Knowledge-Based Systems 54 (2013) 243–254

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Finding time series discord based on bit representation clustering

Guiling Li [a], Olli Bräysy [b,c], Liangxiao Jiang [a], Zongda Wu [d,*], Yuanzhen Wang [e]

[a] School of Computer Science, China University of Geosciences, Wuhan 430074, China
[b] VU University Amsterdam, De Boelelaan 1081, 1081 HV Amsterdam, Netherlands
[c] Procomp Solutions Oy, Kiviharjuntie 11, FI-90220 Oulu, Finland
[d] Oujiang College, Wenzhou University, Wenzhou 325035, Zhejiang, China
[e] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China

ARTICLE INFO

Article history:
Received 31 January 2013
Received in revised form 9 September 2013
Accepted 12 September 2013
Available online 28 September 2013

Keywords:
Time series data mining
Discord discovery
Clustering
Pruning
Dimensionality reduction

ABSTRACT

The problem of finding time series discord has attracted much attention recently due to its numerous applications and several algorithms have been suggested. However, most of them suffer from high computation cost and cannot satisfy the requirement of real applications. In this paper, we propose a novel discord discovery algorithm *BitClusterDiscord* which is based on bit representation clustering. Firstly, we use PAA (Piecewise Aggregate Approximation) bit serialization to segment time series, so as to capture the main variation characteristic of time series and avoid the influence of noise. Secondly, we present an improved K-Medoids clustering algorithm to merge several patterns with similar variation behaviors into a common cluster. Finally, based on bit representation clustering, we design two pruning strategies and propose an effective algorithm for time series discord discovery. Extensive experiments have demonstrated that the proposed approach can not only effectively find discord of time series, but also greatly improve the computational efficiency.

1. Introduction

Time series discord is the subsequence of a time series, which has the biggest difference in all the subsequences of the time series [1]. Recently, finding time series discord has attracted much attention due to its numerous applications [1–7]. Let us take a specific application example from the health care sector. Both Electrocardiogram (ECG) and Electroencephalogram (EEG) can be deemed as time series [8–14]. It is an important routine for doctors to analyze whether there is a discord in a patient's ECG or EEG so as to determine the patient's health status. There are also numerous other applications. Time series discord discovery is e.g. very valuable in data mining tasks such as anomaly detection, improving quality of clustering and data cleansing [1].

Several algorithms on time series discord discovery have already been published. Naïve method [1] considers each subsequence as discord candidate, and uses two layer loops to detect whether the candidate is discord. The complexity of naïve method is $O(n^2)$. Simple pruning method [1] adopts pruning strategy based on naïve method, providing some improvements but far less than enough. The efficiency of these methods is not good enough for real applications. For example, in the health care sector it is very important for the doctor to rapidly diagnose the patient's symptoms according to ECG checklists and start the immediate treatment when necessary.

As time series often has the characteristic of high dimensionality, we choose to reduce the dimensionality of raw time series through approximate bit representation. Clustering is another strategy to accelerate discord discovery through filtering the candidates effectively, we apply clustering based on the bit representation. According to the obtained clustering results, we notice that a pruning process is also required to reduce the required computing time. Thus, in this paper we propose a novel pruning method for clustering.

The main contributions of this paper are highlighted as follows:

- *Bit representation clustering*: We propose an improved K-Medoids clustering algorithm based on bit representation for time series, called as *BitCluster*. It provides a primary filtering and clearly accelerates the discord discovery.
- *Pruning based discord discovery*: We propose a discord discovery algorithm after the bit representation clustering, called as *BitClusterDiscord*. In order to improve efficiency, we design two pruning strategies. The first is a heuristic pruning whereas the second is based on the cluster center distance.
- *Experimental study*: We conduct extensive experiments on diverse datasets to test the proposed algorithm from multiple aspects. The results show that our algorithm finds discords effectively, scales well and is more effective than the previous methods.

* Corresponding author.
E-mail addresses: guiling@cug.edu.cn (G. Li), olli.braysy@pp.inet.fi (O. Bräysy), ljiang@cug.edu.cn (L. Jiang), zongda1983@163.com (Z. Wu), wangyz2005@163.com (Y. Wang).

0950-7051/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2013.09.015

The rest of the paper is organized as follows. In Section 2 we discuss the previous work on time series discord discovery. In Section 3 we give the problem statement, and in Section 4 we describe the bit representation for time series. In Section 5 we present the improved clustering algorithm based on bit representation. We propose the novel discord discovery algorithm in Section 6 and in Section 7 we show the experimental results and evaluations. Finally, in Section 8 we conclude our work and point out the future work.

2. Related work

Existing time series discord discovery algorithms can be categorized into three types: naïve method, simple pruning method and method based on dimensionality reduction techniques.

