A Framework for Similarity Search of Time Series Cliques with Natural Relations

Bin Cui, Senior Member, IEEE, Zhe Zhao, and Wee Hyong Tok

Abstract—A Time Series Clique (TSC) consists of multiple time series which are related to each other by natural relations. The natural relations that are found between the time series depend on the application domains. For example, a TSC can consist of time series which are trajectories in video that have spatial relations. In conventional time series retrieval, such natural relations between the time series are not considered. In this paper, we formalize the problem of similarity search over a TSC database. We develop a novel framework for efficient similarity search on TSC data. The framework addresses the following issues. First, it provides a compact representation for TSC data. Second, it uses a multidimensional relation vector to capture the natural relations between the multiple time series in a TSC. Lastly, the framework defines a novel similarity measure that uses the compact representation and the relation vector. We conduct an extensive performance study, using both real-life and synthetic data sets. From the performance study, we show that our proposed framework is both effective and efficient for TSC retrieval.

Index Terms—Time series clique, natural relation, compact representation, similarity search.

1 INTRODUCTION

Time series databases are prevalent in multimedia, finance, health, and traffic monitoring applications. These applications provide data analysts with the ability to perform sophisticated analysis on the time series data. In order to effectively and efficiently analyze time series data, many techniques [1], [2], [3], [4], [5], [6] have been proposed. These techniques are for processing multiple, individual time series. For example, motion data are extracted from the video clips of a basketball game, and stored in a time series database. In the time series database, there are multiple, individual time series. Each of the time series can refer to the information for a player at different time points in the game. Applications can then be built to provide users with the ability to analyze the scenes in the video which relates to a user-specified query. These results (i.e., scenes) that are returned can be ranked based on the similarity to the query.

Existing techniques cannot handle complex time series queries which require the natural relations between the time series to be used during similarity matching. The natural relations are prevalent in many time series databases, where each time series is correlated with several other time series in the database. For example, in the time series database that we described earlier, multiple basketball players from Teams A and B appear in a scene. Each of the players in the game moves and performs a specific action. In one scene, a player from Team A passes a ball to a teammate. The teammate aims and shoots the ball into the basket. At the same time, players from Team B are actively trying to prevent the player from getting the ball and shooting the ball into the basket. The moving trajectories of all the players in the game are correlated to each other. The position and action of each player in the game is reflective of the current game situation. We can easily observe the natural relations that exist in the time series database. If we process each time series independently, we will not be able to make use of the inherent natural relations.

We refer to groups of time series with natural relations as Time Series Cliques (TSC), which can be defined as \( \{T_1, T_2, \ldots, T_n\} \), where \( T_i = \{(x_m, t_m) | l = 1, 2, \ldots, L_i\} \) refers to a single time series, \( n \) is the number of the time series in the TSC and can be different values in different TSCs. For each \( T_i \), \( x_m \in \mathbb{R}^k \) denotes an observed value at a certain time, where \( k \) represents the dimensionality of each element in the time series, \( t_m \in \mathbb{R}^1 \) describes a time tick, and \( L_i \) is the length of the \( i \)th time series. The patterns of each time series and their natural relations can be extracted from the tuples in \( \{T_1, T_2, \ldots, T_n\} \). In each TSC, the time series exhibit natural relations with each other. Efficient similarity retrieval on TSC data can be used in many real-world applications. An example of TSC is illustrated in Fig. 1. The TSC data consist of the motion feature vectors of multiple players in a hockey game, e.g., Fig. 1a shows the trajectories of five players in the court, and Fig. 1b represents time interval for each trajectory, respectively. Other real-world applications, where TSC data can be found, include: video, and audio clips, electrocardiogram (ECG) readings, readings from seismic stations, etc.

These examples motivate the need for effective and efficient TSC similarity search algorithms. Based on the definition of TSC, we formally define TSC similarity search as follows:
Given a query TSC object \( Q = \{q_1, q_2, \ldots, q_l\} \), where \( l \) is the number of the time series in \( Q \) and each time series \( q_i = \{(x_m, t_m)\mid m = 1, 2, \ldots, L_i\} \), and a TSC database \( D = \{TSC_1, TSC_2, \ldots, TSC_N\} \). Each TSC object \( TSC_i = \{T_{i1}, T_{i2}, \ldots, T_{iN}\} \) and in \( TSC_i \), \( T_{ij} = \{(x_m, t_m)\mid m = 1, 2, \ldots, n_{ij}\} \), where \( N_i \) is the number of the time series in the \( i \)th element of database \( D \) and \( n_{ij} \) is the length of the \( i \)th time series in \( TSC_i \). The similarity search on TSC is to find the most similar \( TSC_s \) from database \( D \) via a predefined function \( \text{dist}(Q, TSC_i) \), where

\[
s = \arg \min_{i=1,2,\ldots,N} \text{dist}(Q, TSC_i).
\]

A key difference between TSC and existing time series similarity search is that the former requires the use of natural relations between the time series. In contrast, if the data are treated as single time series, the natural relations between the multiple time series will be lost. The TSC similarity search problem is a novel fusion of time series and spatiotemporal similarity search. It provides a powerful generalization of existing time series and spatiotemporal retrieval problems.

In order to reduce the computational complexity of the TSC similarity search problem, and avoid pairwise comparison between all the time series in multiple TSCs, a suitable compact data structure, which captures the intrinsic properties of the TSC and the inherent natural relations need to be identified. In contrast, in the absence of such a data structure, a brute-force approach will need to compare the trajectories of one scene with the trajectories in other scenes. A TSC similarity search algorithm can effectively and efficiently make use of the compact data structure to identify similar patterns in multiple time series. During TSC similarity search, the algorithm must be able to take into account the natural relations between the group of time series in TSCs. Most importantly, the similarity measure must be generic, and easily used to capture natural relations for different application domains.

In this paper, we focus on the new problem of performing similarity search over TSC databases. In order to validate the requirements for TSC processing, we presented our preliminary study of TSC processing in [7]. In this paper, we extended [7] with an in-depth investigation and performance analysis of TSC processing using different algorithms. Specifically, this paper makes the following additional contributions: First, we provide a comprehensive analysis of related work. Second, we present an in-depth discussion of the issues and solutions for feature extraction, feature transformation, and similarity matching algorithms for TSC processing. Third, we conduct an extensive performance analysis, using both synthetic and real-life data sets.

The main contributions of the paper are:

- We propose two data structures for Time Series Clique processing, which merges the relation of time series with their shapes or patterns. First, we propose a relation vector for TSC (RT) which effectively describes the intrinsic natural relations that exist between the multiple time series in a TSC. Second, we propose a Compact Representation for TSC (CT), which captures the dominant information in a TSC.
- We propose algorithms that leverages RT and CT to effectively and efficiently compute the distance between two TSCs.
- We conduct extensive performance analysis of the proposed algorithms using both synthetic and real-life data sets. The empirical results show that the proposed approach is superior compared with other TSC processing systems.

