In Proc. 12th Int. Conf. on Database Systems for Advanced Applications (DASFAA '07), Bangkok, Thailand, 2007

Interval-focused Similarity Search in Time Series Databases

Johannes Aßfalg, Hans-Peter Kriegel, Peer Kröger, Peter Kunath, Alexey Pryakhin, Matthias Renz

Institute for Computer Science, Ludwig-Maximilians Universität München
Oettingenstr. 67, 80538 Munich, Germany
http://www.dbs.ifi.lmu.de/
{assfalg,kriegel,kroegerp,kunath,pryakhin,renz}@dbs.ifi.lmu.de

Abstract. Similarity search in time series databases usually deals with comparing entire time series objects or subsequence search. In this paper, we formalize the notion of interval-focused similarity queries which take a set of intervals specifying relevant time frames as additional parameter and compare the time series objects only within this user-defined time focus. We propose an original method to efficiently support interval-focused distance range and k-nearest neighbor queries implementing a filter/refinement architecture. In our broad experimental evaluation we show the superiority of our novel approach compared to existing approaches on several real-world data sets.

1 Introduction

Similarity search in time series databases has attracted a lot of research work recently. Existing work usually focus either on a full comparison, i.e. the entire time series are compared by using an appropriate distance function, or on subsequence matching, i.e. all time series objects that "match" a subsequence are retrieved. However, in many applications, only predefined parts of the time series are relevant for a similarity query rather than the entire time series data. The time intervals of these predefined parts are fixed for all time series. Usually, these parts are specified by the user depending on the analysis focus and change from query to query. We call such type of queries where only a small part of the entire time series is relevant *interval-focused similarity queries*. Obviously, interval-focused similarity is a generalization of a full comparison of the time series. On the other hand, the subsequence matching approach is orthogonal to interval-focused similarity. In interval-focused similarity search, the interval relevant to the query is fixed for all time series objects. In subsequence matching, the matching sequences usually do not correspond to a common time frame.

The notion of interval-focused similarity queries is an important concept in many applications. In stock marketing analysis, the behavior of the courses of different securities is examined w.r.t. a given set of events such as political crises or seasonal phenomena. The time courses need to be compared using interval-focused similarity queries that take only some relevant time periods into account

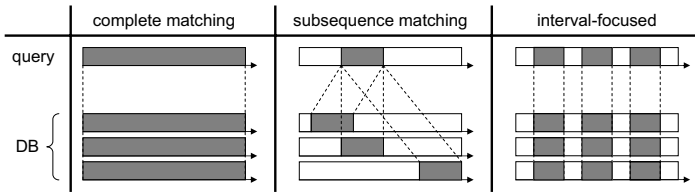

Fig. 1. Different approaches for time series analysis.

(e.g. a certain time period after the events). The analysis of the annual balances of a company is usually also focused on specific time intervals (e.g. months), i.e. the balances of specific months are compared using interval-focused similarity queries. In environmental research, the analysis of environmental parameters such as the temperature or the ozon concentration measured over long time periods at various locations usually focus on a given period during the year, e.g. compare the temperatures occurring only in the first week of July each year. Last but not least, in behavior research, brain waves of animals are recorded throughout a given time period, e.g. a day. Researchers often want to compare the brain waves of different individuals during a significant time interval, e.g. during feeding. Obviously, in all these applications, the focus of the analysis task frequently changes from time to time and is not known in advance.

In this paper, we formalize the novel notion of interval-focused similarity queries which is an important generalization of comparing entire time series. In addition, we propose an original method to efficiently support interval-focused distance range and k-nearest neighbor queries that implements a filter/refinement architecture. Furthermore, we discuss how the interval representation approximating the time series can be efficiently accessed using an index structure. The remainder is organized as follows. We discuss related work in Section 2. The novel notion of interval-focused similarity search is formalized in Section 3. In Section 4, we introduce the concept of interval-based representation of the time series. We further show how these representations can be managed efficiently in order to upper and lower bound the distance between time series objects. Based on these bounds we present a filter-refinement architecture to support interval-focused similarity queries efficiently. We discuss two methods for generation interval representations of time series in Section 5. Section 6 provides an experimental evaluation of our proposed methods. Section 7 concludes the paper.

