Efficient Mining of Event Periodicity in Data Series

Hua Yuan(✉), Yu Qian, and Mengna Bai

School of Management and Economics,
University of Electronic Science and Technology of China,
Chengdu 611731, China
{yuanhua,qiany}@uestc.edu.cn, mengnabai@163.com

Abstract. This paper investigates the problem of efficiently discovering periodicity of a certain event in data series. To that end, the current work argues firstly that the periodicity of an event in data series may be formalized as the distribution period, the structure period, or the both. Along this line, a partition method, $\pi(n)$, is proposed to divide the data series into length-equal and position-continuous segments. Based on the results of implementing $\pi(n)$ on a data series, we propose two new concepts of distribution periodicity and structure periodicity. Then, a cross-entropy-based method, namely CEPD, is proposed to mine the periodicity of data series. The experimental results show that CEPD can be used to mine feasible event periodicity in data series, especially, with very low level of time consumption and high capability of noise resilience.

Keywords: Data series · Cross entropy · Distribution periodicity · Structure periodicity

1 Introduction

Data series is commonly used in presentation of the events sequentially happened in real world, such as the weather data for a location [22], the gene expression data [12], the finance fluctuation data [20,32], the web site visiting traffic [5,29], and the consumption sequence of a user [1]. Data series is mostly characterized by being composed of repeating cycles [19], especially, for those data series generated by user behaviors [30]. For instance, "The vendors purchase *twice a month* from the suppliers," "Bob visits gym *every Tuesday*," and so on. Basically, such repeating patterns could reveal important observations about the behavior and future trends of the events represented by the data series, and hence would lead to more effective decision making [16]. These gave rise to an important process for mining regular patterns within a data series.

In general, event(s) may show three types of periodicity in a data series: the symbol periodicity, the partial (sequence) periodicity, and the full-cycle (segment) periodicity [19]. Given a periodicity mining task, the methods proposed in the literature would like to treat the task of periodicity detection as a process of finding temporal regularities within the data series [8,13,19,28,30]. Although these methods had performed well for event periodicity mining in certain situations, nevertheless, they also face some technical challenges:

– First, the common problem for these approaches is their computational performance, especially in a big data environments. To address this issue, they would assume previously that users either know the value of the period beforehand or are willing to try various period values until satisfactory periodic patterns emerge [7]. However, if there are multiple events embedded in a data series, then more prior information is needed for event periodicity mining task, or it will make the mining methods present relative poor performance, both on efficiency and completeness [31].
– Second, these methods mainly identify the structural periodicity of the events over a set of periods (time intervals), i.e., those events which occurred at a fixed position in each period may be considered as having periodicity [13].
– Third, data collected from the real-world, which is the input of mining algorithms, are affected by several components; among them, noise is an unavoidable problem [25]. Therefore, the event periodicity mining methods are expected to provide better robustness to noise [11,18].

In this work, we will introduce the *distribution periodicity* and *structure periodicity* to measure the periodic information of an event in a data series. The main contributions of this paper lie in two aspects: first, to the best of our knowledge, it is the first time to distinguish the idea of distribution periodicity and structure periodicity; second, based on the minimum cross entropy principle, an efficient method is proposed to mine the periodicity of an event in data series, which also has a better performance on noise resilience.

2 Related Work

There are lots of studies proposed in the literature of data stream mining. In summary, they can be categorized into types of signal-processing-based, data-structure-based, and statistics-based method.

Signal-Processing-Based Method. The signal processing method in periodic pattern mining is mainly reflected in the data processing and transformation. [6] used the Haar Wavelet Transform and discrete fourier transform (DFT) for time series indexing. The algorithm presented by [24] is the first one that exploits the information in both periodogram and autocorrelation to provide accurate periodic estimates without upsampling. In this work, both DFT and power spectral density (PSD) estimation method are introduced to deal the time series data. A convolution-based algorithm is proposed for segment periodicity and symbol periodicity, and the periodic patterns of unknown periods are also discovered without affecting the time complexity [7]. As pointed out in [18] and [13], the fast Fourier transform (FFT) [3] can also be used to identify periodicity. However, there are two problems in the FFT method. First, it does not cope well with random off-segments in periodic patterns. Further, the computational efficiency is very complicate when events in data series are sparse [30].

Data-Structure-Based Method. In earlier studies, the work in [10] use a sliding window over the data sequence and extract its features, then [26] presented mining technology from a time series database based on a moving-window. Also, by using an expanding sliding windows, [9] improved the accuracy of the discovered periodicity rates. In recent, a pattern-growth approach which is based on a tree structure, called Periodic Frequent-tree (PF-tree) has been discussed for mining periodic patterns [23]. In the paper of [23], the authors use a so called Periodic-frequent pattern tree to capture the database contents and generate the complete set of periodic-frequent patterns.

Since partial periodicity is very common in practice, [13] studied an interesting data mining problem of searching for partial periodic patterns in time-series databases, their algorithm based on a max-subpattern tree offers excellent performance. Promoted by this research, [4] proposed a new structure, the abbreviated list table (ALT), and several efficient algorithms to compute the partial periods. Sheng et al. [21] developed an algorithm to utilize optimization steps to find dense periodic areas in the time series.