The rest of this paper is organized as follows: Section 2 introduces the related work. In Section 3, we introduce a brute-force matching method for TSC similarity search, which performs exhaustive search over all the TSCs. In Section 4, we propose a novel algorithm for similarity search in TSC databases. In Section 5, we present a comprehensive performance analysis of the proposed algorithms. We conclude in Section 6.

## 2 Related Work

There is a long stream of research on time series databases. We classify the existing work into three categories, i.e., time series matching, pattern and correlation mining on multiple time series, and feature extraction and representation of multiple time series relations.

### 2.1 Time Series Matching

In existing similarity search over time series databases, the time series is transformed from its original form \( \{(X_m, t_m)\mid m = 1, 2, \ldots, L\} \) into a more compact representation.
The search algorithm leverages on two steps: dimensionality reduction [8], [9], [10], [1], [3], [11] and data representation in the transformed space.

Various dimensionality reduction techniques have been proposed for time series data transformation. These includes: Discrete Fourier Transform (DFT), Singular Value Decomposition (SVD) [12], [8], [13], Discrete Wavelet Transform (DWT) [9], [10], and Piecewise Aggregate Approximation (PAA) [8]. Another approach for dimensionality reduction is to make use of time series segmentation [1], [3], [11].

Besides dimensionality reduction, the choice of time series representation is also important. Two types of time series representations are commonly used: numeric and symbolic. One of the commonly used numeric representation is the real sequence [6]. Symbolic representation is generated by a symbol table that reflects each data vector of sequence into symbols [5]. The symbol table can be either predefined or built from data sets.

Trajectory data can be considered as a specific form of time series, which has been applied in moving object search or video retrieval fields [3], [4], [11], [14], [6]. In these works, the trajectories first are segmented by their control points [4], [11], [14], or inflection points [3], and then different representation methods, i.e., either a scalable numeric representation in [14], [6] or symbolic representations in [3], [4], [11], [5] have been adopted for similarity measure.

### 2.2 Time Series Mining

Multiple time series mining has also been recently explored. Existing multiple time series research have focused on pattern mining and finding correlation between multiple time series, over patterns and observed values from group of individual time series. Papadimitriou et al. [15] proposed the SPIRIT system. SPIRIT performs incremental Principal Component Analysis (PCA) over stream data, and delivers results in real time. SPIRIT discovers the hidden variables among n input streams and automatically determines the number of hidden variables that will be used. The observed values of the hidden variables present the general pattern of multiple input series according to their distributions and correlations. BRAID [16] addressed the problem of discovering lag correlations between data streams. BRAID focuses on a time and space efficient method for finding the earliest and highest peak in the cross-correlation functions between all pairs of streams.

Another closely related area of work is research on computing the group nearest neighbor query [17]. The query of Group KNN is a set of high-dimensional data points. The group nearest neighbor query returns a group of data points in the database which are similar to the set of query based on the patterns and relations. The data set used in the Group KNN problem consists of independent points, and does not consider whether natural relations exist between these points.

These existing approaches cannot be easily extended for the TSC similarity search problem because of the lack of a clearly defined similarity measure. Most importantly, existing approaches are unable to deal with the natural relations that exist between the multiple time series.

### 2.3 Representation of Time Series Relations

Several approaches for capturing natural relations among multiple time series have been proposed. Allen [18] defined 13 interval relations that exist between multiple time series. These include: before, overlaps, during, etc. These relations are used to describe the relative position of two intervals. Fleischmann et al. [19] deployed Allen’s descriptor to capture temporal information by representing the events in video data based on a lexicon of hierarchical patterns of movements. An improved interval relation description method for local temporal relations, i.e., Time Series Knowledge Representation (TSKR) was proposed by Mörchen and Ullsch in [20]. TSKR expresses temporal knowledge in time series data. In addition, the temporal relation, 3D Z-String [21] was used to represent moving objects’ spatiotemporal relations. In 3D Z-string, the objects in a video are projected onto the x, y, and time-axis to form three strings representing the relations and relative positions. The temporal overlapping and spatial position are well defined in 3D Z-String.

In summary, none of the above methods can address the challenges of effective and efficient TSC similarity search. The existing methods lack a compact, powerful, and measurable representation for the patterns and the natural relations that are inherent in the TSCs. In addition, it is hard to identify a generic relation descriptor that can be applied to various application domains.

### 3 Brute-Force TSC Matching

In this section, we first present a straightforward approach for solving the TSC matching problem, which exhaustively compares the time series in TSCs. In such a Brute-Force approach for TSC similarity matching, all the possible matches between two TSCs are identified. Once the possible matches are found, the similarity between all the time series in each matching result are computed. The maximum similarity (i.e., minimum distance) is then used to represent the similarity between two TSCs.

Let us revisit the example shown in Fig. 1. We can observe that there are two similar TSCs, i.e., TSC1 (Figs. 1a and 1b) and TSC2 (Figs. 1c and 1d). These TSCs are extracted from a computer simulation of a hockey ball game [22]. Specifically, Figs. 1a and 1c are the trajectories of players in the TSC1 and TSC2. Figs. 1b and 1d are the time intervals of respective trajectories in the TSCs.

To measure the similarity between the above TSCs, i.e., TSC1 and TSC2, the Brute-Force approach is to find all the corresponding players in TSC2 for all the players in the TSC1, which are the players whose trajectories and intervals in Figs. 1b and 1d are of the same color as the players’ in Figs. 1a and 1c. After finding the matched players, we can measure the similarity of each pair by distance function between trajectories, and calculate the sum of their distance as the value of dist(TSC1, TSC2).

The above approach can be easily generalized to find the matching result between the time series in TSC1 and TSC2 for different application domains. A distance function DIS(Ti, Tj), where Ti ∈ TSC1 and Tj ∈ TSC2, can be used for comparing the similarities between two
TSCs. The function $DIS(T_i, T_j)$ measures the similarity between two single time series. In the literature, various distance functions have been proposed. These range from the simple euclidean distance between two vectors to the state-of-art distance functions described in [6]. During the matching process, the sum of $DIS(T_i, T_j)$ for all the matched pairs is computed. Thereafter, the minimum distance is recorded as the distance $\text{dist}_{BF}$ between two TSCs $TSC_1$ and $TSC_2$. The Brute-Force approach guarantees the minimum of

$$\sum_{T_i \in TSC_1 \cap T_j \in TSC_1} DIS(T_i, T_j),$$

where $T_j \in TSC_2$ matches $T_i$ in $TSC_1$. In summary, by enumerating all possible matches between the time series in two TSCs, we can calculate the minimum distance of Brute-Force approach by the following equation:

$$\text{dist}_{BF}(TSC_1, TSC_2) = \min_k \left( \sum_{<T_m, T_n> \in M_k} (DIS(T_m, T_n)) \right),$$

(1)

where $M_k$ is subset of the Cartesian product between the set of time series in $TSC_1$ and $TSC_2$, each $M_k$ represents a possible matching result between the two sets, $T_m \in TSC_1$ and $T_n \in TSC_2$.

