The related feature extraction topic is relevant for major machine learning problems [43]–[45].
4) We can explore other applications for our data conversion, sequence mining algorithm, and pattern filtering techniques because they are valid for any time series. We can apply these techniques to any time-series data including stock pricing and weather forecasting. The characteristics intrinsic to the respective applications should then be addressed.

Our novel approach and findings are useful and directly applicable to petroleum data mining, time-series analysis, sequence pattern mining, and granular computing research.

REFERENCES

[1] T.-C. Fu, F.-L. Chung, V. Ng, and R. Luk, "Pattern discovery from stock time series using self-organizing maps," in *Proc. KDD Workshop Temporal Data Mining*, 2001, pp. 26–29.
[2] S.-Q. Zheng, H.-F. Zhang, and J.-W. Bao, "Application of chaotic theory to oil production rate time series prediction," in *Proc. IEEE Int. Conf. Intell. Comput. Intell. Syst.*, vol. 3, 2009, pp. 690–693.
[3] C. C. Aggarwal, Y. Li, J.-Y. Wang, and J. Wang, "Frequent pattern mining with uncertain data," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 29–38.
[4] J. L. E. K. S. Lonardi and P. Patel, "Finding motifs in time series," in *Proc. 2nd Workshop Temporal Data Mining*, 2002, pp. 53–68.
[5] R. Iváncsy and I. Vajk, "Frequent pattern mining in web log data," *Acta Polytechnica Hungarica*, vol. 3, no. 1, pp. 77–90, 2006.
[6] A. Ng and A. W.-C. Fu, "Mining frequent episodes for relating financial events and stock trends," in *Advances Knowledge Discovery Data Mining*. Berlin, Germany: Springer, 2003, pp. 27–39.
[7] S.-Y. Liu, L. Kang, L. Chen, and L. M. Ni, "How to conduct distributed incomplete pattern matching," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 4, pp. 982–992, Apr. 2014.
[8] F. Min, Y.-X. Wu, and X.-D. Wu, "The apriori property of sequence pattern mining with wildcard gaps," *Int. J. Funct. Informat. Personalised Med.*, vol. 4, no. 1, pp. 15–31, 2012.
[9] D. Gusfield, *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge, U.K.: Cambridge Univ. Press, 1997.
[10] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proc. 8th ACM SIGMOD Workshop Res. Issues Data Mining Knowl. Discovery*, 2003, pp. 2–11.
[11] N. Zhong, Y.-F. Li, and S.-T. Wu, "Effective pattern discovery for text mining," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 1, pp. 30–44, Jan. 2012.
[12] C. Giannella, J.-W. Han, J. Pei, X.-F. Yan, and P. S. Yu, "Mining frequent patterns in data streams at multiple time granularities," in *Next Generation Data Mining*, vol. 212. New York, NY, USA: MIT Press, 2003, pp. 191–212.
[13] Y. Y. Yao, "Granular computing: Basic issues and possible solutions," in *Proc. 5th Joint Conf. Inf.Sciences.*, vol. 1, 2000, pp. 186–189.
[14] D.-Q. Miao, G.-Y. Wang, Q. Liu, T.-Y. Lin, and Y. Y. Yao, *Granular Computing: Past, Present And Future Prospects*. Beijing, China, Science Press, 2007.
[15] W. Pedrycz, A. Skowron, and V. Kreinovich, *Handbook of Granular Computing*. Hoboken, NJ, USA: Wiley, 2008.
[16] D. Liu, D. Liang, and C.-C. Wang, "A novel three-way decision model based on incomplete information system," *Knowl. Based Syst.*, vol. 91, pp. 32–45, Jan. 2016.
[17] S.-M. Gu and W.-Z. Wu, "On knowledge acquisition in multi-scale decision systems," *Int. J. Mach. Learn. Cybern.*, vol. 4, no. 5, pp. 477–486, 2013.
[18] W.-H. Xu and W.-T. Li, "Granular computing approach to two-way learning based on formal concept analysis in fuzzy datasets," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 366–379, Feb. 2016.
[19] J. Yang, W. Wang, P. S. Yu, and J.-W. Han, "Mining long sequential patterns in a noisy environment," in *Proc. ACM SIGMOD*, 2002, pp. 406–417.
[20] Y.-X. Wu, L.-L. Wang, J.-D. Ren, W. Ding, and X.-D. Wu, "Mining sequential patterns with periodic wildcard gaps," *Appl. Intell.*, vol. 41, no. 1, pp. 99–116, 2014.
[21] H.-R. Zhang and F. Min, "Three-way recommender systems based on random forests," *Knowl. Based Syst.*, vol. 91, pp. 275–286, Jan. 2016.
[22] X. Liang, R.-C. Chen, Y. He, and Y. Chen, "Associating stock prices with web financial information time series based on support vector regression," *Neurocomputing*, vol. 115, pp. 142–149, Sep. 2013.
[23] H. H. Nguyen and C. W. Chan, "Applications of data analysis techniques for oil production prediction," *Eng. Appl. Artif. Intell.*, vol. 18, no. 5, pp. 549–558, 2005.
[24] G. G. Žylius, R. Simutis, and V. Vaitkus, "Evaluation of computational intelligence techniques for daily product sales forecasting," *Int. J. Comput.*, vol. 14, no. 3, pp. 157–164, 2015.
[25] D. E. Keller, A. M. Fischer, M. A. Liniger, C. Appenzeller, and R. Knutti, "Synthetic future daily weather time-series over Switzerland in consistency with RCM projections," *EGU General Assembly*, vol. 17, p. 10823, Apr. 2015.
[26] A. Anguera, J. A. Lara, D. Lizcano, and M. A. Martínez, and J. Pazos, "Sensor-generated time series events: A definition language," *Sensors*, vol. 12, no. 9, pp. 11811–11852, 2012.
[27] B.-K. Yi and C. Faloutsos, "Fast time sequence indexing for arbitrary lp norms," in *Proc. VLDB*, 2000, pp. 385–394.
[28] X.-P. Ge and P. Smyth, "Deformable Markov model templates for time-series pattern matching," in *6th ACM SIGKDD*, 2000, pp. 81–90.
[29] K.-P. Chan and A. W.-C. Fu, "Efficient time series matching by wavelets," in *Proc. IEEE 15th Int. Conf. Data Eng.*, Mar. 1999, pp. 126–133.
[30] S.-Y. Liu, L. Kang, L. Chen, and L. Ni, "Distributed incomplete pattern matching via a novel weighted bloom filter," in *Proc. IEEE 32nd Int. Conf. Distrib. Comput. Syst.*, Jun. 2012, pp. 122–131.
[31] H. Tang and S. S. Liao, "Discovering original motifs with different lengths from time series," *Knowl. Based Syst.*, vol. 21, no. 7, pp. 666–671, 2008.
[32] Y. Tanaka, K. Iwamoto, and K. Uehara, "Discovery of time-series motif from multi-dimensional data based on MDL principle," *Mach. Learn.*, vol. 58, no. 2, pp. 269–300, 2005.
[33] A. Mueen, E. J. Keogh, Q. Zhu, S. Cash, and B. Westover, "Exact discovery of time series motifs," in *Proc. SDM*, 2009, pp. 493–498.
[34] S. Gupta, A. Ray, and E. Keller, "Symbolic time series analysis of ultrasonic data for early detection of fatigue damage," *Mech. Syst. Signal Process.*, vol. 21, no. 2, pp. 866–884, 2007.
[35] A. Khatkhate, A. Ray, E. Keller, S. Gupta, and S. C. Chin, "Symbolic time-series analysis for anomaly detection in mechanical systems," *IEEE/ASME Trans. Mechatron.*, vol. 11, no. 4, pp. 439–447, Aug. 2006.
[36] C. Piccardi, "On parameter estimation of chaotic systems via symbolic time-series analysis," *An Interdiscipl. J. Nonlinear Sci.*, vol. 16, no. 4, p. 043115, 2006.
[37] S.-J. Yen and Y.-S. Lee, "Mining time-gap sequential patterns," in *Proc. Adv. Rese. Appl. Artifi. Intell.*, 2012, pp. 637–646.
[38] M.-H. Zhang, B. Kao, D. W. Cheung, and K. Y. Yip, "Mining periodic patterns with gap requirement from sequences," in *Proc. SIGMOD*, 2007, pp. 623–633.
[39] X.-N. Ji, J. Bailey, and G.-Z. Dong, "Mining minimal distinguishing subsequence patterns with gap constraints," in *Proc. ICDM*, 2005, pp. 194–201.
[40] C. Li and J.-Y. Wang, "Efficiently mining closed subsequences with gap constraints," in *Proc. SDM*, 2008, pp. 313–322.
[41] G. Chen, X.-D. Wu, X.-Q. Zhu, A. N. Arslan, and Y. He, "Efficient string matching with wildcards and length constraints," *Knowl. Inf. Syst.*, vol. 10, no. 4, pp. 399–419, 2006.
[42] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *ACM SIGMOD Rec.*, vol. 22, no. 2, pp. 207–216, 1993.
[43] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, *Feature Extraction: Foundations and Applications*, vol. 207. New York, NY, USA: Springer, 2008.
[44] J.-H. Li, C.-L. Mei, and Y.-J. Lv, "Incomplete decision contexts: Approximate concept construction, rule acquisition and knowledge reduction," *Int. J. Approx. Reasoning*, vol. 54, no. 1, pp. 149–165, 2013.
[45] L.-Y. Wen and F. Min, "A granular computing approach to symbolic value partitioning," *Fundam. Inf.*, vol. 142, nos. 1–4, pp. 337–371, 2015.

