Efficient Similarity Retrieval in Music Databases

Maria M. Ruxanda Christian S. Jensen

Department of Computer Science
Aalborg University
Denmark
{mmr,csj}@cs.aau.dk

Abstract

Audio music is increasingly becoming available in digital form, and the digital music collections of individuals continue to grow. Addressing the need for effective means of retrieving music from such collections, this paper proposes new techniques for content-based similarity search. Each music object is modeled as a time sequence of high-dimensional feature vectors, and dynamic time warping (DTW) is used as the similarity measure. To accomplish this, the paper extends techniques for time-series-length reduction and lower bounding of DTW distance to the multi-dimensional case. Further, the Vector Approximation file is adapted to the indexing of time sequences and to use a lower bound on the DTW distance. Using these techniques, the paper exploits the lack of a ground truth for queries to efficiently compute query results that differ only slightly from results that may be more accurate, but also are much more expensive, to compute. In particular, the paper demonstrates that aggressive use of time-series length reduction together with query expansion results in significant performance improvements while providing good, approximative query results.

1 Introduction

Radio broadcasting has entered the digital age, and the record companies are selling digital music on-line. As a consequence, digital audio music is more and more widespread, and the personal collections of music stored on MP3 players, PCs, and media centers grow to thousands of songs. This development calls for efficient data management techniques for digital music databases, which includes techniques for music information retrieval.

Techniques for music similarity retrieval can be grouped into three main categories. The first is the so-called metadata similarity retrieval, where different metadata are created manually (e.g., as done by Pandora [22]) or obtained automatically (e.g., as done by All Music Guide [2]) from the music and used for querying. Manual creation and update of metadata lacks in "scalability," and the automatic approach is limited to what has been provided manually by others.

The second category of techniques uses collaborative filtering (e.g., as done by Amazon [1]). Here, past user behavior is exploited. For example, each time a person listens to or purchases a piece of music, this can be recorded. It then becomes possible to offer guidance akin to "those who liked this music also like this other music." One advantage of collaborative filtering is that it scales well to large databases. However, it does not work well when new music is introduced.

The last category of techniques, which is the subject of this paper, is content-based retrieval. Here, the audio signal itself is processed and features are extracted and used for querying. These techniques are automatic, scale well to large music collections, and can be integrated with the other two categories of techniques. However, the length and high dimensionality of the extracted features offer challenges when it comes to the efficient processing of queries.

Content-based retrieval consists of two general steps. The first step concerns feature extraction from the music signals. This paper utilizes state-of-the-art music feature vectors, but does not study feature extraction. Feature extraction is an active area of research within the audio signal processing field. Music feature vectors can be quite high-dimensional. For instance, the popular AR coefficients [21] that are being used successfully for genre classification can be $\approx$100 dimensional. A song is usually represented by a time sequence of such feature vectors, one for each short time-frame of the song. The second step in content-based retrieval concerns the storage and indexing of feature vectors, and their subsequent use in querying.

We represent a music object in the database as a time sequence of high-dimensional feature vectors. In this setting, we aim at providing techniques for performing efficient similarity retrieval. Efficiency implies

International Conference on Management of Data
COMAD 2006, Delhi, India, December 14–16, 2006

two aspects: applying a similarity measure that is as meaningful and intuitive as possible from the point of view of the users, and retrieving the result of a query with low I/O and CPU costs relative to the size of the database. In this paper, we address both aspects. We are not aware of other approaches that represent music as multi-dimensional time series.

The database community has studied the problem of similarity querying for time series databases (e.g., [3,9,15,16,20]). Using the Euclidean distance for time series results in variations in time shifting and time scaling having an overly large effect. As a framework for absorbing such variation with smaller effect, dynamic time warping (DTW) distance has been proposed. This notion of distance allows very good matching of similar subsequences that are shifted within a limited range along the time dimension. Thus, dynamic time warping has been used in speech recognition [12, 25] as well as in fields such as bioinformatics [4], video-data management [10], fingerprint matching [18], and the classification of handwritten text [24].

Indexing of data when using DTW is challenging, as DTW distance is not a metric and moreover is expensive to compute for large time series. Several techniques have been proposed for indexing one-dimensional time-series under DTW [14, 17, 30, 31]. These propose different lower bounding functions for DTW, and Keogh [14] and Zhu and Shasha [31] also study the idea of time-series length reduction. Vlachos et al. [26] investigate DTW indexing for multi-dimensional time series. Their method works with 2D trajectories, which are split in Minimum Bounding Rectangles and stored in an R-tree [11]. However, when the dimensionality of the space is ≈ 100, using a hierarchical tree structure is more expensive than linear search.