The Brute-Force TSC similarity matching approach is an exhaustive matching-based algorithm. The approach is very costly, as it requires an enumeration between all pairwise single time series. Meanwhile, this method can measure the patterns between time series in two TSCs well, but ignores the natural relations of time series inside the TSC. In order to reduce the computational cost and solve the challenge of evaluating the natural relations, we present a novel approach for TSC similarity search in next section, which exploits compact representation and relation vector of TSCs.

4 OUR APPROACH

Our goal is to design a representation that describe and measure the general information extracted from a TSC. The representation needs to capture both the intrinsic patterns and natural relations in a TSC. Overall, the computational complexity of the proposed TSC similarity search algorithm, which leverages the representation, must be significantly smaller compared to the Brute-Force approach.

In order to achieve this, we propose two data structures, called CT (Compact representation of TSC) and RT (Relation vector of TSC), as a compact way of representing a TSC and its inherent natural relations. CT consists of principal components (PC) that are extracted from the multiple time series in a TSC. RT captures the natural relations among the time series in a TSC, and is derived by analyzing the relations which is represented in a matrix. A TSC is completely described by a combination of its CT and RT. The CT and RT can be used to effectively compute the overall distance (i.e., $\text{dist}(TSC_1, TSC_2)$).

Using CT and RT representations for the TSCs, we propose a novel framework for similarity search in TSC databases. The overall framework of our approach is shown in Fig. 2. First, we generate the principal components of the observed values in multiple time series using Principal Component Analysis. Second, we make use of the transformed values on the first few PCs to form a (multiple) generalized time series—CT. Third, we make use of the feature correlation of multiple time series to form relation descriptors $R_{ij}$ for each pair of time series $T_i$ and $T_j$ in a TSC. Taking the earthquake waves (EW) in the flowchart as example, the figure shows three time series extracted from three earthquake observations in Tibet, as well as the spatial locations of these observations on the top right. It would be very useful for geologists to detect earthquakes with similar patterns, e.g., the nearby epicenter, the waveform of the earthquake, etc. We use PCA to extract the time series of the hidden variables to build CT. And to build RT, we first generate relation descriptors for each pair of the time series, based on their shapes and the location information, and then build relation matrix, which is factorized by Singular Value Decomposition to generate the RT. The CT and RT are used together to measure the similarity between two TSCs.

In general, there are $n^2$ relation descriptors in a TSC containing $n$ time series. Fourth, these high-dimensional relation descriptors are transformed into lower dimensional vectors $R_{ij}$ based on their PCs. These transformed relation descriptors are used to form the relation matrices $RM \{R_{ij}\}$. Finally, we generate a signature using Singular Value Decomposition of this matrix. The highest signatures of the series of matrices and the first transformation vector of PCA $W_1$ are used to obtain the RT feature.

4.1 Compact Representation of TSC

One of the key challenges of performing TSC similarity search is to efficiently match the patterns of multiple time series in TSCs. Using a compact representation can avoid the need to perform pairwise comparisons of all the time series, which is computationally expensive.

In the database literature, PCA is commonly used in time series analysis [3], [15] for dimensionality reduction, and to
facilitate fast and accurate retrieval. Fig. 3 shows several groups of objects and their corresponding principal components. The key idea in PCA is to transform a set of observed values in the high-dimensional vector space into a new vector space. Given a data matrix $X$, and a transformation matrix $W$. Each column of $X$ corresponds to a data vector in the original space. Each row of $W$ corresponds to a transformation vector. The transformation vector transforms the data vectors from the original space to the value of certain principal components. The PCA transformation is given as:

$$ Y = X \cdot W. $$

$Y$ denotes the data matrix in the new vector space. In the new vector space, each dimension corresponds to the principal component of the original space. It is generated from the original dimensions based on the statistical distribution of the observed values.

In our work, we make use of PCA to produce $CT$, a compact representation of the patterns of multiple time series in a TSC. Each $CT$ consists of one or several time series of the principal components extracted from the original TSC. In the SPIRIT system [15], these principal components, also known as hidden variables, are used primarily to perform pattern detection for data stream management. In our scenario, a $CT$ is used to represent the general feature of multiple time series patterns and to measure the similarity of TSCs. $CT$ provides a powerful summary of the time series in a TSC. It is generated based on each time series' observed values at every time tick and their (dis)/similarity.

Given a TSC, which consists of $n$ time series $T_1, T_2, \ldots, T_n$, for each time series $T_i$, the observed value in time tick $t_j$ is $x_{ij}$. Denote $i$ is the number of time ticks, i.e., the length of time series, the TSC can also be represented by $ln$-dimensional vectors where the vectors are the values of $n$ time series in a certain time tick. For each time tick, we build a $n$-dimensional vector $X_j = (x_{1j}, x_{2j}, \ldots, x_{nj})$. Each dimension refers to a time series in $T_1, T_2, \ldots, T_n$. Note that, we can normalize the time series with different lengths, which is a general solution for time series matching.

We perform PCA transformation on these $n$-dimensional vectors to identify the first few principal components. In the implementation, we choose the first $n', n' \leq n$ PCs. These $n'$ PCs can capture the most dominant information of the original data, even if $n' \ll n$. We only consider these PCs and their associated observed values which have been transformed in PCA dimensions.

As $n$ is the dimensionality of the original space, the $n \times n$ transformation matrix $W$ is given as follows:

$$ W = (W_1, W_2, \ldots, W_n) $$

$$ W_i = (w_{i1}, w_{i2}, \ldots, w_{in})^T, $$

where each row $W_i, i = 1, \ldots, n$ in $W$ is a $n$-dimensional transformation vector of a Principal Component. The transformation vector transforms the data from the original space to the dimensions of principal components by linear combination.

Thus, the new observed values of time tick $t_j$ on the dimensions of transformed PCA space are

$$ \tilde{X}_j = X_j \cdot W = (X_j \cdot W_1, X_j \cdot W_2, \ldots, X_j \cdot W_n). $$

The derived values in each time tick $t_j$ in $CT$ are the first $n'$ principal components of $X_j$, which are the first $n'$ dimensions of $X_j$. The transformation matrix which transforms the origin vector space into first $n'$ dimensions of PCs is

$$ W' = (W_1, W_2, \ldots, W_{n'}). $$

Therefore, $CT$ can be represented as

$$ CT = X \cdot W' = (X_1 \cdot W', X_2 \cdot W', \ldots, X_l \cdot W'), $$

where $l$ is the length of time series. The $l \times n'$ data matrix in this equation represents $n'$ time series (CT) which are generalized from the multiple patterns of original $n$ time series.