2 Related Work

The (dis)similarity between two time series is usually measured by an appropriate distance function, e.g. the Euclidean distance, Dynamic Time Warping (DTW), Pearson's correlation coefficient, or angular separation, also known as cosine distance. Recent approaches focus either on an entire matching of the query time series with the database objects, or on subsequence matching.

Entire matching approaches consider the complete time course using any of the above mentioned distance measures (cf. Figure 1 (left)). Since the length of a time series is usually very large, the analysis of time series data is limited by the well-known *curse of dimensionality*. The GEMINI method [5] can exploit any dimensionality reduction for time series as long as the distance on the reduced data representation is always a lower bound of the distance on the original data (lower bounding property). In [10], the GEMINI framework is adapted for k-nearest neighbor search. Several dimensionality reduction techniques have been successfully applied to similarity search in time series databases, e.g. [1, 11, 4, 12, 7, 2, 6, 3, 12]. In [9] the authors use a clipped time series representation rather than applying a dimensionality reduction technique. Each time series is represented by a bit string indicating the intervals where the value of the time series is above the mean value of all values of the time series. A distance function that lower bounds the Euclidean distance and DTW is proposed. Obviously, entire matching is a special case of interval-focused similarity. Since all mentioned approximation techniques employing dimensionality reduction or clipping are not designed for interval-focused similarity queries they cannot optimally support this novel query type, especially if the intervals relevant for the query are changing over time and are not known beforehand. In that case, the proposed methods need to approximate the entire time series objects. To answer interval-focused queries these methods need to access the entire approximations rather than only the relevant parts. Subsequence matching approaches usually try to match a query subsequence to subsequences of the database objects (cf. Figure 1 (middle)). The similarity is not affected by the time slot at which o best matches the subsequence q. Usually, a subsequence matching problem is transferred into an entire matching problem by moving a sliding window over each time series object in the database and materializing the corresponding subsequence. If the length of the query subsequence changes, a new sliding window has to be moved over each database time series again. Obviously, subsequence matching is orthogonal to interval-focused similarity. In interval-focused similarity, the time slot relevant for matching is fixed. Two time series are not considered similar even if they have a similar subsequence but at different time intervals. In addition, the concept of interval-focused similarity allows to specify multiple relevant time intervals of different length.

3 Problem Statement and Contributions

Let $\mathcal{D}$ denote a database of n time series. A time series $X = [x_1, \ldots, x_N]$ of length N is a sequence of N values, where x_i denotes the value corresponding to the time slot $i \in \mathcal{T} = \{t_1, \ldots, t_N\}$ and $\mathcal{T}$ is the domain of time. We assume that all time series are normalized within the interval $[MAX, MIN]$, i.e. $\max_{x_i \in X} x_i = MAX$ and $\min_{x_i \in X} x_i = MIN$ for all time series objects $X \in \mathcal{D}$.

A *(time) interval* $I = (lT_I, uT_I) \in \mathcal{T} \times \mathcal{T}$ is a pair of time slots where lT_I denotes the start slot and uT_I denotes the end slot. Given a time series $X \in \mathcal{D}$ and an interval I, the *interval sequence of X corresponding to I* is a time series

of length $(uT_I - lT_I) + 1$ consisting of the values of X between the start and the end time slot of I, i.e. $X_I = [x_{lT_I}, \ldots, x_{uT_I}]$. A set of k intervals is denoted by $\mathcal{I} = \{I_1, \ldots I_k\}$.

Due to space limitations, we focus on the L_p-norms which are classical distance measures for time series, especially the Euclidean distance ($p = 2$). The proposed concepts can easily be adapted to DTW. The L_p-norm between two time series X and Y is defined as

$$L_p(X, Y) = \sqrt[p]{\sum_{i=1}^{N} (x_i - y_i)^p}.$$

As discussed above, interval-focused similarity specifies a given part of the time series (i.e. an interval) as relevant, whereas the remaining part of the time series is irrelevant. The relevant part may change from query to query. Let $I = (lT_I, uT_I)$ be a relevant interval. The L_p-norm between X and Y w.r.t. I is defined by