In this paper, we extend known DTW indexing techniques to work with high-dimensional time series. We approximate the original time series by reduced-length time series. Due to the high dimensionality of the feature vectors, hierarchical index structures are inappropriate. We instead index them using the (Vector Approximation (VA) file [27]. We adapt the VA-file to support ϵ range queries for equal-length time sequences when using a lower bound on the DTW distance as the distance measure. The highly subjective nature of music similarity means that a ground truth for queries does not exist. We propose techniques that exploit this, by efficiently computing query results that are similar to results that are significantly more expensive to compute. Results of an empirical performance study are reported that demonstrate that relatively aggressive use of time series length reduction together with query enlargement yields significantly improved query performance while not affecting the query results substantially.

The rest of the paper is organized as follows. Section 2 presents background information and related work. Section 3 reviews the lower bounding of DTW for one-dimensional time series and introduces a lower bounding function for multi-dimensional time series. Section 4 covers time series length reduction, the indexing of multi-dimensional time series, and a new range search algorithm for the VA-file. Finally, experimental results are presented in Section 5, and Section 6 concludes the paper.

2 Background

The similarity measure used for similarity retrieval is a very important consideration. Dynamic time warping (DTW) constitutes a flexible and promising framework for music similarity. DTW has been used in speech processing for several decades [12, 25], and it was introduced to the database community in 1994 by Berndt and Clifford [6], who showed how to speed up DTW computation using dynamic programming and also demonstrated its applicability as a time series similarity measure. However, not being a metric means that DTW is not directly applicable in the context of large databases. Computing DTW distances for every possible query result is not feasible when working in a very large database.

2.1 DTW Computation

We briefly review the definition of the DTW distance in the context of multi-dimensional time series.

Let Q and C be two d-dimensional time series of length n and m, respectively:

$$\begin{array}{ll} Q = (Q_1, ..., Q_n) & Q_i = (q_{i1}, ..., q_{id}) \\ C = (C_1, ..., C_m) & C_j = (c_{j1}, ..., c_{jd}) \end{array}$$

To align two sequences using DTW, a n-by-m matrix is constructed where cell (i, j) contains the distance $d(Q_i, C_j)$ (where d is, e.g., the Euclidean distance) between elements Q_i and C_j. A warping path W proceeds from cell $(1, 1)$ to cell (n, m), and it represents an alignment between pairs of elements from Q and C.

$$W = w_1, w_2, ..., w_k, ..., w_K, \max(m, n) \leq K < m+n-1$$

Thus, $w_1 = (1, 1)$ and $w_K = (n, m)$. A warping path W has two more additional properties. First, W must be continuous: Given $w_k = (a, b)$ then $w_{k-1} = (a', b')$, where $a - a' \leq 1$ and $b - b' \leq 1$. This restricts the allowable steps in the warping path to adjacent cells. Second, W must be monotonic: Given $w_k = (a, b)$, then $w_{k-1} = (a', b')$, where $a - a' \geq 0$ and $b - b' \geq 0$. This forces the points in W to be monotonically spaced in time.

The cardinality of the set of possible warping path $\mathcal{W}$ grows exponentially with the lengths of the time series. The DTW distance is the minimum over all paths

in $\mathcal{W}$ of the squared root of the sum of the elements along the path.

Definition 1 [14]

$DTW(Q,C) = \min_{W \in \mathcal{W}} \sqrt{\sum_{k=1}^{K} w_k}$

Figure 1 exemplifies DTW computation for the time series:

$$Q = -12, 10, 5, 1, 3, -2, -1, 4, 7, -3$$
$$C = 21, 2, 4, 11, 6, 11, -4, -5, 1, 6$$

The DTW distance for one-dimensional time series can be computed using dynamic programming in time $O(nm)$ [6].