With the PCA transformation, the derived values in $CT$ are the combination of each dimension of $X_j, j = 1, 2, \ldots, l$ and such combination guarantees that the most dispersion represents the most useful information; hence, the patterns of multiple time series are generalized in $CT$ according to their appearances. Therefore, $CT$ can be the signature describing the general features of patterns in TSC, and using $CT$ in TSC similarity search can avoid matching time series between two TSCs. Comparing to the Brute-Force method which finds the best matches between every time series pair, $CT$ is a more general feature and is easier to measure the similarity, although information will be lost by only preserving the first one or few principal components, measuring the similarity between CTs is much more efficient than Brute-Force method. Moreover, the lost information can be compensated using $RT$ which will be introduced in next section.

### 4.2 Relation Vector Generation for TSC

The natural relations that are inherent in a TSC need to be captured for effective TSC retrieval. In the previous section, we saw how the key information of multiple time series can be represented using the $CT$ feature. In this section, we show how the natural relations that are inherent in the time series can be captured.

One of the key challenges for representation of the natural relations is defining a general measure for capturing and describing the relations from different application domains. In addition, the natural relations between multiple time series can be complex. In [21], [20], [19], [18], spatiotemporal relation descriptors of moving objects and time series intervals have been proposed. However, such
descriptors are generally not comparable and relevant to specific application domains.

We propose a novel concept of Relation vector for TSC (RT). A RT is a multidimensional vector, which can be used to describe the natural relations among multiple time series in a TSC. The RT consists of a set of signatures, which are extracted from a relation matrix. The relation matrix is obtained based on the relation descriptors of every time series pair in TSC. We first perform dimensionality reduction on all the relation descriptors to transform the relation descriptors to fewer meaningful dimensions, i.e., their principal components, according to the correlation of dimensions in the relation descriptors of TSC. Then, we use these transformed vectors to form a relation matrix of the TSC. The extraction of RT can be summarized as follows:

- For every two time series in a TSC, a relation descriptor is used to capture their intrinsic relationship. The relation descriptor is represented by a high-dimensional vector. Each dimension of the vector measures the difference between specific properties of two time series. For example, the dimension can capture the difference of the sampling positions for each time series, as well as the location or interval in which the time series are obtained.
- We make use of the PCA transformation to transform the high-dimensional relation descriptors into a summarized space. The transformation vector corresponding to the distribution of the vectors in origin space is recorded as one component of RT. The transformed relation descriptors are used to form the other part of RT in the next step.
- The relation descriptors which are obtained after performing PCA constitutes a series of $n \times n$ matrices. We refer to these matrices as the Relation Matrix of the TSC. Without loss of generality, we use $n$ to represent the number of time series in TSC objects. Each matrix corresponds to one dimension of the principal component of the relation descriptors, and the $\alpha$th row and $\beta$th column of the matrices corresponds to the relation descriptor between the $\alpha$th time series and $\beta$th time series. For each matrix, we use the singular value from the Singular Value Decomposition as the signature. The signatures of the series of matrices form the other part of RT feature.

It is important to note that the row or column order of time series is generally not specified in the relation matrix. Each column/row describes a time series’ relation with other members in TSC. Hence, when the rows or columns of the relation matrix are swapped, the similarity between the RTs is preserved. This leads to the invariance nature of signatures of the relation matrix to elementary row and column transformations on the matrix.

### 4.2.1 Relation Descriptor between Time Series

In this section, we propose the notion of relation descriptor. The relation descriptor is used to describe the natural relation between two time series, e.g., $T_m$ and $T_n$. Relation descriptor is a high-dimensional vector, where each dimension measures the difference between time series on a specific property, e.g., the difference of the start location of trajectories, the difference of begin time, etc.

The basic natural relations between $T_m$ and $T_n$ refer to the differences between the typical observed values in the time series and the associated time intervals. Such relation is commonly found in various application fields, e.g., the spatiotemporal relation of moving trajectories [21] in video clips. Since the observed values and interval of a time series are generally represented in uniform data format despite of different application areas, we can define the difference on these relations in the relation descriptor. In this paper, we define two basic relations, i.e., variance in spatial domain (VIS) and variance in temporal domain (VIT).

**Variance in spatial domain.** Given a time series as follows: \((x_1, t_1), (x_2, t_2), \ldots, (x_l, t_l)\), where $l$ is the length, \(\{x_i\}\) is the set of observed values, and \(\{t_i\}\) is the set of time ticks. The variance in spatial domain is a high-dimensional vector, each dimension of which captures the difference between the sampled observed values of $T_m$ and $T_n$

\[
\left(\|x_{m_1} - x_{n_1}\|, \|x_{m_2} - x_{n_2}\|, \ldots, \|x_{m_k} - x_{n_k}\|\right),
\]

where $x_{m_i}, x_{m_2}, \ldots, x_{m_k}$ is the sampled observed values of $T_m$, and $x_{n_1}, x_{n_2}, \ldots, x_{n_k}$ is the sampled observed values of $T_n$. Note that $s_1, s_2, \ldots, s_k$ can be generated based on some sampling method; in our approach, we only adopted equal length sampling for simplicity.

**Variance in temporal domain.** VIT is a descriptor of the relation between intervals of $T_m$ and $T_n$. Motivated by the discussion in [18], [20], we propose a 4D vector $VIT(T_m, T_n)$ to describe the differences of intervals in our approach

\[
VIT(T_m, T_n) = (t_{m_1} - t_{n_1}, t_{m_1} - t_{n_2}, t_{m_2} - t_{n_1}, t_{m_2} - t_{n_2}),
\]

where $t_{m_1}$ and $t_{n_1}$ are the first time ticks of $T_m$ and $T_n$, $t_{m_2}$ and $t_{n_2}$ are the last time ticks, respectively, the first two dimensions of VIT are the differences of two time series’ begin time and end time, which indicate the relation of which time series begins first and which one ends first; besides, $t_{m_1} - t_{n_1}$ and $t_{m_2} - t_{n_2}$ which are used to describe the overlap status between two time series’ interval, are defined as follows:

\[
t^{m,n}_0 = \begin{cases} 
0, & m = n, \\
0, & t_{m_1} - t_{n_1}, \ t_{m_1} \geq t_{m_2}, \ t_{m_2} < t_{n_1}, \end{cases} \tag{7}
\]

\[
t^{m,n}_1 = \begin{cases} 
(t_{m_1} - t_{n_1}), & t_{m_1} \geq t_{m_2}, \ t_{m_2} < t_{n_1}, \end{cases} \tag{7'}
\]

\[
t^{m,n}_2 = \begin{cases} 
0, & m = n, \\
0, & t_{m_1} - t_{n_1}, \ t_{m_1} \geq t_{m_2}, \ t_{m_2} < t_{n_1}, \end{cases} \tag{8}
\]

\[
t^{m,n}_3 = \begin{cases} 
(t_{m_2} - t_{n_1}), & t_{m_1} \geq t_{m_2}, \ t_{m_2} < t_{n_1}, \end{cases} \tag{8'}
\]