Statistics-Based Method. Some basic static methods such as autocorrelation and ranking are commonly used. [2] proposed an algorithm for finding approximate periodicities in large time series data, utilizing autocorrelation function and FFT. And it can discover weak periodic signals in time series databases. [14] investigated an interesting type of periodic pattern, called partial periodic (PP) correlation. Especially, a more suitable measurement, information, is introduced in [27] to naturally value the degree of surprise of the pattern within a data sequence. In [30], the authors presented a variance-based approach to model periodicity, which is to detect event periodicity basing on the statistical variance of the gaps at which a pattern occurs in data series (i.e., the variance of the interarrival times of the pattern). Ghosh et al. [11] have demonstrated the use of a sequential Monte Carlo method to detect and track the periodicity in discrete event streams. Unlike other methods, this technique does not rely on the underlying process sticking to a constant phase.

3 Concepts and Model Formulation

3.1 Data Series and Event Periodicity

A data series S is an ordered sequence of $|S|$ feature values:

$$S = (s_1 s_2 ... s_t ... s_{|S|}),\ \ s_t \in \mathbb{R}, \tag{1}$$

where s_t is the value of the feature at position t, for example, the feature might be the daily average stock price of a company.

In order to facilitate the calculation, we usually transfer S into a more easy-to-compute formation. If we discretize the feature values in S into nominal discrete events (e.g, stock price goes "up", "down" or "flat"), then the set of feature values can be denoted as $\Sigma = \{a, b, c, ...\}$[7] by representing each event as a symbol (i.e., $a =$ "goes up", $b =$ "goes down", $c =$ "goes flat", etc.). As a result, S can be viewed as a sequence of $|S|$ events (symbols) drawn from a finite event set of Σ_S. Further more, let e_t be any event occurred at position t in S, then a set of events happened sequentially over **continuous position space** of $[1, |S|]$ can be specified as follows:

$$S = (e_1 e_2 ... e_t ... e_{|S|}),\ e_t \in \Sigma_S. \tag{2}$$

Using x to denote the focal event in Σ_S, and $\tilde{x}$ denotes any event type in $\Sigma_S \setminus \{x\}$, if $e_t = x$, then we said that event x appears at position t in S. Especially, when we only concern about whether event x occurred at position t, BS_x is then can be encoded as a **binary data series**,

$$BS_x = (b_1 b_2 ... b_t ... b_{|S|}),\ b_t \in \{0, 1\}. \tag{3}$$

where b_t is specified as follows:

$$b_t = \begin{cases} 1 & \text{if } e_t = x \text{ (i.e., } x \text{ appears at position } t\text{);} \\ 0 & \text{otherwise (i.e., } \tilde{x} \text{ appears at position } t\text{).} \end{cases}$$

Definition 1. *An event $x \in \Sigma_S$ is said to have* **periodicity** *(or x is a periodic event) in data series S, if its appearances are shown repeated periodically in S.*

Apparently, if event x appears periodically in data series S, we can expect that the appearances of code "b_t" in BS_x are also periodically.

3.2 Data Series Partitioning

To mine the periodicity of event x in data series S, a feasible way is to divide S into segments [17].

Given a set of partition methods $\prod$ defined over S, if there always exists a partitioning scheme $\pi(n) \in \prod$ (where $n < |S|$) such that n is the (distribution or structure) period of event x, then $\prod$ is called a **complete partition set** with respect to the periodicity of x in S. Moreover, $\pi(n)$ is then called a **"good" partition** for detecting the periodicity of x in S.

Accordingly, we propose a simple and complete partitioning method, i.e., $\pi(n)$, $n \in [1, |S|]$, to divide the data series S into segments iteratively as follows:

Step 1: Begin with the first position $t = 1$;
Step 2: Every n position-continuous elements are partitioned into a same segment P, i.e., the first n events are in P_1, the second n events are in P_2, and so on. As a result, $\pi(n) = \{P_1 | P_2 | \cdots | P_{\lceil \frac{|S|}{n} \rceil}\}$ partitions S into $\lceil \frac{|S|}{n} \rceil$ length-equal[1] segments, and $P_j = \big(e_{(j-1)*n+1}, \cdots, e_{j*n}\big)$, where $j \in [1, \lceil \frac{|S|}{n} \rceil]$.

[1] $|P_{\lceil \frac{|S|}{n} \rceil}| \le n$ is allowed.

For example, $BS = (00111000100110001000)$, then method $\pi(4)$ will partition BS into five segments as follows:

$$\pi(4) = \{\underbrace{0011}_{\mathcal{P}_1}|\underbrace{1000}_{\mathcal{P}_2}|\underbrace{1001}_{\mathcal{P}_3}|\underbrace{1000}_{\mathcal{P}_4}|\underbrace{1000}_{\mathcal{P}_5}\}.$$

By implementing $\pi(n) = \{P_1|P_2|\cdots|P_{\lceil\frac{|S|}{n}\rceil}\}$ on a binary time-series BS, we can obtain a **partition matrix** by rearranging all the segments P_j as follows:

$$\left\|\begin{matrix} P_1 \\ P_2 \\ \vdots \\ P_j \\ \vdots \\ P_{\lceil\frac{|S|}{n}\rceil} \end{matrix}\right\| = \left\|\begin{matrix} b_1 & \dots & b_n \\ b_{n+1} & \dots & b_{2n} \\ \vdots & \dots & \vdots \\ b_{(j-1)*n+1} & \dots & b_{j*n} \\ \vdots & \dots & \vdots \\ b_{(\lceil\frac{|S|}{n}\rceil-1)*n+1} & \dots & b_{(\lceil\frac{|S|}{n}\rceil)*n} \end{matrix}\right\|.$$

In general, the matrix has a total of $\lceil\frac{|S|}{n}\rceil$ rows and n columns. Row $j \in [1,\cdots,\lceil\frac{|S|}{n}\rceil]$ is just the contents of j-th segment, i.e., P_j. Column $\tau \in [1,\cdots,n]$ is corresponding to the appearances of event x at the τ-th position, which is referred to as $C_\tau = \left(b_\tau, ..., b_{(j-1)*n+\tau}, ..., b_{(\lceil\frac{|S|}{n}\rceil-1)*n+\tau}\right)^T$.

