14th International Conference, DASFAA'09, Brisbane, Australia, pp.354-368, 2009

Periodic Pattern Analysis in Time Series Databases

Johannes Assfalg, Thomas Bernecker, Hans-Peter Kriegel, Peer Kröger, Matthias Renz
{assfalg,bernecker,kriegel,kroegerp,renz}@dbs.ifi.lmu.de

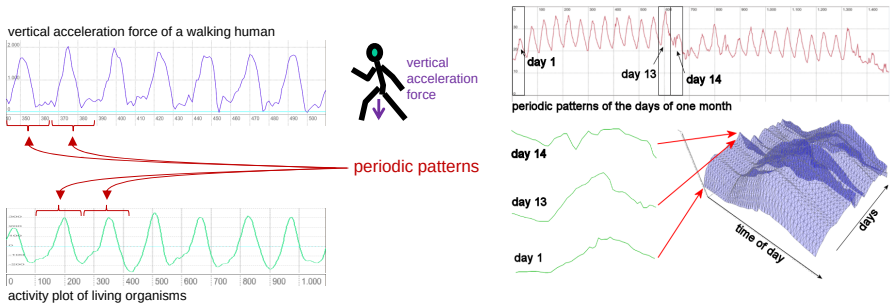
Institute for Informatics, Ludwig-Maximilians-Universität München, Germany

Abstract. Similarity search in time series data is used in diverse domains. The most prominent work has focused on similarity search considering either complete time series or certain subsequences of time series. Often, time series like temperature measurements consist of periodic patterns, i.e. patterns that repeatedly occur in defined periods over time. For example, the behavior of the temperature within one day is commonly correlated to that of the next day. Analysis of changes within the patterns and over consecutive patterns could be very valuable for many application domains, in particular finance, medicine, meteorology and ecology. In this paper, we present a framework that provides similarity search in time series databases regarding specific periodic patterns. In particular, an efficient threshold-based similarity search method is applied that is invariant against small distortions in time. Experiments on real-world data show that our novel similarity measure is more meaningful than established measures for many applications.

1 Introduction

In a large range of application domains, e.g. environmental analysis, evolution of stock charts, research on medical behavior of organisms, or analysis and detection of motion activities we are faced with time series data that feature cyclic activities composed of regularly repeating sequences of activity events. In particular for the recognition and analysis of activities of living organisms, cyclic activities play a key role. For example, human motions like walking, running, swimming and even working are composed of cyclic activities that correspond to significant motion events.

In this paper, we focus on similarity search on time series with a special focus on cyclic activities, in particular on the evolution of periodic patterns that repeatedly occur in specified periods over time. Examples of such time series are depicted in Figure 1(a). The upper time series shows the motion activity of a human, in particular the vertical acceleration force that repetitively occurs during a human motion like walking or running. Consecutive motion patterns show similar but distinct characteristics. We can observe changes in the shape of consecutive periodic patterns that are of significant importance if, for example, we want to analyze the motion behavior of any person. Many other applications that take advantage of the ability to examine the evolution of periodic patterns can be found in the medical or in the biological domain. For example, chronobiologists are highly interested in exploring the relationship between the activity of a cell or a complete organism and the amount as well as the duration of daylight affecting the cell or organism. Obviously, an important task is the identification of similar periodic



(a) Evolution of periodic patterns in medical and biological applications.

(b) Original and dual-domain representation of a time series.

Fig. 1. Origin, application and representation of periodic patterns.

patterns in daylight cycles and biological responses like the concentration of hormones (cf. Figure 1(a)). Another important domain where we find lots of time series containing periodic patterns is the environmental research. Examples are time series that describe the change of temperature values measured several times a day for each day within a month. In this case, the periodic pattern is the temperature course of a day. In order to be able to track the evolution of such periodic patterns, we propose to string consecutive patterns together to a sequence of patterns as shown in the example depicted in Figure 1(b). A time series is then split into a sequence of subsequences, which we call dual-domain time series. It represents the temporal behavior along a "second" time axis, e.g. each hour of a day. The original time domain is thus made coarser, e.g. it now represents each day of the entire time period. This way, we are able to define structures modelling the characteristic of the evolution of periodic patterns. In our example application, the new time series model allows us to examine the evolution of global climatic changes by considering the summer or winter months of the last 20 years. Contrary to [2], where the focus lies on the determination of periodicity features or the detection of motion directly from the periodic patterns, we take our attention to methods that help us to analyze the evolution of periodic patterns.

Given the new time series model, we are now interested in the examination of things that happen at a certain time. Thereby, we have our focus on the relationship between the times of both time domains at which certain events occur. Here, we take special emphasis on events that refer to an exceeding of a given activity threshold. Similarity search methods based on events that refer to exceeding of a given activity threshold have been introduced in [3, 4]. Given a certain threshold value τ, this approach reduces single-domain time series to a sequence of intervals corresponding to time periods where the amplitude value of a time series exceeds τ. Our approach represents periodic patterns as polygons, that analogously correspond to threshold-exceeding amplitude values. This approach is useful for a lot of application domains, where the exact value of a time series is less important than the fact whether a certain amplitude (activity) threshold is exceeded or not. Furthermore this approach is more robust to noise and errors in mea-

surement. We are subsequently able to identify similar threshold-exceeding patterns by comparing polygons. In order to efficiently perform similarity queries, we extract relevant feature information from those polygons.