These four features can effectively model the pairwise temporal relation of time series, such as duration and overlapping status. It is also necessary to make $VIT(T_m, T_n)$ and $VIT(T_n, T_m)$ satisfy the antisymmetry relation: $VIT(T_m, T_n) = -VIT(T_n, T_m)$, because of the following two reasons:

- VIT describes the relation between the temporal intervals of two time series, which will be used to measure the distance between two temporal intervals.
Therefore, it must satisfy: \( \| VIT(T_m, T_n) \| = \| VIT(T_n, T_m) \| \) in measuring the similarity of two temporal intervals \( T_m \) and \( T_n \).

- VIT is a part of RT which is used to represent and measure the similarity of pairwise natural relations. Therefore, VIT should be measurable to compute the similarity of two temporal relations.

The four components of proposed VIT measurement satisfy the antisymmetry relation, the combination of which can represent all possible temporal relations between two time series and can be further integrated with VIS to form a complete relation descriptor for multiple time series in a clique.

In summary, the relation descriptor of \( T_m \) and \( T_n \) is the combination of two components, i.e., the differences between observed values and intervals. The relation descriptor is a high-dimensional vector which summarizes the overall intrinsic relation in both spatial and temporal domains by considering various properties of \( T_m \) and \( T_n \). Such descriptor is simple and robust for different applications as well as can be easily processed due to its vector form.

Note that, the VIS and VIT features are only two typical instances to describe the general information of relations between time series. Besides these relations on spatial and temporal domains, we can also explore other application-specific features to capture extra relations existing in time series cliques, such as the location of monitor station for earthquake waves, the personal ability or scores of the players that move in different trajectories. In many real-life applications, these information may play a critical role in describing the characters and organization of a TSC. These extra relations can be easily accumulated by appending new relations to the relation descriptors, e.g., the location difference between earthquake monitor stations, and the capability difference between players. In many applications, the feature selection is critical in similarity search problems. However, in this work, our purpose is to design a general relation descriptor to describe the overview information of natural relations among time series in a TSC for similarity measure base on our proposed framework; thus, we only adopt the common VIS and VIT features for clarity of presentation, and more domain-specific relations can be easily integrated.

### 4.2.2 Relation Matrix Construction for TSC

Inspired by the ERMH-BoW method which generates a relative motion histogram in a matrix form to describe relations of local video motion feature [23], we generate a Relation Matrix for TSC based on the relation descriptors introduced previously to better represent the natural relations and perform further feature extraction.

The relation descriptors of each time series pair in TSC are vectors in a high-dimensional vector space. Since the relation descriptors are generated from the differences of observed values and time intervals, there may exist redundant information as well as high dimensionality of such vectors may introduce expensive computational cost at the similarity search stage. Furthermore, the dimensions of relation descriptors are isolated from each other; thus, natural relations based on multiple properties cannot be well represented. Therefore, dimensionality reduction and dimension reorganization on the raw relation descriptors needs to be performed in order to obtain a good representation. In our approach, we make use of Principal Component Analysis to transform the relation descriptors in high-dimensional vector space to a new space. The new space consists of fewer dimensions, which exhibit higher correlation between the dimensions.

Given a TSC object which has \( n \) time series \( T_1, T_2, \ldots, T_n \), we can generate \( n^2 \) relation descriptors \( R_{ij} \) of each pair of time series \( T_i \) and \( T_j \) from the TSC object, where \( R_{ij} = \{ VIS(T_i, T_j), VIT(T_i, T_j) \} \) if we only consider the spatial and temporal relations. Using these \( n^2 \) vectors, we perform PCA to find the Principal components according to the data’s latent correlation and distribution on these dimensions. Set \( r \) as the dimensionality of each \( R_{ij} \), and \( r \times r \) transformation matrix \( W \) for PCA as \( (W_1, W_2, \ldots, W_r) \), we reserve only first few PCs which contain the majority information. The number of PCs remained is denoted as \( r' \).

Thus, we obtain the transformed relation descriptors \( \widetilde{R}_{ij} \)

\[
\widetilde{R}_{ij} = (R_{ij} \cdot W_1, R_{ij} \cdot W_2, \ldots, R_{ij} \cdot W_r),
\]

where we utilize \( r' \) PCs and transform all the relation descriptors \( R_{ij} \) into \( r' \)-dimensional vectors \( \widetilde{R}_{ij} \).

In the next step, we use the obtained \( n \times n \times r' \)-dimensional vectors \( \widetilde{R}_{ij}, i,j = 1, \ldots, n \) to form Relation Matrix \( RM \) of the TSC object. Each element in \( RM \)’s \( ith \) row and \( jth \) column is the relation descriptor of time series \( T_i \) and \( T_j \)

\[
RM_{ij} = \widetilde{R}_{ij}, \quad i,j = 1,2,\ldots,l.
\]

Because \( \widetilde{R}_{ij} \) is an \( r' \)-dimensional vector, \( RM \) is actually a series of matrices \( RM_1, RM_2, \ldots, RM_r \), each of which corresponds to one dimension of \( \widetilde{R}_{ij} \), represented by

\[
\begin{align*}
\widetilde{R}^1_{ij}, \widetilde{R}^2_{ij}, \ldots, \widetilde{R}^{r'}_{ij} \\
RM_{1ij} &= \widetilde{R}^1_{ij} = R_{ij} \cdot W_1 \\
RM_{2ij} &= \widetilde{R}^2_{ij} = R_{ij} \cdot W_2 \\
&\vdots \\
RM_{r'ij} &= \widetilde{R}^{r'}_{ij} = R_{ij} \cdot W_{r'}.
\end{align*}
\]

### 4.2.3 Generation of RT

The relation matrix has been generated to represent the relations in TSCs, however such data structure cannot be directly deployed for similarity measure of TSC objects. A feasible way is to use vectors to represent natural relations in TSCs for efficient processing, e.g., find a group of signatures to represent the relation matrix.

In the matrix \( RM \), the \( ith \) row, i.e., \( (\widetilde{R}_{i1}, \widetilde{R}_{i2}, \ldots, \widetilde{R}_{il}) \), represents the relation between time series \( T_i \) and other time series in the TSC object. The similar meaning can also be shown from the \( ith \) column of \( RM \).