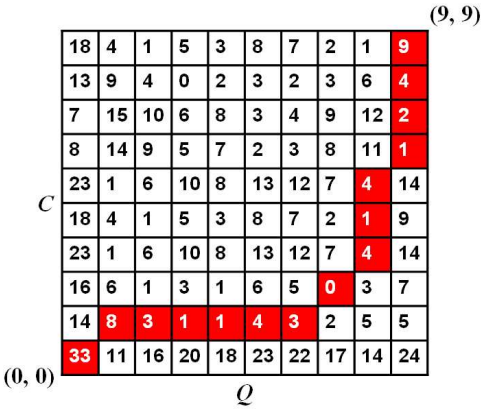


Figure 1: DTW Computation—$DTW(Q,C) = \sqrt{78}$

Constraints may be imposed on warping paths that limit how far a path may stray from the diagonal, thus avoiding pathological paths and also speeding up the computation [23]. For example, for our music time series database, we want to avoid mapping the same feature vector from one song to a large number of feature vectors from another song.

Two well-known constraints are the Sakoe-Chiba band [25] and the Itakura parallelogram [12](see Figure 2). Figure 3 shows DTW computation for the same

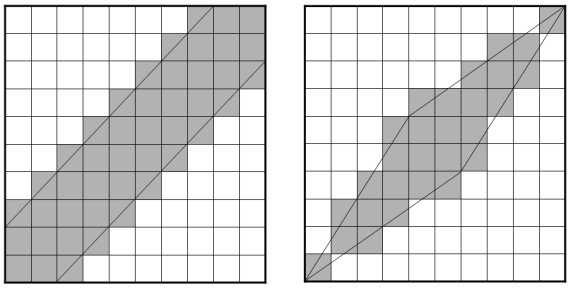


Figure 2: Constrained DTW

time series as in Figure 1, but with a Sakoe-Chiba band constraint. In this case, we obtain $DTW(Q,C) = \sqrt{80}$. The constraint is defined by a warping range $r = 2$, which produces a band of width 5 (i.e., $2r + 1$) along the diagonal.

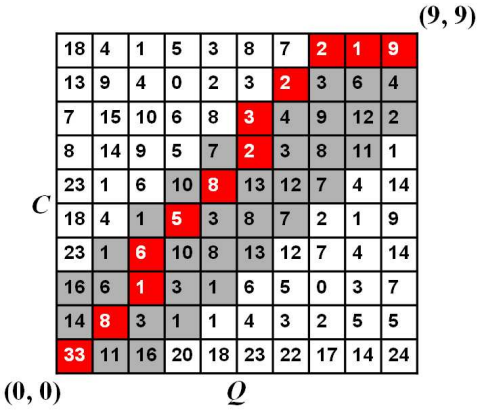


Figure 3: Constrained DTW Computation—$DTW(Q,C) = \sqrt{80}$

2.2 Related Work

Yi et al. [30] are the first to investigate the use of DTW in large databases. They propose an approximative indexing of DTW using FastMap [8]. The idea is to embed the time sequences into an Euclidean space that approximately preserves the distances between them, and then use a classical multi-dimensional index structure like the R*-tree [5]. They also introduce a lower bound for filtering out unlikely matches that are inevitably introduced. The method outperforms sequential scan, but the time to build the index grows drastically with the length of the time series.

Kim et al. [17] are the first to introduce an exact algorithm for indexing time series under DTW. They extract four features from the time sequences and index them in a multi-dimensional index structure. In addition, they propose another lower bounding function for DTW, which is defined on the four features and guarantees no false negatives. However, the method allows the extraction of exactly four features, while actually only one of them, determined at query time, is used in the lower bound. Therefore, the technique generates many false positives that need to be refined using expensive DTW distance computations.

Keogh [14] proposes a technique for exact indexing of DTW that guarantees no false negatives. The technique can be applied only for equal length time series and DTW computation where the warping path is restricted (e.g., as shown in Figure 2). The core idea is to define the envelope of a time series using the term that defines the range of the warping path and apply a new lower bound function computed on the basis of the envelope of time series. Indexing of DTW is achieved by performing a length reduction on both the time series and the envelope using Picewise Aggre-

gate Approximation (PAA). The reduced length time series are indexed using the R*-tree. Experimental results show that the pruning power of this lower bound is 3–4 times better than that of Yi's lower bound and 6 times better than that of Kim's lower bound.

In follow-on work, Zhu and Shasha [31] have improved on the technique for time series length reduction using PAA. They propose a new definition of the envelope of a reduced time series, which is tighter that Keogh's envelope. This yields a tighter lower bound distance for DTW, which is shown experimentally to be 3 times tighter than Keogh's lower bound.