The main contributions of this paper are the following: We introduce a new similarity measure for time series that takes two time domains into account. For the similarity measures we propose feature-based representations of dual-domain time series and show how they can be organized in an efficient way. The rest of this paper is organized as follows: First we introduce a matrix representation of dual-domain time series. Afterwards we introduce the so-called intersection set, that consists of the polygons generated by a threshold plane intersecting the dual-domain time series at a given threshold value τ. Furthermore, we present an approach to efficiently process index-supported similarity search based on periodic patterns. For that purpose we employ different features that are extracted from the intersection sets. Finally we evaluate the efficiency as well as the effectiveness of our approach in a broad experimental section.

2 Related Work

There are a lot of existing approaches performing similarity search on time series. Searching patterns can be supported by the Dynamic Time Warping approach (DTW) that is introduced for data mining in [7] and that presents a possibility to match the most corresponding values of different time series. Since the length of time series is very often quite large, the DTW approach suffers from its quadratic complexity with respect to the length of the time series. Thus a number of dimensionality reduction techniques exist. For example the Discrete Wavelet Transform (DWT) [1], the Discrete Fourier Transform (DFT) [16], the Piecewise Aggregate Approximation (PAA) [15, 23], the Singular Value Decomposition (SVD) [20], the Adaptive Piecewise Constant Approximation (APCA)[14], Chebyshev Polynomials [9], or the Piecewise Linear Approximation (PLA) [17] could be used. In [12], the authors propose the GEMINI framework, that allows to incorporate any dimensionality reduction method into efficient indexing, as long as the distance function on the reduced feature space fulfills the lower bounding property. However, those solutions are hardly applicable for searching similar patterns because in most cases, important temporal information is lost. In contrast to those solutions, in [21] the authors propose a bit sequence representation of time series. For each amplitude value, a corresponding bit is set if this value exceeds a certain threshold value. Similarity is finally computed based on those bits in an efficient way, since this approach lower bounds the Euclidean Distance or DTW. However, it is not possible to specify a certain threshold value at query time. This problem is addressed with inverse queries in [19].

Many approaches for similarity search on time series are based on features extracted from time series, i.e. in [18, 10, 13]. A similarity model for time series that considers the characteristics of the time series was proposed in [22], where a set of global features including periodicity, self-similarity, skewness and kurtosis among others is used to compute the similarity between time series. The features proposed in [5] are calculated over the whole amplitude spectrum. Thus, time-domain properties can be captured over the whole available amplitude range.

In this paper, we consider properties for the whole amplitude range of dual-domain time series which is novel to the best of our knowledge.

3 Time Series Representation

3.1 Dual-Domain Time Series

Intuitively, a dual-domain time series is a sequence of sequences, i.e. we have an amplitude spectrum and – in contrast to traditional single-domain time series – two time axes. More formally, a dual-domain time series is defined by

$$X_{dual} = \langle \langle x_{1,1}, \ldots, x_{1,N-1}, x_{1,N} \rangle, \ldots, \langle x_{M,1}, \ldots, x_{M,N-1}, x_{M,N} \rangle \rangle$$

where $x_{i,j}$ denotes the value of the time series at time slot i in the first (discrete) time domain $T = \{t_1, \ldots, t_N\}$ and at time slot j in the second (discrete) time domain $S = \{s_1, \ldots, s_M\}$. In the following, we call the $x_{i,j}$ *measurement configurations*. We assume $\forall i \in 1, \ldots, N-1 : t_i < t_{i+1}$ and $\forall j \in 1, \ldots, M-1 : s_j < s_{j+1}$.

Both axes T and S may also be any other ordered domain such as a spatial axis or a color spectrum, so that the concepts presented in this paper can also be applied to such types of data. The concepts can further be extended to the case of a multi-domain representation of time series. For the sake of presentation, we focus on dual-domain time series with two time domains, i.e. T and S are domains of discrete time slots.

3.2 Intersection Sets

As proposed in [3, 4], time series considering a single time domain can be represented as a sequence of intervals according to a certain threshold value τ. For the recognition of relevant periodic patterns that are hidden in the matrix representation of dual-domain time series, we extend this approach to a novel abstract meaning. Hence, we consider an abstraction of the time series. In case of multiple domains, we speak of an *n-domain* time series, where the dual-domain case corresponds to $n = 2$. Adding the amplitude axis to the n-domain time series yields an $(n+1)$-dimensional surface.