$$L_p^I(X, Y) = \sqrt[p]{\sum_{i=lT_I}^{uT_I} (x_i - y_i)^p}.$$

We want to define interval-focused similarity such that we are not limited to one relevant interval. Rather, we want to be flexible to specify a set of relevant intervals $\mathcal{I}$ that is again specified at query time. Thus, the L_p-norm between X and Y w.r.t. $\mathcal{I}$ is defined by

$$L_p^{\mathcal{I}}(X, Y) = \sqrt[p]{\sum_{I \in \mathcal{I}} L_p^I(X, Y)^p}.$$

Note that the intervals $I \in \mathcal{I}$ can be of varying length and, thus, the influence of each interval on the complete sum may be different. In some applications, it may be interesting to weight the intervals, such that the contribution to the overall distance of each interval is similar. This can be easily achieved by multiplying a weighting factor w_I to each summand. In order to achieve similar influence of each interval I regardless of its length $|I|$, we can set $w_I = 1/|I|$.

Interval-focused distance range query: Given a query time series Q, a distance $\varepsilon \in \mathbb{R}$, and a relevant set of intervals $\mathcal{I}$, an *interval-focused distance range query* retrieves the set $\mathrm{DRQ}(Q, \varepsilon, \mathcal{I}) = \{X \in \mathcal{D} \mid L_p^{\mathcal{I}}(Q, X) \leq \varepsilon\}$.

Interval-focused k-nearest neighbor query: Given a query time series Q, a number $k \in \mathbb{N}$, and a relevant set of intervals $\mathcal{I}$, an *interval-focused k-nearest neighbor query* (kNN query) retrieves the set $\mathrm{NNQ}(Q, k, \mathcal{I}) \subseteq \mathcal{D}$ containing at least k time series such that
$\forall X \in \mathrm{NNQ}(Q, k, \mathcal{I}), \hat{X} \in \mathcal{D} - \mathrm{NNQ}(Q, k, \mathcal{I}) : L_p^{\mathcal{I}}(Q, X) \leq L_p^{\mathcal{I}}(Q, \hat{X})$.

In this paper, we claim the following contributions: After we have formalized the notion of interval-focused similarity queries, we describe a new efficient representation of time series based on interval boxes in the following. In addition,

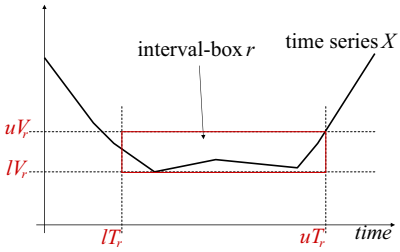

Fig. 2. Illustration of the interval box approximation of a given time series X.

we show how this representation can be used to efficiently support interval-focused similarity search using an existing index structure. The key benefit of our novel representation is that we only need to access those parts of the time series objects that are relevant for a given query. Furthermore, we define a lower and upper bound on the interval box representation for any L_p-norm, and describe an efficient multi-step filter/refinement architecture for interval-focused similarity queries.

4 Distance Approximation of Time Series Objects

The basic idea of our approach is to represent each time series object of the database by sequences of intervals. These intervals can be efficiently managed by an index such as the RI-tree [8]. In addition, if we store the maximum and minimum amplitude of the time series within the intervals, these intervals can be used to compute upper and lower bounds of the true distance between different time series. If an interval-focused similarity query is launched specifying a set of relevant time frames $\mathcal{I}$, only the intervals of the database objects that intersect any $I \in \mathcal{I}$ need to be accessed in order to estimate the lower and upper bounding distance approximations.

4.1 Representing Time Series Objects by Interval Boxes

We approximate each time series $X \in \mathcal{D}$ by a set of intervals. For each interval, we further store the maximum and minimum amplitude of X within the interval. This results in a minimum-bounding box around X within the specified interval (cf. Figure 2) called *interval box*. Formally, an interval box r is given by $r = (lT_r, uT_r, lV_r, uV_r)$, where (lT_r, uT_r) specifies the time interval, $lV_r = \min_{lT_r \leq i \leq uT_r} x_i$, and $uV_r = \max_{lT_r \leq i \leq uT_r} x_i$.