All the above DTW indexing techniques have been designed and studied for only one-dimensional time series. In this paper, we extend the DTW lower bound function and the reduction of length of the time-series of Keogh [14], and the reduction of length of the envelope using the PAA technique of Zhu and Shasha [31], to the multidimensional case.

Lee et al. [19] and Kahveci et al. [13] study multidimensional time series, but consider only Euclidean distance. More recently, Vlachos et al. [26] have investigated indexing of multi-dimensional time series under DTW. They consider only the case of 2D trajectories of moving objects. Their main idea is to split the trajectories into MBRs and store them in an R-tree. For a given query, a Minimum Bounding Envelope (MBE) is constructed that covers all possible matching areas of the query under warping conditions. The MBE is decomposed into MBRs, and the R-tree is probed, thus finding the potential candidates for the query result.

Reducing the length of a time series does not reduce its dimensionality. It is known that hierarchical index structures cease to perform better than a linear scan when the dimensionality of the space exceeds ≈ 20 [27]. As we are faced with a dimensionality of ≈ 100, we build on the VA-file [27–29], which is known to still perform better than linear scan when the dimensionality of the space exceeds 20. Studies of dimensionality reduction techniques is an interesting, but orthogonal issue. The techniques we have considered, when reducing the dimensionality from ≈ 100 dimensions to 20, cause substantial information loss and lead to a refinement step that is not competitive.

We adapt the VA-file to index the reduced time series and use as the distance measure the lower bounding function of DTW. We define a new search algorithm over the VA-file that performs efficient range queries when indexing reduced length time series.

3 Lower Bounding the DTW

DTW computation is expensive, especially for long time series. One solution is to apply the filter-and-refinement paradigm, thus using an inexpensive lower bounding function for filtering out the music objects that may be good match, performing afterwards a refinement over the selected candidates using the DTW. A good lower bound function needs to fulfill two requirements: it must be inexpensive to compute, and it must be a relatively tight lower bound. The second requirement ensures that relatively few candidates are left for the refinement.

3.1 One-Dimensional Lower Bounding

The best known lower bounding function [14] is defined for one-dimensional time series and holds for the case where the argument time series have the same length and the warping paths are constrained. Further, the distance for each element in a warping path is defined to be the squared Euclidean distance, and the dynamic time warping distance is the square root of the sum of these.

As described in Section 2.1, the indices of the cells in a warping path are constrained: given a cell $w_k = (i,j)_k$ then $j - r \leq i \leq j + r$, where r defines the warping range. Both the Sakoe-Chiba band and the Itakura parallelogram fall under this definition, while for the Sakoe-Chiba band r is independent of i and for the Itakura parallelogram r is dependent on i. Using the warping range r, the envelope of a time series Q has been defined as follows.

Definition 2 **[14]** The envelope of one-dimensional time series $Q = (q_1, ..., q_n)$, denoted $Env(Q)$ is a pair of time series $L = (l_1, ..., l_n)$ and $U = (u_1, ..., u_n)$:

$$l_i = \min(q_{i-r} : q_{i+r}) \qquad u_i = \max(q_{i-r} : q_{i+r})$$

A lower bounding measure for $DTW(Q, C)$ for one-dimensional time series is:

$$LB(Env(Q), C) = \sqrt{\sum_{i=1}^{n} \begin{cases} (c_i - u_i)^2 & \text{if } c_i > u_i \\ (c_i - l_i)^2 & \text{if } c_i < l_i \\ 0 & \text{otherwise} \end{cases}}$$

Lemma 1 **[14]** For two time series Q and C of length n, for a constraint on the warping paths of the form $j - r \leq i \leq j + r$, the following inequality holds:

$$LB(Env(Q), C) \leq DTW(Q, C)$$

3.2 Extension to Multiple Dimensions

We proceed to extend the lower bounding function definition to multi-dimensional time series. We assume that the time series have the same length and that the warping paths are constrained.

Let $Q = (Q_1, ..., Q_n)$ where $Q_i = (q_{i1}, ..., q_{id})$ be a d-dimensional time series.