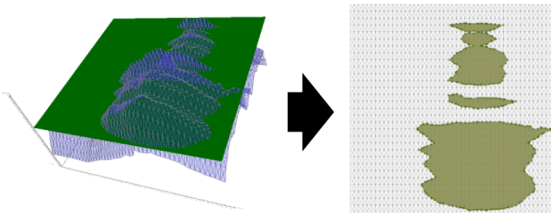


Fig. 2. Dual-domain time series with a threshold plane and the intersection polygon set.

The dual-domain time series can be structured using an elevation grid which is created by the grid squares of the measurement configurations $x_{i,j}$ where $1 \leq i \leq N$ and

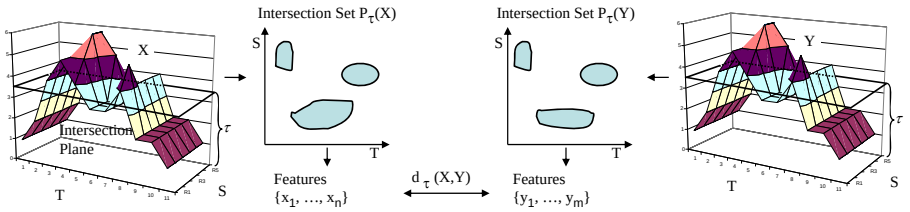


Fig. 3. Similarity measure for a given threshold τ.

$1 \leq j \leq M$ (cf. Section 3.1). Each grid square of a time series X_{dual} (in the following denoted as X for simplicity) can be denoted by $(x_{i,j}, x_{i+1,j}, x_{i+1,j+1}, x_{i,j+1})$ where $1 \leq i \leq N-1$ and $1 \leq j \leq M-1$. In our case, the threshold line for τ corresponds to an n-dimensional threshold hyperplane which intersects the time series. The result of this intersection is a set of n-dimensional polygons $P_\tau(X) = \{p_1, \ldots, p_K\}$ which we call the *intersection set*. The intersection set is created by intersecting the plane with the amplitudes of each of the grid squares. An example of an intersection set is depicted in Figure 2. The polygons of an intersection set with respect to a certain value of τ contain those amplitude values of the time series that are above the threshold plane τ, and thus, they deliver all the information about the periods of time during which the n-domain values of the time series exceed τ. With this abstraction, we are able to compare two time series with respect to coherences in time.

4 Similarity Query Processing

4.1 Similarity Measure and Feature Transformation

In order to analyze dual-domain time series based on periodic patterns with respect to a certain threshold τ, we have to define a distance value for such time series. As described above, the patterns of interest emerge as polygons forming intersection sets that are created. So the distance value $d_\tau(X, Y)$ of two dual-domain time series X and Y should reflect the dissimilarity of their corresponding intersection sets. In order to save computational cost and to allow for the usage of index structures like the R^**-tree* [6], we derive local or global features for the polygons and compare these features instead of the exact polygons (cf. Figure 3). In the following sections, we describe several local and global features suitable for capturing the characteristics of the intersection sets.

4.2 Similarity Measure based on Local Features

Local features describe a polygon p belonging to an intersection set $P_\tau(X)$ for a given dual-domain time series X and a given threshold value τ. Let a polygon p consist of $|p|$ vertices $v_1, \ldots, v_{|p|}$. Let vertex v_i be defined by the tuple $(x_i, y_i) \in T \times S$. Then the *Polygon Centroid* feature (PC) describes the position of the vertices by calculating their

central point. Formally, the PC feature of a polygon p is defined as

$$PC(p) = \frac{1}{|p|}\sum_{i=1}^{|p|} v_i = \left(\frac{1}{|p|}\sum_{i=1}^{|p|} x_i, \frac{1}{|p|}\sum_{i=1}^{|p|} y_i\right).$$

The *Polygon MBR* feature (PM) is a conservative approximation of a polygon. It describes a polygon by means of its minimal bounding rectangle (MBR). Formally, the 4-dimensional PM feature of a polygon p is defined as

$$PM(p) = \left(\min_{i=1..|p|}(x_i), \min_{i=1..|p|}(y_i), \max_{i=1..|p|}(x_i), \max_{i=1..|p|}(y_i)\right).$$

The number of polygons varies for different intersection sets and so does the number of local features. In order to calculate the distance value $d_\tau(X, Y)$ based on local features we employ the Sum of Minimal Distance (SMD) measure [11]. The SMD matches each polygon (i.e. the corresponding local feature) of $P_\tau(X)$ to its best matching partner of $P_\tau(Y)$ and vice versa:

$$d_\tau(X,Y) = \tfrac{1}{2}\left(\frac{1}{|P_\tau(X)|}\sum_{x\in P_\tau(X)}\left(\min_{y\in P_\tau(Y)} d(x,y)\right) + \frac{1}{|P_\tau(Y)|}\sum_{y\in P_\tau(Y)}\left(\min_{x\in P_\tau(X)} d(x,y)\right)\right)$$

where x and y are the features describing the elements of the intersection set and where $d(x, y)$ is a distance function defined on these features. In our case, this distance function is the Euclidean distance.