Definition 2. *The total appearances of event x in segment P_j ($j \in [1,\cdots,\lceil\frac{|S|}{n}\rceil]$) is called the* **support** *of x in P_j, it is defined as*

$$supp(x|P_j) = \sum_{b_t \in P_j} b_t. \tag{4}$$

Lemma 1. *$supp(x|P_j) + supp(\tilde{x}|P_j) = n$, $j \in [1,...,\lceil\frac{|S|}{n}\rceil - 1]$*[2].

Definition 3. *The total appearances of event x in C_τ ($\tau \in [1,\cdots,n]$) is called the* **support** *of x at position τ, which is represented by:*

$$supp(x|C_\tau) = \sum_{b_t \in C_\tau} b_t. \tag{5}$$

Lemma 2. *$supp(x|C_\tau) + supp(\tilde{x}|C_\tau) = \lceil\frac{|S|}{n}\rceil$, $\tau = 1,...,n$.*

Further more, we can define that the total appearances of event x in S is called the **support** of x, which is defined as

$$supp(x) = \sum_{b_t \in BS_x} b_t. \tag{6}$$

[2] $supp(x|P_{\lceil\frac{|S|}{n}\rceil})+supp(\tilde{x}|P_{\lceil\frac{|S|}{n}\rceil})$ may less than n while incomplete partition happened in the last segment.

Accordingly, the **distribution** of x in P_j, i.e., $p(P_j)$, and the **distribution** of x in C_τ, i.e., $q(C_\tau)$, are defined as following respectively,

$$p(P_j) = \frac{supp(x|P_j)}{supp(x)}, \quad \text{and} \quad q(C_\tau) = \frac{supp(x|C_\tau)}{supp(x)}. \tag{7}$$

3.3 Distribution Periodicity and Structure Periodicity

An event x is said to have **distribution periodicity** in S with respect to "good" partition $\pi(n)$, if its *support* (appearance) in each segment is equal. For the distribution periodicity, we have the following theorem:

Theorem 1. *If event x has a distribution period of n in S, then the ideal distribution of x in $\lceil \frac{|S|}{n} \rceil$ segments is as*

$$p_n = \left\{ \frac{1}{\lceil \frac{|S|}{n} \rceil}, ..., \frac{1}{\lceil \frac{|S|}{n} \rceil}, ..., \frac{1}{\lceil \frac{|S|}{n} \rceil} \right\}. \tag{8}$$

Proof. The good partition $\pi(n) = \{P_1|P_2|\cdots|P_{\lceil \frac{|S|}{n} \rceil}\}$ divides S into $\lceil \frac{|S|}{n} \rceil$ equal length segments. Since x shows distribution periodicity with respect to partition $\pi(n)$, we can expect that $supp(x|P_i) \approx supp(x|P_j)$ for any $i \neq j$, where $i, j = \{1, .., \lceil \frac{|S|}{n} \rceil\}$. Moreover, with Lemma 1, we obtain:

$$supp(x) = \sum_{j=1}^{\lceil \frac{|S|}{n} \rceil} supp(x|P_j) \approx \lceil \frac{|S|}{n} \rceil \times supp(x|P_j).$$

That is, the distributions of x in $\{P_j\}$, i.e., $\frac{supp(x|P_j)}{supp(x)}$, are equally to $1/\lceil \frac{|S|}{n} \rceil$.

Definition 4. *An event x is said to have* **structure periodicity** *in S with respect to "good" partition $\pi(n)$, if its position (time point) in each segment is the same.*

For the structure periodicity, we have the following theorem:

Theorem 2. *If event x has a structure period n in S at position $\tau^{\#} \in [1, ..., n]$, then the ideal distribution of x on the n positions is as*

$$q_n = \{q(C_1), ..., q(C_{\tau^{\#}}), ..., q(C_n)\} = \{0, ..., 1, ..., 0\}. \tag{9}$$

Proof. If event x has structure periodicity in data series BS_x with respect to "good" partition $\pi(n)$, then

- $b_{i*n+\tau} = b_{j*n+\tau}$, here $i, j \in [0, \cdots, \lceil \frac{|S|}{n} \rceil - 1]$ and $\tau \in [1, ..., n]$; and
- $\exists\ \tau^{\#} \in [1, ..., n]$ such that $b_{\tau^{\#}} = 1$ and $b_\tau = 0$ $(\tau \neq \tau^{\#})$.

We obtain $\forall\ j \in [0, \cdots, \lceil \frac{|S|}{n} \rceil - 1]$, $b_{j*n+\tau^{\#}} = 1$ and $b_{j*n+\tau} = 0$ holds. Then,

$$supp(x|C_{\tau^{\#}}) = \sum_{j=0}^{\lceil \frac{|S|}{n} \rceil - 1} b_{j*n+\tau^{\#}} = \lceil \frac{|S|}{n} \rceil,$$

and $supp(x|C_{\tau})_{\tau \neq \tau^{\#}} = 0$. Based on Lemma 2, we know that $q(C_{\tau^{\#}}) = 1$, and $\{q(C_{\tau})\}_{\tau \neq \tau^{\#}}$ are all 0.

If $\pi(n)$ is the "good" partition for the distribution periodicity of x in S, and $\pi(n)$ is also the "good" partition for the structure periodicity of x in S, then x is said to have a **perfect periodicity** in S.