Definition 3 The envelope of d-dimensional time series Q, denoted $Env(Q)$, is a pair of d-dimensional time series $L = (L_1, ..., L_n)$ and $U = (U_1, ..., U_n)$ where

$$\begin{aligned} L_i &= (l_{i1}, ..., l_{id}), \quad l_{ij} = \min(q_{(i-r)j} : q_{(i+r)j}) \\ U_i &= (u_{i1}, ..., u_{id}), \quad u_{ij} = \max(q_{(i-r)j} : q_{(i+r)j}) \end{aligned}$$

A lower bounding measure for $DTW(Q,C)$ is:

$$LB_{\mathrm{d-dim}}(Env(Q),C) = \sqrt{\sum_{i=1}^{n} d(Env(Q_i),C_i)} \text{ where}$$

$$d(Env(Q_i),C_i) = \sum_{j=1}^{d} \begin{cases} (c_{ij}-u_{ij})^2 & \text{if } c_{ij} > u_{ij} \\ (c_{ij}-l_{ij})^2 & \text{if } c_{ij} < l_{ij} \\ 0 & \text{otherwise} \end{cases}$$

Lemma 2 For two d-dimensional time sequences Q and C of length n and a constraint on the warping paths of the form $j-r \leq i \leq j+r$, the following inequality holds:

$$LB_{\mathrm{d-dim}}(Env(Q),C) \leq DTW(Q,C)$$

Proof: See the appendix.

4 Indexing DTW for Multi-Dimensional Time Series

A lower bounding function for DTW is used for eliminating, at low cost, time series that cannot possibly satisfy the query. However, although the lower-bound computations are inexpensive, using these in conjunction with a linear scan of all time series still results in substantial I/O. To scale up to large databases, we must avoid a linear scan. However, the effective indexing of long and high-dimensional time series is difficult.

The GEMINI framework [9] for length reduction of time series has been applied successfully to one-dimensional time series. In this framework, a one-dimensional time series can be reduced so much that it can be indexed using a structure such as an R*-tree. The approach generates false positives, so subsequent refinement is necessary. To avoid false negatives, the length reduction transform must be lower bounding the DTW so that the distance between any two reduced length time series is not exceeding their original distance.

4.1 Time Series Length Reduction

The method used in [14] and [31] to perform length reduction for 1-dimensional time series is Piecewise Aggregate Approximation (PAA).

Definition 4 **[14]** The PAA of the one-dimensional time series $C = (c_1,...,c_n)$ is $\bar{C} = (\bar{c}_1,...,\bar{c}_N)$ where $1 \leq N \leq n$ and:

$$\bar{c}_i = \frac{N}{n} \sum_{p=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} c_p$$

The PAA reduction of the envelopes using the method in [31] is as follows.

Definition 5 **[31]** The PAA of the envelope of one-dimensional time series C, denoted $Env(\bar{C})$ is:

$$\bar{l}_i = \frac{N}{n} \sum_{p=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} l_p, \quad \bar{u}_i = \frac{N}{n} \sum_{p=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} u_p$$

where $i = 1,2,...,N$.

Lemma 3 **[31]** With the preceding definitions and with C and Q being equal-length, one-dimensional time series, the following holds:

$$LB(Env(\bar{Q}),\bar{C}) \leq DTW(Q,C)$$

We proceed to extend the above definitions to apply to multi-dimensional time series. Thus, let $C = (C_1,...,C_n)$, where $C_i = (c_{i1},...,c_{id})$ be a d-dimensional time series.

Definition 6 The PAA of the d-dimensional time series C is $\bar{C} = (\bar{C}_1,...,\bar{C}_N)$ where $\bar{C}_i = (\bar{c}_{i1},...,\bar{c}_{id})$ and

$$\bar{c}_{ij} = \frac{N}{n} \sum_{p=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} c_{pj}$$

with $1 \leq N \leq n$.

Definition 7 The PAA of the envelope of the d-dimensional time series Q, denoted $Env(\bar{Q})$ is the pair $\bar{L} = (\bar{L}_1,...,\bar{L}_N)$ and $\bar{U} = (\bar{U}_1,...,\bar{U}_N)$, where $\bar{L}_i = (\bar{l}_{i1},...,\bar{l}_{id})$, $\bar{U}_i = (\bar{u}_{i1},...,\bar{u}_{id})$ and:

$$\bar{l}_{ij} = \frac{N}{n} \sum_{p=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} l_{pj}, \quad \bar{u}_{ij} = \frac{N}{n} \sum_{p=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} u_{pj}$$

Lemma 4 With the preceding definitions and with C and Q being equal-length, d-dimensional time series, the following holds:

$$LB_{\mathrm{d-dim}}(Env(\bar{Q}),\bar{C}) \leq DTW(Q,C)$$

Proof: The proof is very similar to that of Lemma 3 [31], the extension to the d-dimensional case being straightforward.