The set of interval boxes approximating X is denoted by $rep(X)$. We discuss methods for generating interval boxes for a given time series later in Section 5. So far, we claim no further constraints for the interval boxes $r \in rep(X)$ as far as $\forall i : lT_r \leq i \leq uT_r : lV_r \leq x_i \leq uV_r$.

4.2 Distance Estimation Using Interval Boxes

In the following, we will discuss how we can estimate the true distance between a query object Q and any $X \in \mathcal{D}$ by means of an upper and a lower bound using the information of $rep(X)$ rather than using the complete representation of X.

At each relevant time slot i, we can lower bound the i-th summand of the L_p-norm by the well-known $MINDIST$ between q_i and any interval box $r \in rep(X)$ that overlaps i, i.e. $lT_r \leq i \leq uT_r$. The $MINDIST$ between q_i and any interval box r with $lT_r \leq i \leq uT_r$ is defined as

$$MINDIST(q_i, r) = \begin{cases} lV_r - q_i & \text{if} \quad q_i \leq lV_r \\ q_i - uV_r & \text{if} \quad q_i \geq uV_r \\ 0 & \text{else.} \end{cases}$$

If we do not have any interval box $r \in rep(X)$ that overlaps time slot i, we can only lower bound the true distance between q_i and x_i by 0. If there are several interval boxes $r \in rep(X)$ with $lT_r \leq i \leq uT_r$, we aggregate the maximum over all the corresponding MINDIST values. Formally, at each time slot i, a lower bound of the i-th summand of the L_p-norm between Q and X is given by

$$LB^i(Q, X) = \max\{0, \max_{\{r \,|\, r \in rep(X), lT_r \leq i \leq uT_r\}} MINDIST(q_i, r)\}.$$

Obviously, $LB^i(Q, X) \leq |q_i - x_i|$. We can now extend the lower bound at each time slot i to intervals $I = (lT_I, uT_I)$ as follows:

$$LB^I(Q, X) = \sqrt[p]{\sum_{i=lT_I}^{uT_I} (LB^i(Q, X))^p}.$$

Still, the lower-bounding property $LB^I(Q, X) \leq L_p^I(Q, X)$ is preserved. A lower bound for a set of intervals $\mathcal{I} = \{I \,|\, i \in \mathbb{N}^+\}$ is then defined by

$$LB^{\mathcal{I}}(Q, X) = \sqrt[p]{\sum_{I \in \mathcal{I}} (LB^I(Q, X))^p}.$$

Again, we have the lower-bounding property$LB^{\mathcal{I}}(Q, X) \leq L_p^{\mathcal{I}}(Q, X)$.

Analogously, an upper bounding distance estimation can be determined. At each relevant time slot i, we now need to use the $MAXDIST$ between q_i and any interval box $r \in rep(X)$ that overlaps i, i.e. $lT_r \leq i \leq uT_r$, to define an upper bound of the i-th summand of $L_p(Q, X)$. The $MAXDIST$ between q_i and any interval box r with $lT_r \leq i \leq uT_r$ is defined as $MAXDIST(q_i, r) = \max\{|q_i - lV_r|, |q_i - uV_r|\}$.

If we do not have any interval box $r \in rep(X)$ that overlaps time slot i, we can upper bound the true distance between q_i and x_i by $\max\{|q_i - MAX|, |q_i - MIN|\}$. If there are several interval boxes $r \in rep(X)$ with $lT_r \leq i \leq uT_r$, we aggregate the minimum over all the MAXDIST values. Formally, at each time slot i, an upper bound of the i-th summand of the L_p-norm between Q and X

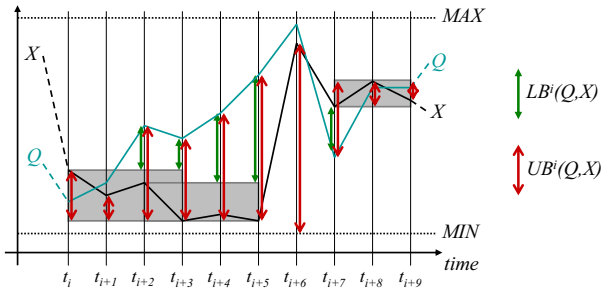

Fig. 3. Lower and upper bounding the L_p-distance within the interval (t_i, t_{i+9}).