3.4 Research Problem

Implementing partition method $\pi(n) \in \Pi$ on a binary time-series BS_x, it would generate two distributions for the appearances of x in data series, i.e.,

$$\hat{p}_n = \{\hat{p}(P_j)\}_{1 \leq j \leq \lceil \frac{|S|}{n} \rceil]} \text{ and } \hat{q}_n = \{\hat{q}(C_{\tau})\}_{1 \leq \tau \leq n}. \tag{10}$$

If there exists a feasible measurement of $d(\cdot)$ that can be used to evaluate the distance between two distributions in (10), then $d(\hat{p}_n, p_n)$ and $d(\hat{q}_n, q_n)$ would show how close a real probability distribution $\hat{p}_n$ ($\hat{q}_n$) is to a candidate distribution of p_n (q_n). Without losing generality, it can be assumed that the more closer $\hat{p}_n$ ($\hat{q}_n$) to p_n (q_n), the more smaller the value of $d(\hat{p}_n, p_n)$ and $d(\hat{q}_n, q_n)$ would be. Along this line, the event periodicity detection is changed to find an optimal partition $\pi(n)$ on S to minimize the distance between the generated two distributions with two distributions respectively:

$$\begin{array}{l} \min_{\pi(n) \in \prod} \{d(\hat{p}_n, p_n)\} \text{ and } \min_{\pi(n) \in \prod} \{d(\hat{q}_n, q_n)\} \\ \text{st. } 2 \leqslant n \leqslant \lceil \frac{|S|}{2} \rceil. \end{array} \tag{11}$$

4 Mining Event Periodicity

4.1 Cross Entropy

In this work, we introduce the cross entropy [15] to measure the similarity between two distributions. Given two distributions of $\hat{p}_n$ and p_n, the **cross entropy** or the Kullback-Leibler (KL) divergence between $\hat{p}_n$ and p_n is defined by

$$KL(\hat{p}||p)_n = \sum_n \hat{p}_n \log \frac{\hat{p}_n}{p_n}. \tag{12}$$

The cross entropy determines the ability to discriminate between two states of the world, yielding sample distributions $\hat{p}_n$ and ideal distribution p_n.

Theorem 3. *$KL(\hat{p}||p)_n \geq 0$, and it is minimized if the distributions match exactly, i.e., $KL(\hat{p}||p)_n = 0$ if $\hat{p}_n = p_n$.*

Theorem 3 provides theoretical clues for finding a feasible n in task of event periodicity detection.

4.2 Identifying Distribution Periodicity

According to the definition of Theorem 1, if event x has distribution periodicity in S with respect to the **"good" partition** $\pi(n)$, then

$$p_n = \left\{ \frac{1}{\lceil \frac{|S|}{n} \rceil}, \cdots, \frac{1}{\lceil \frac{|S|}{n} \rceil} \right\}.$$

The appearances of x in each segment P_j is $supp(x|P_j)$ with respect to $\pi(n)$. Accordingly, the posterior probability distribution of $\hat{p}_n$ is calculated as:

$$\hat{p}_n = \left\{ \frac{supp(x|P_1)}{supp(x)}, \cdots, \frac{supp(x|P_{\lceil \frac{|S|}{n} \rceil})}{supp(x)} \right\}.$$

Known from Theorem 3, a smaller value of $KL(\hat{p}||p)_n$ means the posterior distribution p_n is more close to q_n, which indicates that $p_n \sim q_n$ means $KL(\hat{p}||p)_n \sim 0$ and then $\pi(n)$ may be a "good" partition for detecting the periodicity of event x. Thus, the task of detecting distribution periodicity of event x in S is equal to find a "good" partition $\pi(n^*)$ to minimizes the KL distance:

$$n^* = \arg \min_{\pi(n) \in \prod} \{KL(\hat{p}||p)_n\} \quad \text{st. } 2 \leqslant n \leqslant \lceil \frac{|S|}{2} \rceil. \tag{13}$$

We propose an Algorithm 1 to calculate the minimized $KL(\hat{p}||p)_n$.

Algorithm 1. Calculate $KL(\hat{p}||p)_n$

```
1:  Input: Binary data series $BS_x$;
2:  Output: $\mathbb{KL}$;
3:  $\mathbb{KL} = \phi$;
4:  for $n = 2$ to $\lceil \frac{|S|}{2} \rceil$ do
5:      $KL_n = 0$;
6:      for $j = 1$ to $\lceil \frac{|S|}{n} \rceil$ do
7:          $P_j = \{b_{(j-1)*n+1}, ..., b_{j*n}\}$;
8:          $p_j = \frac{supp(x|P_j)}{supp(x)}$;
9:          $KL_n = KL_n + p_j \log \left(p_j * \lceil \frac{|S|}{n} \rceil\right)$;
10:     end for
11:     $\mathbb{KL} \leftarrow KL_n$;
12: end for
13: return $\mathbb{KL}$;
```

The proposed method traverses all the n in $[2, \lceil \frac{|S|}{2} \rceil]$ to find the most feasible $\pi(n)$ such that the value of $KL(\hat{p}||p)_n$ can be minimized. Based on the partition results provided by $\pi(n)$, we have to calculate $\lceil \frac{|S|}{n} \rceil$ values, i.e., $supp(x|P_j)_{j=\{1,\ldots,\lceil \frac{|S|}{n} \rceil\}}$, which can be obtained in $O(1)$ time. Therefore, the overall complexity of Algorithm 1 is very efficient of $\sum_2^{\lceil \frac{|S|}{2} \rceil} \lceil \frac{|S|}{n} \rceil = O(|S| \ln |S|)$.