4.2 Similarity Range Queries

We define a similarity range query as follows.

Definition 8 Assume a set $\mathcal{DB}$ of d-dimensional time series each of length n. The similarity ϵ-range query for a given d-dimensional time series Q of length n, $\sigma[\epsilon,Q](\mathcal{DB})$, returns all time series $C \in \mathcal{DB}$ for which $DTW(Q,C) \leq \epsilon$.

The approach to computing this query efficiently consists of reducing the length of the time series, indexing the resulting time series, using this structure for identifying the time series that satisfy $LB_{\mathrm{d-dim}}(Env(\bar{Q}), \bar{C}) \leq \epsilon'$, and finally performing refinement on the resulting candidates according to the true DTW distances. Because distances in the reduced space are much smaller than distances in the original space, an $\epsilon' < \epsilon$ is used instead of ϵ.

As music similarity is highly subjective, there is no ground truth for similarity queries in music databases, such as the one considered here. This presents an opportunity to exploit the possibility of providing to the user with an approximative query result, trading query accuracy for better performance without the lower accuracy being noticeable by the user.

As we shall see, time series length reduction significantly improves query performance. Intuitively, as time series length is reduced further and further, an increasingly large ϵ' value must be used to ensure that query results remain reasonably accurate. It turns out that the performance is relatively insensitive to increases in ϵ'.

4.3 Indexing Multi-Dimensional Time Series Using the VA-File

As we use the VA-file, we first briefly review the core idea underlying this technique. The VA-file [27] divides the multi-dimensional space into 2^b cells, allocates an unique bit-string of length b for each cell, and approximates all data points that fall into a cell by this bit-string. The number b is calculated as $b = \sum_{i=1}^{d} b_i$ where d is the dimensionality of the space and b_i is the number of bits assigned per dimension i. The VA-file can simply be seen as an array of these compact approximations.

Initially designed to be used for nearest neighbor queries, the VA-file can also be used for range queries. A range query on the VA-file is performed in a filter step and a refinement step. The filter step avoids reading the actual data, but scans the entire approximation file. For each approximation, a lower bound on the distance to the query is determined based on the boundary points of the rectangular cells used by the approximation. Further, candidates are selected only if this lower bound is within the query range. The refinement step involves calculating the actual distance between the query object and each selected candidate, thus identifying the final result.

To be able to index time series with the VA-file, we perform several adjustments to the VA-file. The VA-file works with multi-dimensional points, so when indexing time series, we must index the individual multi-dimensional points of these. The number of objects indexed is thus N times the number of time series, where N is the length of a reduced-length time series. The VA-file cell boundaries in each dimension are computed based on these objects.

To preserve the notion of time series and to be able to efficiently retrieve the time series that qualify as potential candidates, we perform the following. The points that make up a time series are stored consecutively in the VA-file, thus storing a time series compactly. As all indexed time series have equal length N, when scanning the VA-file during filtering, every consecutive group of N approximations forms a time series. Thus, the first approximation in a group has to be compared with the first element of the query time series, etc. until the Nth element is reached.

The VA-file has been designed to be used only with L_p metric distances. Instead, we use the DTW lower bounding distance, which can be viewed as a modified Euclidean distance, but is not a metric. However, as will be shown in Section 5, the VA-file retrieval precision is not affected by this change.

For improved performance, an approximation of the similarity ϵ-range query described in Definition 8 is computed in several steps: the time series are length reduced and indexed in the VA-file, an ϵ'-range search over the VA-file with $\epsilon' < \epsilon$ is performed, and the potential candidates returned by the VA-file search are refined by computing the actual DTW distances for the original time series.

We substantiate by exhaustive experiments that time series length reduction and appropriately chosen values for ϵ' achieve improved query performance, at the cost of approximate query results. The larger ϵ' becomes, the more precise the answers become, but the computational cost also increases. We study this trade-off in Section 5. Straightforward choice for the range to be used in the reduced-length space may be $\epsilon' = \frac{N}{n}\epsilon$. However, we shall see in Section 5 that more precise results are obtained when using values for ϵ' that are larger. In this paper, the value of ϵ' is obtained experimentally, and it is left for future work to provide analytical means of selecting a good ϵ'.