In order to improve the efficiency of TSC similarity retrieval, the costly pairwise matching of time series needs to be avoided. Each row or each column of the RM describes a certain time series’ natural relation to other time series. It is important to note that the row or column order of time
series in an RM is not specified. This allows row or column swap in the RM, which does not change the value of the signatures. This ensures that the signatures extracted from the RM are invariant to elementary row and column transformation. Consequently, the singular value for each matrix in RM, is used as the signature. The RM is represented by an \( r' \)-dimensional vector, where each value in the \( i \)th dimension is the highest singular value of the corresponding matrix \( R M_i \).

The singular value can be computed using the Singular Value Decomposition of each matrix. It is invariant to elementary row and column transformation. The basic idea of SVD is as follows: For every matrix, \( A \in \mathbb{R}^{m \times n} \) can be decomposed into

\[
A = U \Sigma V^T, 
\]

where \( U \in \mathbb{R}^{m \times r} \), \( V \in \mathbb{R}^{n \times r} \), and \( \Sigma \in \mathbb{R}^{r \times r} \), with \( r \leq \min(m, n) \) as the rank of \( A \). Each row \( v_i \) of \( V \equiv [v_1, \ldots, v_r] \) is the right singular vector of \( A \) and forms an orthonormal basis of its row space. Similarly, each column \( u_i \) of \( U \equiv [u_1, \ldots, u_r] \) is the left singular vector and forms a basis of the column space of \( A \). \( \Sigma \equiv \text{diag} \{ \sigma_1, \ldots, \sigma_r \} \) is a diagonal matrix with positive values \( \sigma_i \), called the singular values of \( A \).

SVD comes from two orthonormal transformations, i.e., \( A = P V^T \) and \( A = U Q^T \), where \( U \) and \( V \) are orthonormal basis of the row space and column space, respectively. The SVD is a special case that \( P \) and \( Q \) correspond to each other. However, the \( P \) and \( Q \) can effectively represent \( A \) with a few first columns or rows, respectively. Such stratification means that the transformation \( V \) or \( U \) transforms the vectors in \( A \) to a new space where the first few values can effectively represent these vectors. In such case, the singular values \( \sigma_1, \sigma_2, \ldots, \sigma_r \) form the diagonal matrix \( \Sigma \) which can be used as the signatures of \( A \). The importance of each value decreases in the order of \( \sigma_1, \sigma_2, \ldots, \sigma_r \).

We use the first singular values of \( R M_1, R M_2, \ldots, R M_r \) to form an \( r' \)-dimensional vector as the signature of the relation matrix. The elements of RM are the relation descriptors after transforming to the first \( r' \) PCs, and the singular values from RMs of different TSCs is influenced by their transformation vectors of principal components; therefore, we use the singular value signature of RM along with the \( W_1 \) in (9), which is the first transformation vector of the relation descriptors, as Relation Vector of Time Series Clique (RT). The signature is a generalized descriptor of RM which stores the natural relation information extracted from the PCA space of the original relation descriptors, and \( W_1 \) represents the transformation direction of the first PC. Therefore, RT is an \( r' + r \)-dimensional vector, where the first \( r' \) dimensions store the Relation Matrix signature and the other \( r \) dimensions store the transformation vector \( W_1 \). The algorithm of generating the RT feature from Time Series Cliques is shown in Fig. 4.

A RT consists of two parts: The transform vector \( W_1 \) in (9), and the singular values of the Relation Matrices. These two parts of a RT, correspond to the distribution of the relation descriptor and signatures of the Relation Matrix, respectively. In our proposed framework, PCA and SVD are used mainly to summarize all the Relation Descriptors to a single descriptor. The single descriptor describes the pattern of the inner relation of the TSC, and is used for similarity matching. The \( W_1 \) generated by PCA is used to record the distribution information of the relation descriptors in a TSC. However, the \( W_1 \) does not capture the detailed information of the natural relation in the TSC. Therefore, SVD is used to extract the signatures for the matrices, i.e., the first singular value, which are used as the other component of RT. The use of PCA and SVD on the transformed data are applied to relation descriptors from different domains. Hence, it is important to note that it is not purely a dimensionality reduction process. PCA reduces the dimensionality and records the distribution information on the dimensions of extracted features of relation descriptors. SVD summarizes the matrices into a vector using unique singular values by mixing the dimensions of time series.

The feature RT generated by our framework is a powerful way for representing the natural relations of a TSC. It has the following desirable properties. First, the singular value captured in the RT is invariant to elementary row and column transformations in RM. This allows the signature of a RM to be preserved even if the ordering of two time series in the relation matrix are changed. This enables multiple time series matching to be performed effectively. Second, the RT is a simple, multidimensional vector. This allow the distance between TSCs to be measured efficiently. Third, the RT is generated from relation descriptors of pairwise time series in a TSC, and the relation descriptor is a simple and robust high-dimensional vector. This allows it to be used in many real-life applications.

### 4.3 Similarity Search on Time Series Cliques

We extract features from Time Series Cliques and generate \( CT \) and \( RT \) as the feature vectors. Specifically, \( CT \) provides a compact representation of characteristic patterns of time.
series in a TSC, and RT represents the natural relations among the time series in a TSC. Both of these two descriptors are generalized from detailed features, and hence can be efficiently used for similarity search in TSC databases.

Given two TSC objects which are represented by aforementioned features, e.g., \( TSC_i = \{CT_i, RT_i\} \) and \( TSC_j = \{CT_j, RT_j\} \), we define the distance function between two TSCs \( \text{dist}(TSC_i, TSC_j) \) based on the similarity measure for compact representation CT, i.e., \( \text{DIS}_C(CT_i, CT_j) \) and relation vector RT, i.e., \( \text{DIS}_R(RT_i, RT_j) \). CT expressed in (4) is the time series of PCs; thus, the distance function of \( \text{DIS}_C(CT_i, CT_j) \) can be the same as the distance function between time series \( \text{DIS}(T_i, T_j) \) in (1). Note that, based on the demand of specific application, any existing distance function of time series matching can be applied here, as it is orthogonal to our mechanism. For simplicity of presentation, we deploy the Euclidean Distance as the default distance function. As reported in Eamonn Keogh’s talk in Interface 2003 [24], he studied 11 different distance measures on time series matching, and concluded that no method performed better than traditional euclidean distance and Time Warping distance overall.