4.3 Identifying Structure Periodicity

We consider the opposite side of the above mentioned "good" partition, that is, the "worst" partition for showing the structure periodicity of event x. In such a poor case, the distribution of x would not obey the rule of Theorem 2, which means the distributions of x in $C_t, (t = 1, ..., n)$ are the same instead of a distribution shown in relation (9). Such a distributions of x can be referred as:

$$q_n = \left\{ \frac{1}{n}, \cdots, \frac{1}{n} \right\}.$$

In real, the posterior probability distribution of x in C_τ is:

$$\hat{q}_n = \left\{ \frac{supp(x|C_1)}{supp(x)}, \cdots, \frac{supp(x|C_n)}{supp(x)} \right\}.$$

Using $KL(\hat{q}||q)_n$ to measure the difference between two distributions of $\hat{q}_n$ and q_n, a bigger value of $KL(\hat{q}||q)_n$ indicates that the posterior distribution $\hat{q}_n$ is deviated much from the route of q_n, and thus $\pi(n)$ may be a "good" partition for detecting the structure periodicity of event x. Detecting structure periodicity of event x is thus equal to find a $\pi(n) \in \Pi$ to minimizes the value of $-KL(\hat{q}||q)_n$.

$$n^{\#} = \arg \min_{\pi(n) \in \prod} \{-KL(\hat{q}||q)_n\} \quad \text{st. } 2 \leqslant n \leqslant \lceil \frac{|S|}{2} \rceil. \tag{14}$$

Algorithm 2 is used to calculate all the value of $-KL(\hat{q}||q)_n$ for x.

Algorithm 2. Compute $-KL(\hat{q}||q)$

```
1: Input: Binary data series $BS_x$;
2: Output: $\mathbb{KL}$;
3: $\mathbb{KL} = \phi$;
4: for $n = 2$ to $\lceil \frac{|S|}{2} \rceil$ do
5:     $KL_n = 0$;
6:     for $k = 1$ to $n$ do
7:         $C_k = \{b_k, b_{n+k}, ..., b_{(\lceil \frac{|S|}{n} \rceil - 1)*n+k}\}$;
8:         $q_k = \frac{supp(x|C_k)}{supp(x)}$;
9:         $KL_n = KL_n + q_k * \log(q_k * n)$;
10:    end for
11:    $\mathbb{KL} \leftarrow -KL_n$;
12: end for
13: return $\mathbb{KL}$;
```

There are totally $supp(x)$ appearances of x in S and $\lceil \frac{|S|}{2} \rceil$ partition results, we then can calculate $supp(x|C_\tau)$ for all the partition results by traversing all the appearance of x in S. Along this way, the complexity of Algorithm 2 is $O(supp(x)|S|)$ and the operation time can be optimized, especially, when x is sparsely distributed in S. Note that, the basic operation for this method is to calculate $supp(x|C_\tau)$ with (5) for all the $\lceil \frac{|S|}{2} \rceil$ partition results. Therefore, the complexity of Algorithm 2 is characterized by $\max\{O(|S| \ln |S|), O(supp(x)|S|)\}$.

5 Experimental Results

5.1 Experimental Setup

We conduct a series of experiments to evaluate the performance of the proposed method, namely Cross-Entropy based Periodicity Detection (CEPD). To that end, three algorithms of WARP [8], CONV [7] and VAR [30] are selected for comparisive purpose. Given that the performance of VAR are affected heavily by a user-specified parameter of va, i.e., a bigger threshold value of va will result in an increased time consumption [30], the VAR method will be conducted 3 times with different parameter settings of $va = 0.01$, 0.1, and 0.2 respectively.

All these algorithms suffer from the poor performance to big data and poor resilience to noise. To support this claim and make the experimental results more clear, we conduct a series of experiments using synthetic data. The synthetic data have been generated by controlling parameters of data distribution (uniform or normal), alphabet size (number of unique symbols in the data), size of the data series (total number of symbols), period length, and the amount of noise in the data, which is the same way as done in [7,19]. In addition, each algorithm will be run 10 times, and then take the averaged value of these experimental results as the final result to avoid potential bias.

The confidence of a periodic event x occurring in data series S is the ratio of its actual periodicity to its expected periodicity. Formally, the periodicity confidence of x in S under partition $\pi(n)$ is defined as [13,19]:

$$conf(x)_n = \frac{Actual\_Periodicity(x)}{\lceil \frac{|S|}{n} \rceil}, \tag{15}$$

where $Actual\_Periodicity(x)$ is computed by counting the number of segments in which x is appearanced.

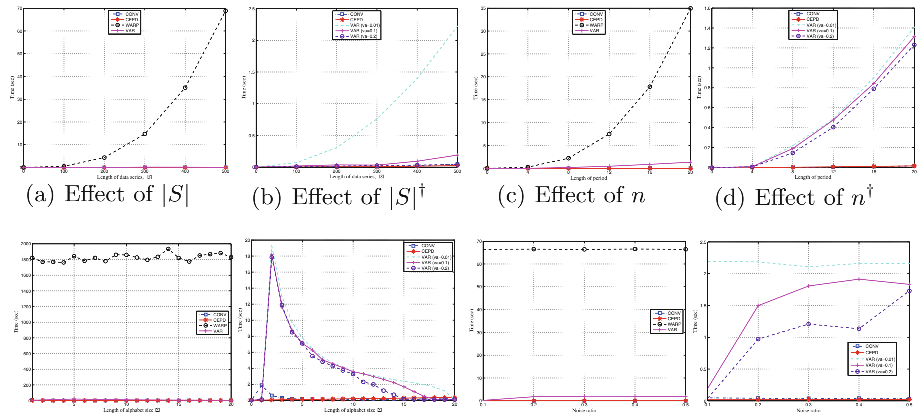
5.2 Efficiency of the Method

In the efficiency experiments, we test the time consumption of the four algorithms of CEPD, WARP, CONV and VAR under the impacts of following circumstance: the total data size $|S|$, the period length n, the alphabet size $|\Sigma_S|$ and the (replacement) noise ratio in S.