CHAO-DONG TAN received the M.S.E. degree in oil production engineering, with a major in oil development, from 1992 to 1995, and the Ph.D. degree from China Petroleum University, Beijing. He studied the CIDA Project from CoreLab Company, Canada, in 1998. His current research interests include data mining and petroleum production engineering.

FAN MIN (M'09) received the M.S. and Ph.D. degrees from the School of Computer Science and Engineering, University of Electronics Science and Technology of China, Chengdu, China, in 2000 and 2003, respectively. He visited the University of Vermont, Burlington, Vermont, from 2008 to 2009. He is currently a Professor with Southwest Petroleum University, Chengdu. He has authored over 110 refereed papers in various journals and conferences, including the *Information Sciences*, *International Journal of Approximate Reasoning*, and *Knowledge-Based Systems*. His current research interests include data mining, recommender systems, and granular computing.

MIN WANG received the B.S. degree from the Southwest University of China, Chongqing, in 2002, and the M.S. degree from the School of Electrical Engineering and Information, Southwest Petroleum University, Chengdu, China. She is currently an Associate Professor with Southwest Petroleum University. Her research interests include data mining, granular computing, and signal processing.

HENG-RU ZHANG received the M.S. degree from the School of Mechatronics Engineering, University of Electronics Science and Technology of China, Chengdu, China, in 2002. He is currently an Associate Professor with Southwest Petroleum University, Chengdu. He has authored over ten refereed papers in various journals and conferences, including the *Information Sciences* and *Knowledge-Based Systems*. His current research interests include data mining, recommender systems, and granular computing.

ZHI-HENG ZHANG is currently pursuing the Ph.D. degree with Southwest Petroleum University, Chengdu, China.