Let us denote by $LVA(\bar{Q}, \bar{C})$ the VA-file lower bound of the distance $LB_{\mathrm{d-dim}}(Env(\bar{Q}), \bar{C})$, where $\bar{Q}$ and $\bar{C}$ are d-dimensional time series of reduced length N and $LVA(\bar{Q}, \bar{C}) \leq LB_{\mathrm{d-dim}}(Env(\bar{Q}), \bar{C})$. An ϵ'-range query over the VA-file for a given $\bar{Q}$ is performed by first selecting the time series $\bar{C}$ for which $LVA(\bar{Q}, \bar{C}) \leq \epsilon'$, and then by further selecting only the time series for which $LB_{\mathrm{d-dim}}(Env(\bar{Q}), \bar{C}) \leq \epsilon'$.

We define $LVA(\bar{Q}, \bar{C})$ as follows. Let $\bar{Q}_i = (\bar{q}_{i1}, ..., \bar{q}_{id})$ be the current query element, $Env(\bar{Q}_i)$ its envelope, and $\bar{C}_i = (\bar{c}_{i1}, ..., \bar{c}_{id})$ the current approximation that is scanned. We denote by $r_{\bar{q},j}$ the partition in which falls $\bar{q}_{ij}$ in dimension j and by $r_{\bar{c},j}$ the partition in which falls $\bar{c}_{ij}$ in dimension j. Any partition $r_{\bar{c},j}$ is represented by lower and upper boundary values, denoted $lVal[r_{\bar{c},j}]$ and $uVal[r_{\bar{c},j}]$.

Definition 9 The VA-file lower bound of the distance between the current scanned element and the current

query element, denoted by $LVA_i(\bar{Q}_i, \bar{C}_i)$ is:

$LVA_i(\bar{Q}_i, \bar{C}_i) = \sum_{j=1}^{d} {lb_j}^2$ where

$$lb_j = \begin{cases} LB(Env(\bar{Q}_i)_j, uVal[r_{\bar{c},j}]) & \text{if } r_{\bar{c},j} < r_{\bar{q},j} \\ LB(Env(\bar{Q}_i)_j, lVal[r_{\bar{c},j}]) & \text{if } r_{\bar{c},j} > r_{\bar{q},j} \\ 0 & \text{if } r_{\bar{c},j} = r_{\bar{q},j} \end{cases}$$

Definition 10 The VA-file lower bound of the distance $LB_{\mathrm{d-dim}}(Env(\bar{Q}), \bar{C})$ is:

$$LVA(\bar{Q}, \bar{C}) = \sqrt{\sum_{i=1}^{N} LVA_i(\bar{Q}_i, \bar{C}_i)}$$

Algorithm 1 efficiently computes an approximation of the similarity ϵ-range query for equal-length d-dimensional time series using the VA-file with time series length reduction.

The algorithm reconstructs the time series from their constituent elements in a cost effective manner. Whenever in the currently scanned time series, the *LVA* lower bound of the current element is larger than ${\epsilon'}^2$ or the part of the time series that has already been scanned generates an *LVA* distance larger than ϵ', the rest of the time series is skipped, thus avoiding useless computations.

Finally, we can describe all steps involved in performing a similarity range query in a database of d-dimensional time series using dynamic time warping distance as the similarity measure.

1. For all time series in the database, compute the reduced-length time series and store them on disk.
2. Build a VA-file for the reduced-length time series.
3. For the query time series Q, compute its reduced-length time series $\bar{Q}$ and the envelopes $Env(Q)$ and $Env(\bar{Q})$ of these two time series.
4. Perform an ϵ'-range query over the VA-file using Algorithm 1.
5. Refine the potential candidates returned in step 4. Compute the true DTW distances for the original-length time series and select those with DTW distance to the query object that does not exceed ϵ.

In summary, a similarity range query is performed in three steps: the VA-file filter, the VA-file refinement, and the final refinement. The VA-file filter step implies scanning only the VA-file, which is several times smaller in size than the reduced length time series database. The first refinement step implies some random disk accesses to read candidate reduced time series, while the last refinement step performs even fewer random disk accesses to read candidate original time series.