The main of idea of naïve method is to define the subsequence with the largest distance to its nearest neighbor as discord. This is done by considering each subsequence as candidate and finding first out the nearest non-self match. The method can be implemented by a two layer nested loop. The outer loop considers each possible candidate subsequence and the inner loop is a linear scan to identify the nearest non-self match of the candidate. This method is simple, easy to implement, and can produce the exact solution. The main problem is that the complexity is $O(n^2)$.

Simple pruning method [1] makes some improvement over the naïve method by applying a random sampling method to prune the candidates. More precisely, in the simple pruning method the above mentioned inner loop is checked through in a random order and the loop may be terminated before the end is reached if certain conditions are met. Here the key idea is that if the distance of certain subsequence to its non-self match is smaller than the smallest distance found until now, even if the non-self match is not the nearest, it can be confirmed that the subsequence is not the discord.

The above two methods are mainly designed for raw time series. By applying time series dimensionality reduction techniques, discord can be found also based on the approximate representation. Keogh et al. [1] presented the first study on the discord problem and proposed a heuristic method called HOT SAX. Their approach is based on SAX (Symbolic Aggregate approXimation), applying the heuristic pruning in two layer loops. In the outer loop, the subsequence with larger distance to its nearest neighbor has a priority to be selected for comparison. Correspondingly, in the inner loop, the subsequence with smaller distance to the current candidate has a priority to be compared. Thus, the method can be terminated sooner. However, SAX should meet the requirement of Gaussian distribution in data and the similarity measure of SAX is not accurate when comparing the adjacent symbols in the search table.

Li et al. [4] applied SAX in image domain and studied shape discord discovery. They map shape sketch to time series, apply SAX representation and then utilize heuristic strategy to find discords. Since the same shape can be converted to different time series with different rotation directions so as to different SAX representations, they handle this problem with rotation invariant Euclidean distance.

Fu et al. [2] found discords by Harr transform. They first make Harr transform for subsequences, and then utilize breadth first split algorithm to order the outer and inner loop. This method relies on the basis function of Harr transform. Bu et al. [3] studied top-k discord discovery in time series databases and proposed WAT algorithm based on Harr wavelet and augmented trie. They first employ Harr wavelet transform to approximate time series. Then, they discretize transformed sequences by symbols and reorder the candidate subsequences using heuristic method with augmented trie.

Most discord discovery algorithms assume that data resides in the memory. To solve disk aware discord discovery in huge date sets, Yankov et al. [5] proposed a two-phase discord detection algorithm. The first phase selects the candidates and the second phase identifies discords. The algorithm requires two linear scans over the disk.

Both symbolic and bit representation methods are used to discretize time series. Symbolic methods, such as SAX and *a*SAX, segment time series and discretize the subsequences into symbols. Bit methods such as PAA (Piecewise Aggregate Approximation) bit serialization [15,16], RLE (Run Length Encoding) [17] and BCM (Bit Coding Mode) [18] segment time series and discretize the subsequences into bit string. The merit of discretization is its simplicity and low storage cost. The effect of feature preserving depends on the concrete discretization method. Bit representation of time series is simple and has low storage cost, and bit operations are simple and efficient. So, in this paper, we study discord discovery problem on the basis of bit representation for time series.

3. Problem statement

In order to describe the problem of discord discovery, we give the related definitions as follows [1].

Definition 1 (*Time Series*). Time Series is a data sequence, where data element arrives with time order, denoted by $T = t_1, \ldots, t_n$, here n is the length of T.

Definition 2 (*Subsequence*). Given a time series T with length n, the subsequence of T is the consecutive position sampling with length m(with $m \ll n$) in T, denoted by $C = t_p, \ldots, t_{p+m-1}$, where $1 \leqslant p \leqslant n - m + 1$.

Definition 3 (*Distance*). *Dist* is a function as:

$$Dist(C, M) \to R \tag{1}$$

wherein the input parameters are two subsequences C and M, the function returns a non-negative number R, called as the distance from M to C. *Dist* function must hold symmetry, i.e., $Dist(C, M) = Dist(M, C)$.

Definition 4 (*Non-self Match*). Given a time series T, including a subsequence C with length m starting from position p and a matching subsequence M starting from position q. If $|p - q| \geqslant m$, M is called as a non-self match to C at the distance $Dist(M, C)$.

Definition 5 (*Time Series Discord*). Given a time series T, containing a subsequence D with length m starting from position l. If D has the largest distance to its nearest non-self match, D is called as the discord of T. That is to say, ∀ subsequence C of T, non-self match M_C of C, non-self match M_D of D, the inequality $\min(Dist(D, M_D)) > \min(Dist(C, M_C))$ is satisfied.

Our work is based on the above definitions, focusing on finding discord in time series.

4. Bit representation

Our discord discovery algorithm is based on bit representation via dimensionality reduction. In this section, we introduce PAA bit serialization and related bit distances.