4.3 Similarity Measure based on Global Features

Contrary to local features, global features try to capture the characteristics of an intersection set by a single feature value or feature vector. In this section, we present three examples for global features.

Let $P_\tau(X)$ be an intersection set as described above. Let furthermore K be the number of polygons $P_\tau(X)$ consists of. Then the *Intersection Set MBR* feature (ISM) is the global version of the local PM feature approximating the complete set of polygons by a minimal bounding rectangle. So, ISM is defined analogously as

$$ISM(P_\tau(X)) = \left(\min_{\substack{i=1..|p|\\k=1..K}}(x_{k,i}), \min_{\substack{i=1..|p|\\k=1..K}}(y_{k,i}), \max_{\substack{i=1..|p|\\k=1..K}}(x_{k,i}), \max_{\substack{i=1..|p|\\k=1..K}}(y_{k,i})\right).$$

The *Intersection Set Centroid* feature (ISC) is the global variant of the local PC feature considering all polygon vertices of the intersection set. Let S be the overall number of all vertices of all polygons of $P_\tau(X)$. Then $ISC(P_\tau(X))$ analogously calculates the central point of all vertices of the intersection set:

$$ISC(P_\tau(X)) = \frac{1}{S}\sum_{i=1}^{S} v_i = \left(\frac{1}{S}\sum_{i=1}^{S} x_i, \frac{1}{S}\sum_{i=1}^{S} y_i\right).$$

A more sophisticated high-level feature is the *Fill Quota* feature (FQ). For each row and each column of the data matrix, the percentage of polygon coverage is computed. Hence, the horizontal and vertical values generate two single-domain time series that describe the position as well as the size of the polygons. The computation of the polygon coverage is processed based on the grid squares (cf. Section 3.1). Each grid square is tested for its contribution to the area of a polygon. For a dual-domain time series X that consists of N rows and M columns, we denote the coverage area of a grid square at the position (i, j) by $A_{i,j}, i = 1..N, j = 1..M$. For the i-th row and the j-th column, the values are computed as follows:

$$FQ(x_i) = \frac{1}{M} \sum_{j=1}^{M} A_{i,j} \text{ and } FQ(y_j) = \frac{1}{N} \sum_{i=1}^{N} A_{i,j}.$$

Afterwards we apply a standard technique for dimensionality reduction to the projected time series so that we store only n feature values $c_1, \ldots, c_n$ (for example Fourier coefficients) for each of the two projected time series $FQ(x)$ and $FQ(y)$. Note that $n \ll N$ and $n \ll M$. This leads to the following definition of the FQ feature:

$$FQ(P_\tau(X)) = (c_1(FQ(x)), \ldots, c_n(FQ(x)), c_1(FQ(y)), \ldots, c_n(FQ(y))).$$

The distance value for two intersection sets based on global features can be calculated without the SMD measure, as for each intersection set we derive the same amount of global features. So in this case, $d_\tau(X, Y)$ is calculated as the Euclidean distance between the associated global features.

5 Efficient Query Processing

In the previous section, we introduced similarity measures which are adequate to compare evolutions of periodic patterns in time series. In this section, we show how similarity queries based on the proposed similarity measures can be performed in an efficient way. In particular, we consider the ε-range query and the k-nearest-neighbor query which are the most prominent similarity query methods and are used as basic preprocessing steps for data mining tasks [8].

The proposed methods are based on the features extracted from the original time series as described in Section 4.1. Since, the feature extraction procedure, in particular the computation of the intersection sets, is very time consuming, it is not feasible to do at query time. For this reason, we propose to do the feature extraction in a preprocessing step and organize the precomputed features in an efficient way. For example, the features can be extracted during the insertion of the object into the database. Thereby we have to solve the problem that the features extracted from the objects are associated with certain amplitude-threshold values. For this reason, we either have to define a fixed threshold which is used for all similarity computations or we have to precompute the features for all possible threshold values. The former solution is too restrictive as it does not allow the option for an adequate threshold readjustment at query time. On the other hand there exists an unlimited number of thresholds which would have to be taken into account leading to immense storage overhead.

5.1 Feature Segments

In the following, we propose a method for the efficient management of preextracted features that allows for the specification of the threshold at query time. In fact, we have to take only a finite number of thresholds into account to derive features for all possible thresholds. In particular, for a dual-domain time series of size n (n different amplitude values), we need to extract only features for at most n different thresholds. The features for the remaining threshold values can be easily computed by means of linear interpolation. Let us assume that we are given all amplitude values of a dual-domain time series in ascending order of their amplitude values. The topology of the intersection sets associated with thresholds equal to two adjacent amplitude values does not change. Furthermore, also the change of the shape of all intersection sets associated with the corresponding thresholds is steady and homogeneous. As a consequence, we only need to extract the features at thresholds specified by all amplitude values occurring in a given time series. The features between two adjacent values of these amplitude values can be generated by linear interpolation, so we store each d-dimensional feature as a $(d+1)$-dimensional *feature segment*. An example is illustrated in Figure 4.