is given by

$$UB^i(Q,X) = \min\{\max\{|q_i - MAX|, |q_i - MIN|\}, \min_{r \in rep(X), lT_r \le i \le uT_r} MAXDIST(q_i, r)\}.$$

Analogously, we define

$$UB^I(q_i, x_i) = \sqrt[p]{\sum_{i=lT_I}^{uT_I} (UB^i(Q,X))^p}$$

for time intervals I, and

$$UB^{\mathcal{I}}(Q,X) = \sqrt[p]{\sum_{I \in \mathcal{I}} (UB^I(Q,X))^p}.$$

for sets of time intervals $\mathcal{I}$. It is easy to prove that $UB^{\mathcal{I}}(Q,X) \ge L_p^{\mathcal{I}}(Q,X)$.

An example for the upper and lower bounding distance estimation is depicted in Figure 3. At time slot t_{i+6} we do not have any interval box representations of X. Thus, the bounds are estimated by $LB^{t_{i+6}}(Q,X) = 0$ and $UB^{t_{i+6}}(Q,X) = \max\{|q_{t_{i+6}} - MAX|, |q_{t_{i+6}} - MIN|\}$. On the other hand, at time slot t_{i+1} the interval box $r = (t_i, t_{i+3}, lV_r, uV_r) \in rep(X)$ is the only interval box that overlaps. We estimate $LB^{t_{i+1}}(Q,X) = MINDIST(q_{t_{i+1}}, r) = 0$ and $UB^{t_{i+1}}(Q,X) = MAXDIST(q_{t_{i+1}}, r) = |q_{t_{i+1}} - lV_r|$.

4.3 Query Processing

In order to compute the upper and lower bounding distance approximations between a query object Q and a database object $X \in \mathcal{D}$ efficiently, we need to determine those interval boxes that intersect the relevant intervals $I \in \mathcal{I}$. For the efficient support of intersection queries, we organize the intervals of the interval boxes in an adoption of the relational interval tree (RI-tree) [8]. An interval intersection query takes a query interval $I \in \mathcal{I}$ and retrieves all intervals

in the RI-tree that intersect with I. Details on the processing of intersection queries using RI-Trees can be found in [8]. In order to determine all interval boxes that intersect with the query intervals we need such an intersection query for all $I \in \mathcal{I}$. This way, we determine for each database object $X \in \mathcal{D}$ those interval boxes $r \in rep(X)$ that intersect with any of the query intervals $I \in \mathcal{I}$ in order to compute $LB^{\mathcal{I}}(Q, X)$ and $UB^{\mathcal{I}}(Q, X)$.

Based on our distance approximations LB and UB introduced above, we can apply the paradigm of filter/refinement query processing to efficiently answer interval-focused distance range and kNN queries. In case of an interval-focused distance range query, we can use both, the upper and the lower bound in the filter step. Each object $X \in \mathcal{D}$ with $LB^{\mathcal{I}}(Q, X) > \varepsilon$ can be identified as true drop because $L_p^{\mathcal{I}}(Q, X) \geq LB^{\mathcal{I}}(Q, X) > \varepsilon$, i.e. $X \notin \mathrm{DRQ}(Q, \varepsilon, \mathcal{I})$. On the other hand, each object $X \in \mathcal{D}$ with $UB^{\mathcal{I}}(Q, X) \leq \varepsilon$ can be identified as true hit since $L_p^{\mathcal{I}}(Q, X) \leq UB^{\mathcal{I}}(Q, X) \leq \varepsilon$, i.e. $X \in \mathrm{DRQ}(Q, \varepsilon, \mathcal{I})$. In case of an interval-focused kNN query, we can only use the lower bound for the filter step. We apply the approach presented in [10] which is optimal w.r.t. the number of candidates that need to be refined.