The first set of experiments is about the effect of data series size, $|S|$, on the efficiency of algorithms, all the methods will be conducted on a set of synthetic data series by varying data size from 100 to 500. These synthetic data series have been generated by following uniform distribution with alphabet size of 4 and embedded identical period of 5. The results are presented in Fig. 1(a) and (b). Obviously, the running time of all the four methods will go increasing dramatically while $|S|$ becomes bigger. WARP has the highest complexity (Fig. 1(a))[3]. Figure 1(b) is a locally magnified image of Fig. 1(a), which shows that both the efficiency of CEPD and CONV are better than that of VAR,

[3] In Fig. 1, symbol † means the experimental results without WARP.

this advantage becomes more obvious with the decrease of va. The experiments indicate that the efficiency of CEPD is superior to the others when data series becomes very large.



(a) Effect of $|S|$ (b) Effect of $|S|^\dagger$ (c) Effect of n (d) Effect of $n^\dagger$

(e) Effect of $|\Sigma_S|$ (f) Effect of $|\Sigma_S|^\dagger$ (g) Effect of noise (h) Effect of noise†

Fig. 1. Efficiency experiments

The next set of experiments are intended to show the efficiency of the algorithms by varying the embedded period size. In the experiment, we fixed the number of alphabets as 4, and embedded period are varied from 4 to 20. The curves of running time taken by CEPD, WARP, CONV and VAR have been plotted in Fig. 1(c), and an amplification version for the comaprison among CEPD, CONV and VAR is shown in Fig. 1(d). The results show that the time consumption of all the four algorithms are increased while the period length becomes longer. Again, WARP has the highest complexity which is followed by VAR, CONV and then CEPD.

The third set of experiments are intended to show the performances of all the four methods under effects of different alphabet size $|\Sigma|$ in a data series. The synthetic data series used in the experiments are embedded with a period of 32 and the number of alphabets are varied from 1 to 20 (smaller than the period value). The experimental results show that, along with the increasing of the alphabet size $|\Sigma|$, the running time of WARP is stable and significantly higher than the other methods. Interestingly, the running time of VAR and CONV increase dramatically when $|\Sigma|$ is relative small, and then fall down when $|\Sigma|$ goes bigger (Fig. 1(e) and (f)). The running time of CEPD increases slowly and lower than that of the other methods.

The fourth experiments are conducted to measure the impact of noise ratio on the time performance of the four methods. To that end, we fixed the length of the time series as 500, and replaced some regular symbols with noise symbols in the experiments. The noise ratio is varied from 0 to 0.5. As we can see, the running efficiency of CEPD, WARP and CONV perform stably under different noise ratio. In other words, the performance of these algorithms are not sensitive to replacement noise (Fig. 1(g) and (h)). This is similar to the results presented in [19]. As for VAR, it performs worse as the ratio noise increasing (Fig. 1(h)).

5.3 Accuracy of the Method

Two different ways are conducted to study the accuracy of the algorithms. The first compares the value of confidence for each method assuming that the period can be identified by all the methods. Whereas, the second compares the period identified by each method when the confidence is maximized. Five synthetic data sets are generated for the experiments (Table 1). In the experiments, the potential period length is set as 10, and the data series is generated by repeating the period 100 times.

Table 1. The generation of synthetic data set (denoted by D).

D	Generation rules	Sample data series
1	1 periodic event	$\underline{1}000$ $\underline{1}000$ $\underline{1}000$ $\underline{1}000$
2	1 random event	1000 0010 0001 0100
3	2 periodic events	$\underline{1}0\underline{1}0$ $\underline{1}0\underline{1}0$ $\underline{1}0\underline{1}0$ $\underline{1}0\underline{1}0$
4	1 periodic and 1 random event	$\underline{1}001$ $\underline{1}100$ $\underline{1}010$ $\underline{1}010$
5	2 random events	0101 1001 1010 0011

For the experiments of confidence comparison, the results are listed in Table 2. As we can see, the CEPD can identify all the events with confidence 100% since all the events in the data series are uniformed distributed with respect to the period. WARP also show a better performance on accuracy (close to 100%) than CONV and VAR. However, CONV performs good on data set 1, 3 and 4, and bad on data set 2 and 5 (the events are randomly distributed). That is to say, CONV prefers to structure period. The VARs show good efficiency on the data series having only 1 event embedded (data set 1 and 2).

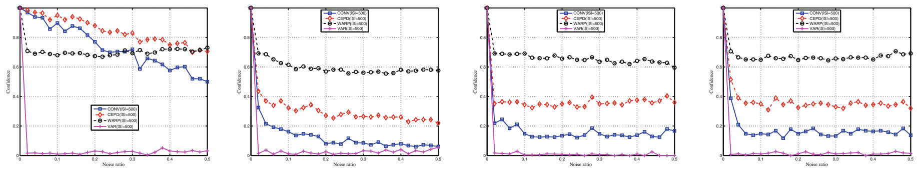
5.4 Noise Resilience

In data series, there are three types of noise: replacement, insertion, and deletion noise [8]. Accordingly, the purpose of the following experiments are to study the behavior of the different methods in periodicity detection with respect to toward these noise as well as some mixtures of them. In the experiments, we used a synthetic time series containing 4 symbols and period size of 10. The noise ratio increased gradually from 0.0 to 0.5. Finally, we report the averaged confidence level of all the symbols at which the actual period of 10 is detected.