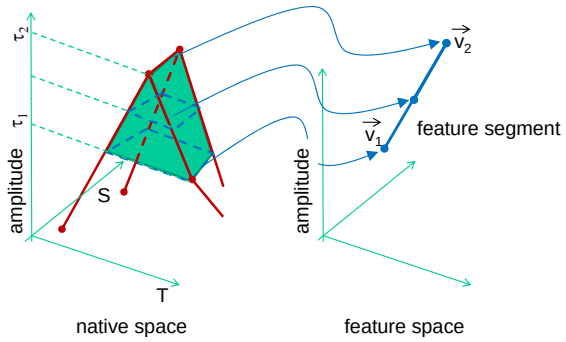


Fig. 4. Extracting feature segments from a time series.

On the left hand side, there is a section of a dual-domain time series from which we extract polygons (dotted lines) for the intersection planes at the two thresholds τ_1 and τ_2. The corresponding features points (vectors) $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ are sketched on the right hand side. The features of all polygons that result from intersection planes at thresholds between τ_1 and τ_2 are represented by the line between $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ called *feature segment*. Consequently, we only need to extract and manage a set of feature segments corresponding to a finite set of thresholds which are bound by the size of the corresponding dual-domain time series. In order to calculate the features of all objects at query time, we have to intersect the feature segments with the intersection plane corresponding to the given query threshold τ. Obviously, at query time we only need to take those feature segments into account that intersect the query threshold τ. The query

cost can be reduced if we use an adequate organization of the feature segments that allows us to access only the feature segments which are relevant for a certain query.

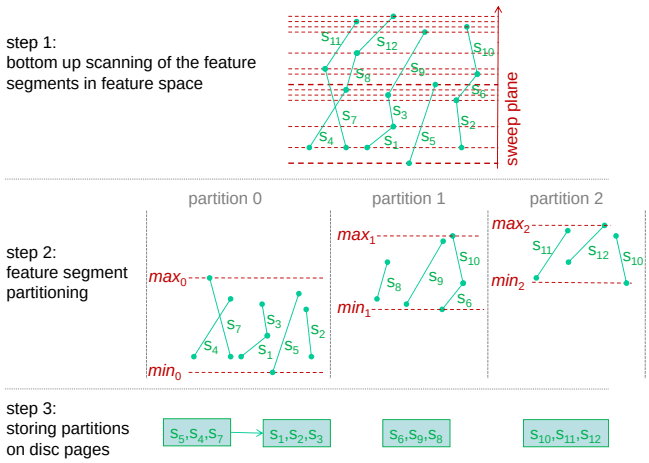


Fig. 5. Partitioning and storing of feature segments.

5.2 Feature Segment Organization

After extracting the feature segments from all time series objects, they are partitioned and stored in disc blocks of equal size. As mentioned above, for efficiency reasons it is necessary to organize the feature segments of all objects in such a way that given a query threshold τ only those segments need to be accessed that intersect τ. For that purpose, we sort the feature segments of all objects according to the lower bound of the segments in ascending order and partition them into groups which are stored into equal-sized disc blocks. This is done using a sweep plane as illustrated in the example depicted in Figure 5 (step 1). During the sweep plane scan over the feature space, the feature segments which have been reached by the sweep plane are collected into a group. After a fixed number k of segments has been collected (the number k is based on the capacity of a disc block. In our example the disc block capacity k is set to 3), the minimal amplitude vale min_i and the maximal amplitude value max_i over all segments collected so far are computed (step 2). Then the algorithm proceeds with the next segments. If a new segment lies within min_i and max_i, it is added to group i using a new disc block. This new block is concatenated to the existing blocks of group i. In case group i is not suitable for the storage of a segment, a new group is created. The bounds of this new group are determined as soon as the first block of the new group is filled.

The resulting disc blocks are organized group-wise within the following indexing structure. As mentioned above, each feature segment group i consists of concatenated disc blocks containing the feature segments and the minimum and maximum value

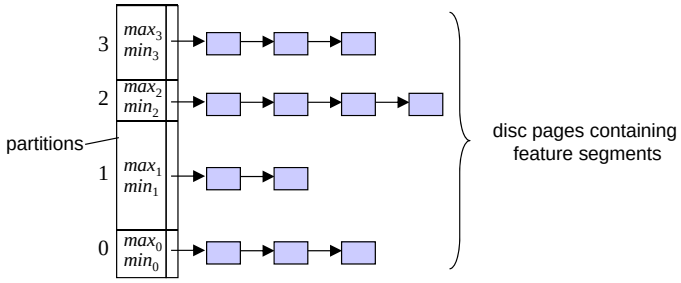


Fig. 6. Efficient organization of disc pages containing the feature segments.

min_i and max_i. The two values min_i and max_i are used to index the feature segment groups. The index consists of an array of triples (min_i, max_i, ref) which correspond to the feature segment group i. The triple entity ref is a pointer to the disc blocks containing the feature segments of the corresponding group. The array entries are sorted in ascending order of the corresponding min_i value. The index structure is illustrated in Figure 6.