5 Generating Approximations

In this section, we will show how to generate adequate interval boxes for a time series. When building the interval boxes we need to take two contradicting considerations into account. On one hand, the number of boxes covering the time series should be low in order to avoid a dramatically increased overhead of the filter step. The performance of the filter step is mainly influenced by the number of interval box approximations to be considered at query time. More boxes lead to higher join cost of the query process. This suggests to construct wide boxes with long intervals. On the other hand, wide boxes will usually worsen the approximation quality since the boxes conservatively approximate the time series. As a consequence, the performance may decrease due to a reduced pruning power of the filter step. This suggests to construct boxes with low approximation error in order to achieve higher values for the lower bounding filter distance $LB^{\mathcal{I}}$ and lower values for the upper bounding filter distance $UB^{\mathcal{I}}$. Following these considerations, the parts of the time series having a flat curvature can be better approximated by interval boxes than parts featuring a high ascending or descending curve (cf. Figure 4 (upper part)). The basic idea of our approach is to optimize the box covering locally. We first identify those parts of the time series which can be well approximated, i.e. subsequences covering the local maximums or minimums of a time series. Then, we try to generate interval boxes that optimally cover the local minimums and maximums of a time series according to a quality criterion given below. Afterwards, we approximate each remaining part of the time series which are not covered yet by one single box.

A high approximation quality of the interval box approximations of a time series is responsible for a good pruning power of our filter step. A high lower bounding distance estimation allows to prune a lot of true drops without the need

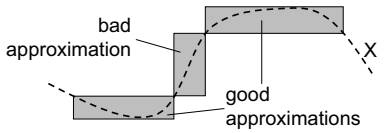

Fig. 4. Interval box approximations.

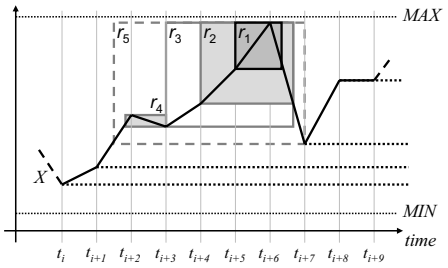

Fig. 5. Generation of covering boxes.

to refine them. A low upper bounding distance estimation enable to identify some of the true hits without any refinement. For this reason we propose to evaluate the approximation quality of an interval box by considering the expectation of the lower and upper bounding distance between any query object and the approximated part of the database object. For the sake of clarity and due to space limitations, we will focus on the expectation of the lower bound distance w.r.t. an interval box approximation. The expectation of the upper bound distance can be integrated analogously.

Given an interval box $r = (lT_r, uT_r, lV_r, uV_r)$, the expected lower bounding distance $LB^{(lT_r, uT_r)}$ between r and any query time series $Q = [q_1, .., q_N]$ which values q_i are assumed to be statistically independent can be computed as follows:

$$E(LB^{(lT_r,uT_r)}(Q,X)) = \sqrt[p]{uT_r - lT_r} \cdot E(LB^i(Q,X)),$$

where

$$E(LB^i(Q,X)) = \int_{MIN}^{MAX} MINDIST(q_i, r) f_i(q_i) = \frac{(MAX - uV_r)^2 + (lV_r - MIN)^2}{2 \cdot (MAX - MIN)}$$

is the expected lower bounding distance according to any time slot $lT_r \leq t_i \leq uT_r$ and $f_i(q_i)$ is the probability density function of the event time series value $q_i \in [MIN, MAX]$. Thereby, we assume that the values of Q are equally distributed between MIN and MAX, i.e.

$$f_i(q_i) = \frac{1}{MAX - MIN}, \forall i \in [MIN, MAX].$$

Now, we can use the expectation of the distance estimations in order to decide for an interval box whether the box setting is more promising than alternative box settings. The higher the expected lower bounding distance w.r.t. an interval box approximation, the higher is its approximation quality.

Next, we will show how interval boxes covering the local extreme values of a time series can be generated nearly optimal according to our quality score. As already mentioned, flat parts, like the local maximums or minimums, of a time series are very adequate for our interval box approximation. We start with the approximation of the local maximums of a time series by searching for each local

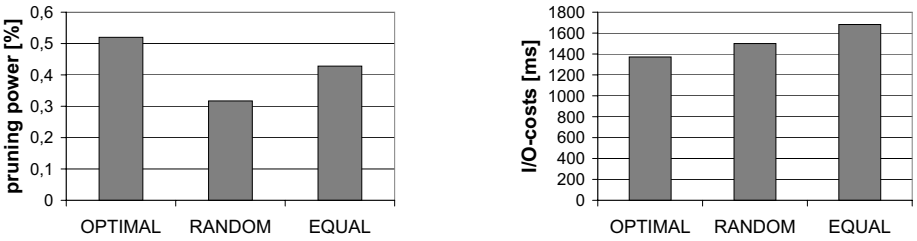

Fig. 6. Evaluation of different interval box generation methods.