\( RT \) is a \((r' + r)\)-dimensional vector including the Relation Matrix Signature and the transformation vector \( W_i \) for the first PC in the PCA of relation descriptors. The euclidean distance can be used to measure the similarity between the Relation Matrix Signatures, because similar Relation Matrices have similar distributions; thus, the \( \Sigma \) in \( A = U \Sigma V_T \) is similar. We can see the relation between the Relation Matrix Signature and \( W_i \) from Fig. 3b which shows two groups of objects and their first PCs. In Fig. 3b, the smaller angle between the two first PCs and the more similar the data distributions on the first PCs are, the more similar the two groups of objects are. For these two groups of objects, we can measure their similarity by their principal component’s direction and the data distribution in such direction. Inspired by this example, the distance function of \( \text{DIS}_R(RT_i, RT_j) \) is defined as follows:

\[
\text{DIS}_R(\ldots) = \sqrt{\text{Euc}(S(RM_i), S(RM_j)) \cdot \sin \langle W_i, W_j \rangle}
\]

(11)

\[
S(RM) = (\text{svd}(RM_1), \text{svd}(RM_2), \ldots, \text{svd}(RM_l)),
\]

(12)

where \( \text{svd}(RM) \) represents the first singular value of \( RM \), and \( \text{Euc}(S(RM_i), S(RM_j)) \) is the euclidean distance between \( S(RM_i) \) and \( S(RM_j) \), \( \sin \langle W_i, W_j \rangle \) is the sine value of the angle between the first PCs from two \( RTs \).

According to the distance functions defined on the two features, the overall distance between two TSC objects \( \text{dist}(TSC_i, TSC_j) \) can be calculated by the geometric value of the two distances:

\[
\text{dist}(TSC_i, TSC_j) = \sqrt{\text{DIS}_C(CT_i, CT_j) \cdot \text{DIS}_R(RT_i, RT_j)}.
\]

(13)

The distance function \( \text{dist}(TSC_i, TSC_j) \) to compute TSC similarity fuses the distance functions of \( \text{DIS}_C \) and \( \text{DIS}_R \) which are the similarity measurement of time series patterns and their natural relations, respectively. In our approach, we extract generalized pattern information from multiple time series as CT and extract intrinsic natural relation vector RT based on pairwise relation matrices. By using these two components for TSC similarity evaluation, we solve the two challenges of effectively evaluating natural relationships and measuring the patterns of multiple time series in TSC; therefore, our approach can perform effectively and efficiently for similarity search in TSC databases. Note that, we can give different weights to two features according to user preference or domain applications if necessary.

5 PERFORMANCE ANALYSIS

In this section, we conduct an extensive performance analysis to evaluate the performance of the proposed TSC similarity search framework. We measure both the query effectiveness and the execution time for TSC retrieval. We compare our proposed approach named TSC with the Brute-Force approach presented in Section 3, and with the hidden-variable method in SPIRIT system [15]. SPIRIT is extended for TSC retrieval by using the hidden variables for similarity measurement between groups of time series. The values of the hidden variables capture the general pattern of multiple input time series based on the value distribution and correlations.

Note that, the selection of distance function is orthogonal to the approaches, since we can simply apply different functions to time series matching, such as euclidean distance, time wrapping distance. In our experiments, we have evaluated different distance functions which yield similar tendency on the performance comparison, which drew similar conclusion as in [24]. Due to the space constraint, we only detail the evaluation on euclidean distance for performance analysis.

All experiments are conducted using Matlab. The hardware consists of a server with a Core 8 2.8 GHZ CPU, and uses the Ubuntu Linux OS. The performance analysis results clearly show the superiority of our proposed approach for similarity search over TSC databases.

5.1 Data Description

As there are no benchmark data set that can be used for TSC similarity search problems, we devise a TSC data set for performance analysis. The TSC data set consists of both real-life and synthetic data sets. The real-life TSC data set consists of earthquake wave and video trajectory (VT) data. The synthetic data sets are generated with different configurations and parameters.

5.1.1 NHL Data Set

The first synthetic TSC data set is created based on the NHL data set to simulate the groups of players in sport games. The NHL data set was used in [22], which consists of National Hockey League players’ 2D trajectories. The trajectories are obtained by digitizing the Philadelphia Flyers’ hockey games during the NHL 2001-2002 season. Every trajectory in the NHL data set is a 2D time series \( \{(X_i, Y_i)|i = 1, 2, \ldots, l\} \), where the length \( l \) is equal to 256.

By checking the players’ movements in some sports game videos, we observe that the players’ trajectories in
The retrieval performance in NHL data set and EW data set is measured by precision/recall, which are commonly used in information retrieval community. Precision measures how precise the search results are (the number of correct results divided by the number of all returned results). Recall measures the completeness of available relevant results (the number of correct results divided by the number of results that should have been returned). As there is no annotated labels for VT data set to evaluate the retrieval performance, P@N, one of the popular metrics used in information retrieval, is used to measure the fraction of the top N videos retrieved that are relevant to the users interest. We choose 20 different video shots, which contain different type of motions, e.g., fast or slow motion with/without camera motion, as the set of queries and for each query the returned videos in top N with the highest similarity, e.g., 3, 5, 7, and 10 will be presented to three evaluators who determine whether the top-N returned images are relevant to the query video.

5.2 Performance Tuning

In the first experiment, we investigate two important parameters of the proposed TSC retrieval method. The first parameter is the number of PCs reserved in CT feature generation. Note in Section 4.1, we extract CT feature by using first $n_0$ PCs of multiple time series. The second parameter is the reduced dimensionality of Relation Descriptor in RT in Section 4.2, i.e., $r_0$ in (13), which is used to capture the natural relations among the time series in a TSC. We use the synthetic NHL data set on this experiment.

We first evaluate the effect of CT feature on TSC retrieval performance by varying the number of PCs $n_0$. As the maximum number of PCs in CT generation is the number of time series in TSC, and the minimum number of time series in the TSCs of our data set is 3, we conduct the experiments by varying $n_0$ from 1 to 3. The results of precision/recall for TSC retrieval are shown in Fig. 6.

We can see from Fig. 6a that the best performance is obtained when only the first PC is reserved. This is because the first PC summarizes sufficient general information of multiple patterns in TSCs, while the retrieval performance by using more PCs may suffer from redundant and useless information. Note that, when $n_0 > 1$, we generate multiple representative time series as CT features, and hence the incurred multiple time series matching may degrade overall performance. Given relatively small number of time series in a TSC, our experimental result is consistent with the findings in SPIRIT [15], where only 1 or 2 hidden variables are suggested to represent the patterns of multiple time series.

Next, we investigate the performance of TSC retrieval by varying the reduced dimensionality of Relation Descriptor in RT feature, $r_0$, which is shown in Fig. 6b. From the figure, we can observe that as more PCs are used, the performance is improved. This is because more information is included in the relation descriptor of the RT. Different from the utilization of PCs in CT, the additional PCs increase the dimensionality of relation vector. We can also observe diminishing performance improvements as the PCs increase. As the most dominant information is reserved in the first few PCs, we can see that the performance improvement becomes marginal when $r_0$ is large than 5.