5.3 Query Processing

At query time, the introduced index allows us to efficiently search for the relevant feature segment group using binary search over the triple entries min_i. Here, we assume that the array fits into main memory. Otherwise, a secondary storage structure is required to index the feature segment groups w.r.t. the min_i value. Following the ref pointer all relevant feature segments can be sequentially accessed by scanning the corresponding disc pages. If a new object is inserted into the database, its feature segments are generated and sorted in ascending order w.r.t. their minimum amplitude value. The algorithm then tries to insert the new feature segments into existing groups. A new group is not generated until the insertion of new segment entries would require an enlargement of the max_i value of one of the existing group i. In case of a deletion of an object, we need a complete scan over the feature segment groups and remove the corresponding entries. Afterwards we try to merge disc blocks of a group that are not completely filled anymore. If a group consists of only one disc block which is less than half full, then the group is deleted and the remaining entries of the disc block are assigned to one of the neighboring groups.

6 Experimental Evaluation

6.1 Datasets

We evaluated the effectiveness of similarity search on dual-domain time series utilizing two real-world datasets. The *TEMP* dataset contains environmental time series data

created by stationary measurements of several years[1]. It consists of 60 temperature time series from the year 2000 to 2004 that have been preclassified corresponding to the seasons summer and winter. Each object consists of up to 31 days and each day is represented by 48 measurements that have been normalized because of matching reasons on different ranges of the temperature measures that occur with different months. The *NSP* dataset is a chronobiologic dataset describing the cell activity of Neurospora[2] within sequences of day cycles. This dataset is used to investigate endogenous rhythms. We converted single-domain time series that describe cell activities by splitting the measurements according to the artificial day cycle. The *NSP* dataset consists of 120 objects and has been classified into five classes according to the day cycle length (16, 18, 20, 22, and 26 hours). The efficiency evaluation was performed on an artificial dataset that contained 10-1000 objects. We created two subsets having different resolutions: Each time series of the dataset ART_{20} consisted of 20×20 measurements. Analogously, the dataset ART_{50} contained objects having a resolution of 50×50 measurements.

6.2 Effectiveness of the Time Series Representation

Considering the single-domain representation of time series, we performed similarity queries on the given datasets utilizing the techniques that are applicable for computing similarity on single-domain time series, such as the Euclidean distance (in the following denoted as *EUCL*), the DTW [7] and the threshold-based approach [3, 4], in the following referred to as *THR*. Later in this section, we outline the obtained results of our newly introduced approach of measuring the distances for comparison. For an explanation of the tables and the curves that appear in this section, we give a short overview of the distance measures that have been considered for the experimental evaluation in Table 1.

Abbreviation	Description
EUCL	Euclidean distance
DTW	Dynamic Time Warping
THR	Threshold Distance (single-domain approach)
ISC	Intersection Set Centroid feature
ISM	Intersection Set MBR feature
FQ	Fill Quota feature
PC	Polygon Centroid feature
PM	Polygon MBR feature

Table 1. Distance measures considered for the evaluation.

Table 2 lists the average precision values for the different similarity measures utilizing the single-domain representation of the datasets. In comparison to the Euclidean distance and the DTW approach, the threshold-based similarity measure hardly leads

[1] http://www.lfu.bayern.de/

[2] Neurospora is the name of a fungal genus containing several distinct species. For further information see *The Neurospora Home Page*: http://www.fgsc.net/Neurospora/neurospora.html.

to higher average precision values. This can be observed for both datasets *TEMP* and *NSP*. As we outline in the following, with our newly introduced dual-domain representation approach in combination with a suitable threshold and the presented features, we are able to improve these results. We also compared the effectiveness of the local and global features that have been introduced in Section 4 using our threshold-based approach for different values of τ.

Measure	EUCL	DTW	THR	ISC	ISM	FQ	PC	PM
$\tau = 0.25$	0.58	0.62	0.55	0.54	0.55	0.55	**0.56**	0.55
$\tau = 0.5$	0.58	0.62	0.58	**0.65**	0.54	0.59	0.61	0.61
$\tau = 0.75$	0.58	0.62	0.56	0.60	**0.67**	0.58	0.61	0.60

(a) Average precision achieved on the *TEMP* dataset.