maximum iteratively in top-down direction. For each local maximum we take all reasonable conservative coverings into account as shown in the example depicted in Figure 5, and try to pick out the best one. Those interval box candidates which cover or are covered by another interval box candidate are evaluated against each other according to our quality score. The candidate with the highest score is chosen for the approximation, the other candidates will be discarded. This procedure will be applied to all local maximums, so that, finally all local maximums are covered by any interval box. Redundant coverings are removed according to our quality score.

The coverings of the local minimums are generated in the same way. Contrary, this time we start at the local minimums and search the corresponding interval box candidates upwards. After generating all local maximum and minimum coverings, we remove those box candidates which are completely covered by another interval box candidate in order to reduce redundant approximations.

Finally in a post-processing step, the remaining gaps between two adjacent but disjunctive interval boxes, i.e. the parts of the time series which are not covered so far by any interval box, are simply approximated by an additional minimal bounding box. The overall covering of a time series can be performed in $O(N)$ time where N denotes the length of the time series.

6 Evaluation

All experiments were performed on a workstation featuring a 1.8 GHz Opteron CPU and 8GB RAM. We used a disk with a transfer rate of 60 MB/s, a seek time of 3 ms, a latency delay of 2 ms, and a cache allocating 80 KByte. The node capacity of the RI-tree was set to 8 KByte. For each experiment, we launched 100 sample queries and averaged the performance. In each query, we choose the relevant intervals randomly such that the sum of the length of each relevant interval equals the desired query focus size.

We first evaluate our method for generating interval box representations in comparison to two naive solutions on a synthetic data set featuring 400 time series of length 6,000. The first competitor ("RANDOM") determines a fixed number of intervals randomly and generates a minimum bounding box for each of these intervals. The second competitor ("EQUAL") works analogously but generates a fixed number of intervals with equal length. The results are shown

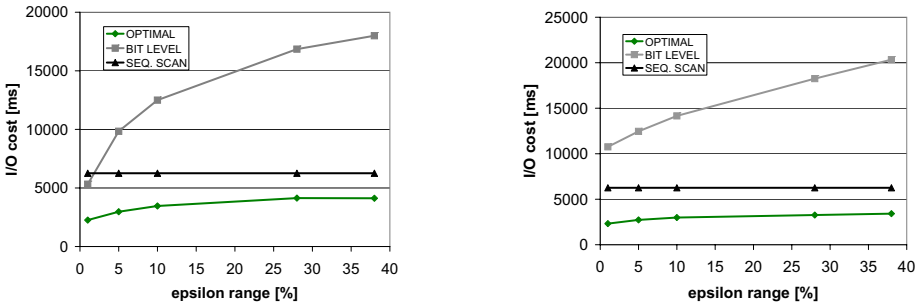

Fig. 7. Performance w.r.t. the selectivity of the query. DS1(left) and DS2 (right).

in Figure 6. As it can be seen, our method ("OPTIMAL") outperforms both competitors in terms of pruning power and I/O cost. This empirically shows that our interval box generation is superior to the two other naive solutions.

Secondly, we evaluate our proposed filter/refinement architecture (OPTIMAL) for answering interval-focused similarity queries compared to the sequential scan (SEQ. SCAN) and the approach proposed in [9] (BIT LEVEL). We choose the second competitor since it is the only approach that does not need to scan the entire time series information for answering interval-focused queries but also proposes a filter/refinement architecture based on a compressed data representation. We used two real-world data sets, "DS1" and "DS2" each featuring 4,800 song-feature time series of length 10,000. The performance of the competitors w.r.t. the selectivity of the query is visualized in Figure 7. The focus size was set to 1% of the time series length. Our approach clearly outperforms both competitors for all settings of the query selectivity. Furthermore, in contrast to "BIT LEVEL" our approach scales well even for large query result sets. The performance of the competitors w.r.t. the size of the query focus is depicted in Figure 8. In this experiment we performed queries featuring a query selectivity of 2% of the dataset. For small focus sizes ($< 6\%$ of the time series length) our approach achieved smaller I/O cost than the competing techniques. However, in many applications using interval focused similarity search a focus size smaller