Table 2. The comparison of confidence (the period is fixed).

Data Set	CEPD	CONV	WARP	VAR ($va = 0.2$)	VAR ($va = 0.1$)	VAR ($va = 0.01$)
1	1	1	1	1	1	1
2	1	0	0.9959	1	1	-
3	1	1	1	-	-	-
4	1	1	0.9989	-	-	-
5	1	0.0813	0.9975	-	-	-

In general, along with the increase of noise ratio, the accuracy of all the algorithms are in decline; especially when considering insertion, deletion or hybrid noise increasing, the accuracy of these algorithms will fall shapely (Figs. 2 and 3). In case that the noises are uniformly distributed in data series, CEPD has the best performance for the replacement noise (Fig. 2(a)). When the insertion, deletion and hybrid noises are embedded uniformly into a data series, WARP shows the best performance on noise resilience, followed by CEPD (Fig. 2(b), (c) and (d)). In case that the noises are normally distributed in data series, similarly, CEPD performances best under the situation that the replacement noises are embedded (Fig. 3(a)). However, for the situations of insertion, deletion and hybrid noises, the comparative results are mixed and no method has a significant advantage over the others in noise resilience (Fig. 3(b), (c) and (d)).



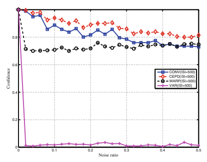
(a) Replace noise. (b) Insertion noise. (c) Deletion noise. (d) RID noise.

Fig. 2. Accuracy (uniformly distributed noise).

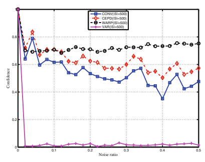
5.5 A Case Study on Real Dataset

A real-world data set, i.e., Amazon access samples data set (AASDS)[4], has been used in the experiments which was created and donated by Amazon.com in 2011 and has been cited for many times. AASDS contains 17612 users' access history from 2005.8 to 2010.8. To study the periodicity of each Amazon user, we take "Day" as the basic time unit, and all the accessing actions are then counted by 24-hours-day, for example, if a user had accessed Amazon.com more than 0

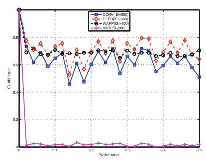
[4] http://archive.ics.uci.edu/ml/datasets/.



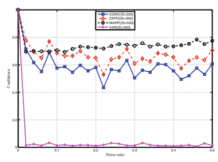
(a) Replace noise.



(b) Insertion noise.



(c) Deletion noise.

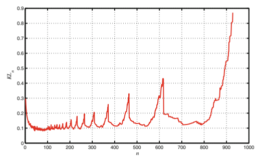


(d) RID noise.

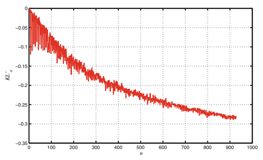
Fig. 3. Accuracy (normally distributed noise).

times in Sep. 2, 2005, then we marked the value of e_t at the position of day Sep. 2, 2005 as "1" in S_A, otherwise, "0" is marked. Finally, we can generate a data series of S_A for the targeted user.

Taking the No. #33400 user as an example, the results are shown in Fig. 4. As we can see that, the minimized value of KL shows that the distribution period of users #33400 accessing Amazon.com can be approximated as $n^* = 15$ days (two weeks). That is to say, the user trended to visit Amazon.com equal times every 15 days. Although the event of user #33400 accessing Amazon.com has a distribution of two weeks, but the multiple relationship between the positions of local minimized $-KL'$ for data set AASDS is weak, which means the user has no regular time point for visiting Amazon.com.



(a) KL of S_A.



(b) $-KL'$ of S_A.

Fig. 4. KL and $-KL'$ in real data series.

6 Conclusion

Discovering the periodicity of event happened in sequential data series is a valuable work for data analyzing. In this paper, a novel and efficient method, namely CEPD is proposed to address the event-based periodicity mining problem in data series. The advantages of the proposed method are summarized as follows: first, it is the first time to distinguish the idea of distribution periodicity and structure periodicity. Second, a simple and complete partition method $\pi(n)$ is proposed. Third, basing on the minimum cross entropy principle and the property of periodic function, we present an efficient method to measure and determine the periodicity of an event. The experimental results show that CEPD has a best performance of running efficiency due to its less complexity.

Acknowledgments. The authors would like to thank the supports of the National Natural Science Foundation of China (71671027/91846105/71572029/71490723).