In the following experiments, we fix the two parameters for performance comparison, i.e., $n_0 = 1$ and $r_0 = 5$.

5.3 Comparison with Other Methods

In this experiment, we compare the retrieval performance of our approach with the Brute-Force matching algorithm and the SPIRIT [15] method. The SPIRIT uses hidden variables to represent the multiple time series that corresponds to the CT feature used in our approach. However, the SPIRIT is enhanced with adaptively choosing the number of hidden variables for better pattern representations. We use NHL data set, EW data set, and VT data set for this experiment, and the results are shown in Fig. 7.
In Fig. 7a, we can see that the SPIRIT performs the worst among the three methods on the NHL data set. This is because the compact representation of multiple time series in SPIRIT, i.e., hidden variables, can only capture the limited pattern information, while ignoring the intrinsic natural relations between the multiple time series in the TSC, which are more valuable to distinguish different TSCs. The Brute-Force algorithm, comparing with the SPIRIT, yields better performance, because the algorithm matches all the time series in TSC exhaustively to explore the pairwise time series similarity. This method can address the challenge of matching different time series in TSC retrieval; however, the Brute-Force method also ignores the natural relations among the multiple time series in TSC. Therefore, the overall retrieval performance of Brute-Force method is also limited.

We can see that in Fig. 7a, our proposed TSC approach using the combination of CT and RT yields the best performance among the three algorithms. CT feature matches the similarity of group of time series by generalizing their characteristic patterns, and the RT information can capture the natural relations among time series in TSCs. The TSC approach can take both factors into effect for TSC retrieval, and hence improve the retrieval effectiveness in TSC databases. While the natural relations in a TSC cannot be captured by the hidden variables of SPIRIT and Brute-Force method, which are important to evaluate the similarity between time series cliques.

The results in Fig. 7b are from the EW data set. These results also show that the TSC approach yields the best performance, though all approaches degenerate compared with the results on NHL data set. This is because the real-life EW data have much noise and temporal variance, and their waves vary from different stations as shown in Fig. 5. Our approach, on the other side, although facing the same problem in generating CT, the RT generated from relation descriptors describing the natural relations can compensate the insufficiencies caused by data characters; hence, the CT+RT framework is more robust in different applications.

Fig. 8 shows the trajectory-based video retrieval results of three approaches on VT data set, where the performance of our method is more than 10 percent better than SPIRIT and Brute-Force methods. The combination of the two feature descriptors in our approach, (CT and RT) provides a scalable, effective, and efficient solution for TSC similarity search. In addition, the TSC retrieval mechanism integrates the compact representation of multiple time series and the natural relations among time series in cliques, and hence can be effectively applied in various application domains.

5.4 Scalability and Robustness
In this section, we investigate the scalability and Robustness of the proposed approaches. We first evaluate the time efficiency for the three approaches in terms of the scalability of the database size in NHL data set. We vary the size of TSC data set by utilizing 40, 80, 120, 160, 200 categories of TSCs, which results in the data size from 3,200 to 16,000 TSCs.

Fig. 9 shows the average time cost of these different approaches for one TSC retrieval in these data sets. We can observe that SPIRIT and the proposed approach are comparable and perform much better than the Brute-Force method. The SPIRIT utilizes the hidden variables by PCA for clique matching which is most efficient. Our TSC approach explores both CT and RT for similarity computation, and the extra computation on RT feature is needed compared with SPIRIT. While the Brute-Force algorithm performs badly by two orders of magnitude. This is because the Brute-Force algorithm needs to calculate all the possible matches for each time series pair in the cliques. Note that, the Brute-Force method has to compute the similarity for \( A_{n_1}^{n_2} \) combinations, where \( n_1 \) and \( n_2 \) are the numbers of time series in two TSCs and \( n_1 \geq n_2 \).

Table 1 shows the precision and recall of different approaches by varying the size on NHL and EW data sets. On NHL data set, we can see that, when the data size is relatively small, e.g., 3,200 or 6,400, the performance of Brute-Force is comparable with our TSC method. By exhaustively calculating the similarity between time series cliques, Brute-Force can yield satisfactory result on small data set where the cliques are more distinguishable. However, when the data size increases, our approach shows significant advantage over the other two competitors; the similarity measurement based on CT and RT can successfully identify the differences among cliques in a large TSC database. On the EW data set, we can see that our method has the best precision when recall is 0.1 in the two-month subset of 1,440 TSCs, followed by the SPIRIT algorithm. And the precisions of all three approaches on recall larger than...
0.1 are all very small, while the Brute-Force method have the best results. When the data size grow to 10,368, five times larger than the subset, we can see that our method dominates other methods, although the complete set contains a lot of noise and some local earthquakes whose magnitudes are lower than 6 but around 5 can be identified as earthquakes as well. Therefore, our method is more robust in terms of the data size scalability. Overall, our proposed TSC approach shows the superiority for both two performance metrics, i.e., efficiency and effectiveness.

We next proceed to evaluate the sensitivity to noise of the proposed method on synthetic NHL data set by producing different variations of TSCs. Specifically, we vary the Gaussian Noise on frequency domain from 10 to 25 percent of its origin value, and vary the time warping noise from 5 to 20 percent. Note that, the default noise is 5 percent and the results have been shown in Table 1. Table 2 demonstrates the results on precision and recall of different approaches. The results show the TSC’s robustness with respect to noise in the sense that it can take sufficiently large degree of variations. The proposed CT and RT feature vectors provide an effective and robust representation of TSC data. Brute-Force method is more sensitive to the noise though it compares each pair of time series from two TSCs exhaustively. The SPIRIT uses very compact representation which is sensitive to the noise on length and frequency; thus, the SPIRIT performs worst in this scenario although it is the most efficient.

### 6 Conclusion

In this paper, we focus on a novel Time Series Cliques similarity search problem. The goal is to retrieve a group of time series with natural relations. This topic has direct impact on many real-world applications, e.g., object-based video retrieval, earthquake monitoring, etc.

In order to capture the intrinsic properties of a TSC, we propose two novel, compact structures, called CT and RT feature vectors for TSC objects. Using a fusion of CT and RT feature vectors, we can effectively capture the general pattern of multiple time series and the natural relations that exist between the time series. Most importantly, the vectors can be used during TSC similarity search to eliminate the need for pairwise comparisons of time series in multiple TSCs. To evaluate the effectiveness and efficiency of our approach, we conducted a comprehensive performance analysis using both synthetic and real-life data sets. To our knowledge, the problem of time series clique similarity search has not been explored by previous works. Our work provides the first and exploratory investigation of the time series clique similarity search problem. The proposed CT+RT framework provides a good foundation for future research in time series clique similarity search for various application domains.

### Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant Nos. 61073019, 60933004, 60873063, and HJG Grant No. 2011ZX01042-001-001.

### References