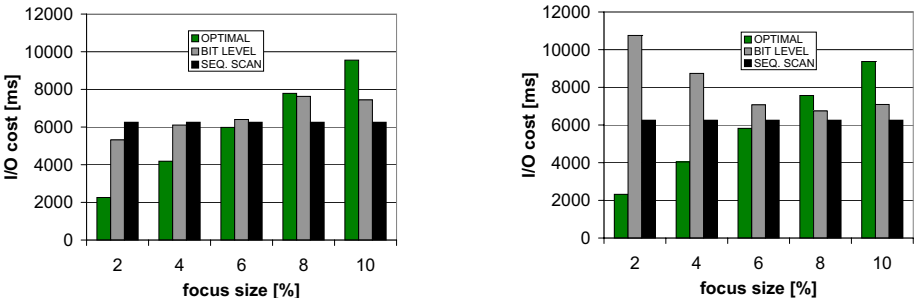

Fig. 8. Performance w.r.t. the size of the query focus. DS1(left) and DS2 (right).

than 5% is reasonable. One can imagine a query on one year records focusing only one certain weak which would correspond to a focus size of about 2%.

7 Conclusions

In this paper, we introduce and formalize the novel concept of interval-focused similarity queries in time series databases which is an important generalization of comparing entire time series. We describe a new efficient representation of time series based on intervals and show how this representation can be used to efficiently support these new query type implementing a filter/refinement approach. Furthermore, we present a method for the generation of the interval-based representation. In our experimental evaluation we show the superiority of our proposed method for answering interval-focused similarity queries in comparison to existing approaches.

References

1. R. Agrawal, C. Faloutsos, and A. Swami. "Efficient Similarity Search in Sequence Databases". In *Proc. 4th Conf. on Foundations of Data Organization and Algorithms*, 1993.
2. O. Alter, P. Brown, and D. Botstein. "Generalized Singular Value Decomposition for Comparative Analysis of Genome-Scale Expression Data Sets of two Different Organisms". *Proc. Natl. Aca. Sci. USA*, 100:3351–3356, 2003.
3. Y. Cai and R. Ng. "Index Spatio-Temporal Trajectories with Chebyshev Polynomials". In *Proc. ACM SIGMOD*, 2004.
4. K. Chan and W. Fu. "Efficient Time Series Matching by Wavelets". In *Proc. IEEE ICDE*, 1999.
5. C. Faloutsos, M. Ranganathan, and Y. Maolopoulos. "Fast Subsequence Matching in Time-series Databases". In *Proc. ACM SIGMOD*, 1994.
6. E. Keogh, K. Chakrabati, S. Mehrotra, and M. Pazzani. "Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases". In *Proc. ACM SIGMOD*, 2001.
7. F. Korn, H. Jagadish, and C. Faloutsos. "Efficiently Supporting Ad Hoc Queries in Large Datasets of Time Sequences". In *Proc. ACM SIGMOD*, 1997.
8. H.-P. Kriegel, M. Pötke, and T. Seidl. "Interval Sequences: An Object-Relational Approach to Manage Spatial Data". In *Proc. SSTD*, 2001.
9. C. A. Ratanamahatana, E. Keogh, A. J. Bagnall, and S. Lonardi. "A Novel Bit Level Time Series Representation with Implication for Similarity Search and Clustering". In *Proc. PAKDD*, 2005.
10. T. Seidl and Kriegel H.-P. "Optimal Multi-Step k-Nearest Neighbor Search". In *Proc. ACM SIGMOD*, 1998.
11. S. Wichert, K. Fokianos, and K. Strimmer. "Identifying Periodically Expressed Transcripts in Microarray Time Series Data". *Bioinformatics*, 20(1):5–20, 2004.
12. B. K. Yi and C. Faloutsos. "Fast Time Sequence Indexing for Arbitrary Lp Norms". In *Proc. VLDB*, 2000.