References

1. Benson, A.R., Kumar, R., Tomkins, A.: Modeling user consumption sequences. In: Proceedings of the 25th International Conference on World Wide Web, WWW 2016, pp. 519–529 (2016)
2. Berberidis, C., Vlahavas, I., Aref, W.G., Atallah, M., Elmagarmid, A.K.: On the discovery of weak periodicities in large time series. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) PKDD 2002. LNCS (LNAI), vol. 2431, pp. 51–61. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45681-3_5
3. Brigham, E.: Fast Fourier Transform and Its Applications, 1st edn. Prentice Hall, Englewood (1988)
4. Cao, H., Cheung, D.W., Mamoulis, N.: Discovering partial periodic patterns in discrete data sequences. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 653–658. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24775-3_77
5. Cetintas, S., Chen, D., Si, L., Shen, B., Datbayev, Z.: Forecasting counts of user visits for online display advertising with probabilistic latent class models. In: Proceeding of the 34th International ACM SIGIR Conference, pp. 1217–1218 (2011)
6. Chan, K.P., Fu, A.W.C.: Efficient time series matching by wavelets. In: Proceedings of the 15th International Conference on Data Engineering, ICDE 1999, pp. 126–133 (1999)
7. Elfeky, M.G., Aref, W.G., Elmagarmid, A.K.: Periodicity detection in time series databases. IEEE Trans. Knowl. Data Eng. **17**(7), 875–887 (2005)
8. Elfeky, M.G., Aref, W.G., Elmagarmid, A.K.: WARP: time warping for periodicity detection. In: Proceedings of the Fifth IEEE International Conference on Data Mining, ICDM 2005, pp. 138–145 (2005)
9. Elfeky, M.G., Aref, W.G., Elmagarmid, A.K.: Stagger: periodicity mining of data streams using expanding sliding windows. In: Proceedings of the 6th IEEE International Conference on Data Mining, pp. 188–199 (2006)
10. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. In: Proceedings of the SIGMOD 1994, pp. 419–429. ACM (1994)
11. Ghosh, A., Lucas, C., Sarkar, R.: Finding periodic discrete events in noisy streams. Proc. CIKM **2017**, 627–636 (2017)
12. Glynn, E.F., Chen, J., Mushegian, A.R.: Detecting periodic patterns in unevenly spaced gene expression time series using lomb-scargle periodograms. Bioinformatics **22**(3), 310–316 (2006)
13. Han, J., Dong, G., Yin, Y.: Efficient mining of partial periodic patterns in time series database. In: Proceedings of International Conference on Data Engineering, pp. 106–115 (1999)
14. He, Z., Wang, X.S., Lee, B.S., Ling, A.C.H.: Mining partial periodic correlations in time series. Knowl. Inf. Syst. **15**, 31–54 (2008)
15. Kullback, S., Leibler, R.A.: On information and sufficienvy. Ann. Math. Stat. **22**, 79–86 (1951)
16. Li, Z., Ding, B., Han, J., Nye, R.K.P.: Mining periodic behaviors for moving objects. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1099–1108 (2010)

17. Li, Z., Wang, J., Han, J.: Mining event periodicity from incomplete observations. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 444–452 (2012)
18. Ma, S., Hellerstein, J.L.: Mining partially periodic event patterns with unknown periods. In: Proceedings of the 17th International Conference on Data Engineering, pp. 205–214. IEEE (2001)
19. Rasheed, F., Alshalalfa, M., Alhajj, R.: Efficient periodicity mining in time series databases using suffix trees. IEEE Trans. Knowl. Data Eng. **23**(1), 79–94 (2011)
20. Ruiz, E.J., Hristidis, V., Castillo, C., Gionis, A., Jaimes, A.: Correlating financial time series with micro-blogging activity. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, WSDM 2012, pp. 513–522. ACM (2012)
21. Sheng, C., Hsu, W., Lee, M.L.: Mining dense periodic patterns in time series data. In: Proceedings of the 22nd International Conference on Data Engineering, ICDE 2006, p. 115. IEEE (2006)
22. Sripada, S.G., Reiter, E., Hunter, J., Yu, J.: Segmenting time series for weather forecasting. In: Macintosh, A., Ellis, R., Coenen, F. (eds.) Applications and Innovations in Intelligent Systems X, pp. 193–206. Springer, London (2003). https://doi.org/10.1007/978-1-4471-0649-4_14
23. Tanbeer, S.K., Ahmed, C.F., Jeong, B.-S., Lee, Y.-K.: Discovering periodic-frequent patterns in transactional databases. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS (LNAI), vol. 5476, pp. 242–253. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01307-2_24
24. Vlachos, M., Yu, P.S., Castelli, V.: On periodicity detection and structural periodic similarity. In: SDM 2005, pp. 449–460 (2005)
25. Wang, R.Y., Storey, V.C., Firth, C.P.: A framework for analysis of data quality research. IEEE Trans. Knowl. Data Eng. **7**(4), 623–640 (1995)
26. Wang, X., Zhang, H., Zhang, D., Xiao, Y.: A moving-window based partial periodic patterns update technology in time series databases. In: 2008 International Symposium on Computational Intelligence and Design, ISCID 2008, vol. 2, pp. 98–101, October 2008
27. Yang, J., Wang, W., Yu, P.S.: Infominer: mining surprising periodic patterns. In: Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2001, pp. 395–400. ACM (2001)
28. Yang, J., Wang, W., Yu, P.S.: Mining asynchronous periodic patterns in time series data. IEEE Trans. Knowl. Data Eng. **15**(3), 613–628 (2003)
29. Yang, Y., Pan, B., Song, H.: Predicting hotel demand using destination marketing organization's web traffic data. J. Travel Res. **53**(4), 433–447 (2014)
30. Yang, Y.C., Padmanabhan, B., Liu, H., Wang, X.: Discovery of periodic patterns in sequence data: a variance-based approach. INFORMS J. Comput. **24**(3), 372–386 (2012)
31. Yuan, Q., Shang, J., Cao, X., Zhang, C., Geng, X., Han, J.: Detecting multiple periods and periodic patterns in event time sequences. Proc. CIKM **2017**, 617–626 (2017)
32. Ziegler, H., Jenny, M., Gruse, T., Keim, D.A.: Visual market sector analysis for financial time series data. In: IEEE VAST, pp. 83–90. IEEE (2010)