Measure	EUCL	DTW	THR	ISC	ISM	FQ	PC	PM
$\tau = 0.25$	0.56	0.52	0.56	0.73	**0.99**	0.74	0.41	0.53
$\tau = 0.5$	0.56	0.52	0.60	0.80	**0.93**	0.63	0.55	0.59
$\tau = 0.75$	0.56	0.52	0.47	0.51	**0.67**	0.46	0.52	0.52

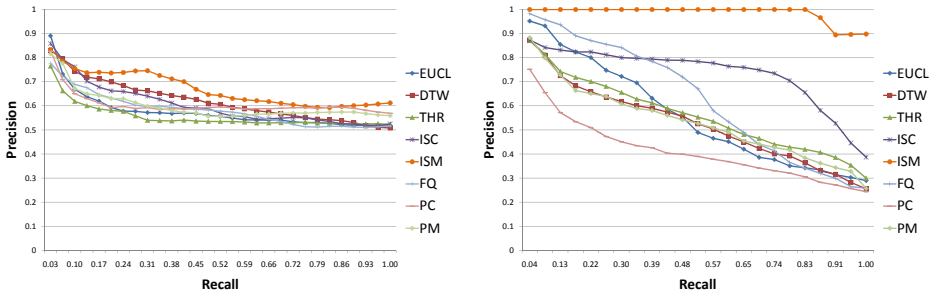
(b) Average precision achieved on the *NSP* dataset.

Table 2. Average precision for different measures on the single-domain and the dual-domain time series representation for a given threshold τ.

The results vary significantly with the threshold τ and also with the datasets. Depending on τ and on the used feature we outperform the Euclidean distance calculated on the time series. For the *TEMP* dataset and the local features (PC and PM) a threshold value of $\tau = 0.5$ yields the best results. The global features perform better for a threshold value of $\tau = 0.75$ (cf. Table 2(a)). The evaluation of the feature-based approach using the *NSP* dataset leads to different results. Especially the utilization of the ISM feature leads to a high degree of effectiveness. In general, similarity search based on the dual-domain time series representation leads to better results with higher average precision in comparison to the single-domain approach. Figure 7 depicts two precision-recall plots that support this statement. In this figure we included the results for the single-domain representation (DTW and Euclidean distance) as well as for the dual-domain representation in combination with different features.

6.3 Efficiency of Threshold-Based Similarity Search on Dual-Domain Time Series.

In order to evaluate the efficiency of our newly introduced approach we performed ϵ-range queries with a varying database size on the datasets ART_{20} and ART_{50} and



(a) Evaluation of the *TEMP* dataset having $\tau = 0.75$.

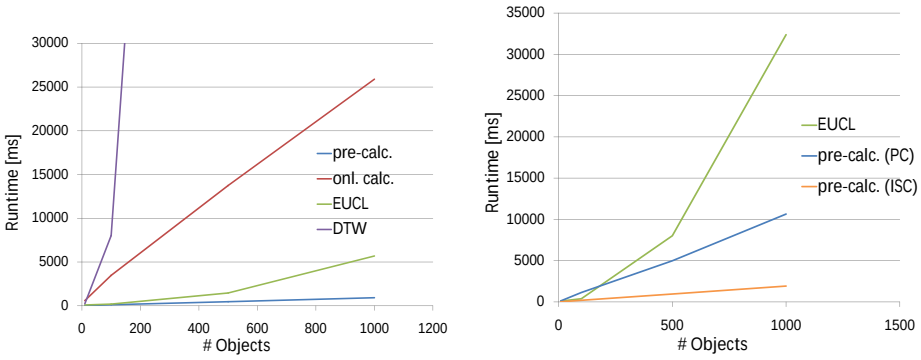
(b) Evaluation of the *NSP* dataset having $\tau = 0.25$.

Fig. 7. Precision-recall plots for different features and different representations.

measured the query time. The query objects were selected randomly and averaged the results. We examined the benefit of precalculating the features by storing them as segments as described in Section 5. The corresponding results are marked with "pre-calc". For comparison, we calculated the intersection sets and features at query time and labelled the results with "onl. calc". Furthermore, we performed similarity search using the traditional measures having a single-domain time series representation (i.e. Euclidean distance and DTW). Figure 8(a) depicts that calculating the required information at runtime is significantly more expensive than retrieving the information from our precalculated segments. However, the DTW can be outperformed anyway. Utilizing pre-calculation yields a better runtime than if the Euclidean distance is applied. Here, the threshold-based approach benefits from its reduction of dimensionality. Obviously, the runtime for the dataset ART_{50} is significantly higher than for the dataset ART_{20}, which is due to the complexity of the data and thus of the intersection sets. Figure 8(b) depicts a difference in the runtime comparing local and global features, representatively performed using the PC and the ISC feature. This is due, on the one hand, to the SMD that has to be applied with the local features but not with the global features, and on the other hand to the number of local features that is significantly higher than that of the global features, since each polygon has to be described separately.

7 Conclusions

In this paper, we proposed a new approach to perform similarity search on time series having periodic patterns in an effective and efficient way. Regarding single-domain time series having periodic patterns, the threshold-based approach can hardly improve the results of similarity computations in comparison to traditional techniques like the Euclidean distance or the DTW. Transforming the time series into the dual-domain space and thus considering the periodicity, we can better observe how the patterns change in time. Furthermore, the ability to focus on relevant amplitude thresholds by utilizing



(a) Results for dataset ART_{20}.

(b) Results for dataset ART_{50}.

Fig. 8. Results of the efficiency evaluation having a threshold value of $\tau = 0.25$.

the extraction of polygons from the time series and the usage of suitable, even simple features enables us to get better results for periodic pattern analysis. The quality of the results when utilizing an arbitrary feature however depends on the datasets. As a consequence, the effectiveness of global and local features varies with the type of data. The results with respect to the performance show a clear tendency. Regarding the traditional techniques and further a straightforward approach of computing the polygons and extracting the features from the time series at query time, we can reduce the runtime of similarity queries significantly by performing the computation and extraction in a preprocessing step. Furthermore, similarity computations using global features can be processed more efficiently than with local features.

References

1. R. Agrawal, C. Faloutsos, and A. N. Swami. Efficient similarity search in sequence databases. In *Foundations of Data Organization and Algorithms, 4th International Conference, FODO'93, Chicago, Illinois, USA, October 13-15, 1993, Proceedings*, pages 69–84, 1993.
2. A. B. Albu, R. Bergevin, and S. Quirion. Generic temporal segmentation of cyclic human motion. *Pattern Recognition, Elsevier Science Inc., New York, NY, USA*, 41(1):6–21, 2008.
3. J. Aßfalg, H.-P. Kriegel, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. Similarity search on time series based on threshold queries. In *Advances in Database Technology - EDBT 2006, 10th International Conference on Extending Database Technology, Munich, Germany, March 26-31, 2006, Proceedings*, pages 276–294, 2006.
4. J. Aßfalg, H.-P. Kriegel, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. Threshold similarity queries in large time series databases. In *Proceedings of the 22nd International Conference on Data Engineering, ICDE 2006, 3-8 April 2006, Atlanta, GA, USA*, page 149, 2006.
5. J. Aßfalg, H.-P. Kriegel, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. Similarity search in multimedia time series data using amplitude-level features. In *Advances in Multimedia*

Modeling, 14th International Multimedia Modeling Conference, MMM 2008, Kyoto, Japan, January 9-11, 2008, Proceedings, pages 123–133, 2008.

6. N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and rectangles. In *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, Atlantic City, NJ, May 23-25, 1990*, pages 322–331, 1990.
7. D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD Workshop*, pages 359–370, 1994.
8. F. K. C. Böhm. The k-nearest neighbor join: Turbo charging the kdd process. In *Knowledge and Information Systems (KAIS), Vol. 6, No. 6*, 2004.
9. Y. Cai and R. T. Ng. Indexing spatio-temporal trajectories with chebyshev polynomials. In *SIGMOD Conference*, pages 599–610, 2004.
10. K. Deng, A. Moore, and M. Nechyba. Learning to recognize time series: Combining arma models with memory-based learning. In *IEEE Int. Symp. on Computational Intelligence in Robotics and Automation*, volume 1, pages 246–250, 1997.
11. T. Eiter and H. Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34(2):109–133, 1997.
12. C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data, Minneapolis, Minnesota, May 24-27, 1994*, pages 419–429, 1994.
13. X. Ge and P. Smyth. Deformable markov model templates for time-series pattern matching. In *KDD*, pages 81–90, 2000.
14. E. J. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In *SIGMOD Conference*, pages 151–162, 2001.
15. E. J. Keogh, K. Chakrabarti, M. J. Pazzani, and S. Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. 2000.
16. F. Korn, H. V. Jagadish, and C. Faloutsos. Efficiently supporting ad hoc queries in large datasets of time sequences. In *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, pages 289–300, 1997.
17. Y. Morinaka, M. Yoshikawa, T. Amagasa, and S. Uemura. The L-index: An indexing structure for efficient subsequence matching in time sequence databases. In *PAKDD*, 2001.
18. A. Nanopoulos, R. Alcock, and Y. Manolopoulos. Feature-based classification of time-series data. *Information processing and technology, isbn 1-59033-116-8*, pages 49–61, 2001.
19. A. Nanopoulos and Y. Manolopoulos. Indexing time-series databases for inverse queries. In *Database and Expert Systems Applications, 9th International Conference, DEXA '98, Vienna, Austria, August 24-28, 1998, Proceedings*, pages 551–560, 1998.
20. K. pong Chan and A. W.-C. Fu. Efficient time series matching by wavelets. In *Proceedings of the 15th International Conference on Data Engineering, 23-26 March 1999, Sydney, Austrialia*, pages 126–133, 1999.
21. C. A. Ratanamahatana, E. J. Keogh, A. J. Bagnall, and S. Lonardi. A novel bit level time series representation with implication of similarity search and clustering. In *Advances in Knowledge Discovery and Data Mining, 9th Pacific-Asia Conference, PAKDD 2005, Hanoi, Vietnam, May 18-20, 2005, Proceedings*, pages 771–777, 2005.
22. X. Wang, K. A. Smith, and R. J. Hyndman. Characteristic-based clustering for time series data. *Data Min. Knowl. Discov.*, 13(3):335–364, 2006.
23. B.-K. Yi and C. Faloutsos. Fast time sequence indexing for arbitrary lp norms. In *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*, pages 385–394, 2000.