Algorithm 1 VA-file ϵ-Range Search

Input: query time series $\bar{Q}$ with envelope $Env(\bar{Q})$
Output: a list of references to potential candidates of the ϵ-range query
{Filter Step}
while scanning the VA-file **do**
  $Sum \leftarrow 0$
  skip $\leftarrow$ false
  {scan current time series $\bar{C}$}
  for $i = 1$ to N **do**
    read the current approximation $\bar{C}_i$
    compute $LVA_i(\bar{Q}_i, \bar{C}_i)$
    if $\sqrt{LVA_i(\bar{Q}_i, \bar{C}_i)} > \epsilon'$ **then**
      skip in the VA-file to the end of the Nth approximation
      skip $\leftarrow$ true
    else
      $Sum \leftarrow Sum + LVA_i(\bar{Q}_i, \bar{C}_i)$
      if $\sqrt{Sum} > \epsilon'$ **then**
        skip in the VA-file to the end of the Nth approximation
        skip $\leftarrow$ true
      end if
    end if
    if *skip* **then**
      break
    end if
  end for
  if $\neg$*skip* **then**
    mark $\bar{C}$ as a good time series
  end if
end while
{Refinement Step}
for all good time series $\bar{C}$ **do**
  read time series $\bar{C}$ from disk
  compute $LB_{\mathrm{d-dim}}(Env(\bar{Q}), \bar{C})$
  if $LB_{\mathrm{d-dim}}(Env(\bar{Q}), \bar{C}) \leq \epsilon'$ **then**
    mark $\bar{C}$ as a potential candidate
  end if
end for
return a list of references to all potential candidates

5 Experimental Results

The experiments reported upon in this section were run on a database of 12,086 music clips, each extracted from a different song. A clip is 30 seconds long. This length is relevant because the maximum legal length of a piece of music that can freely be distributed on Internet is 30 seconds. The music excerpts have been further processed by extracting AR coefficients [21], which have previously been shown to work well for genre classification. More specifically, so-called Mel Frequency Cepstral Coefficients (MFCCs) were initially extracted from short audio frames using a frame size of 30ms and a hop size of 10ms. These were then integrated over larger frames using the Autoregressive

(AR) model, which is a well-known technique for time series regression. In the end, 95-dimensional feature vectors were obtained for each second of music. A music object in our database is thus a 95-dimensional time series of length 30.

To speed up the query processing, each time series is assigned an ID and is written on disk in a separate file in binary format. The candidate time series that need to be refined (during both the VA-file and the final refinement) are represented by their ID, the IDs being stored in sorted order. The original data occupies approximatively 131MB.

In preparation for computing similarity ϵ-range queries, we compute reduced length time series for the entire database and then index them using the VA-file as described in Section 4.3. The VA-file implementation uses 8 bits per dimension. To construct the VA-file partitions in each dimension according to our data distribution, we use code from the open-source library Colt [7]. The reduced length time series approximations are written in the VA-file in the following way: the approximation for the entire reduced time series is written followed by the ID, which is a 4 byte integer.

When scanning the VA-file, we use a small 2.25MB buffer for all experiments. The buffer is used as follows: at startup, we load all the music object approximations that fit, the rest being read from disk when needed during the query processing. Computing the reduced-length time series and generating the VA-file are considered as startup costs. These costs are excluded from the query execution time measurements. DTW distances are computed with a warping range of $r = 2$, $r = 3$, and $r = 4$ that will generate a Sakoe-Chiba band of width 5, 7, and 9, respectively, which is moderately small relative to the length of the time series, thus avoiding pathological warping paths.

For all experiments, we query with 100 randomly selected objects from the database, and we report average measurements over these 100 queries. The code is implemented in Java and all tests have been run on a Pentium 4 PC with a 3.00GHz processor and 2GB main memory, under Windows XP.

We first investigate the pruning power of the VA-file filter and refinement steps as detailed in Algorithm 1 in Section 4.3, to determine whether using the VA-file refinement is effective in reducing query performance.

Figure 4 shows the number of candidates selected after VA-filtering and after the VA-refinement, for different values for ϵ and time series length reductions. Assuming that the distances in the reduced space decrease proportionally with the length reduction, ϵ' is computed as $\epsilon' = \frac{N}{n}\epsilon$, where n is the original time series length and N is the reduced length. We can see that for all cases, the VA-filtering performs very well relative to the VA-refinement. Only very few potential candidates are discarded during VA-refinement.

Figures 5 and 6 further explore VA-refinement,

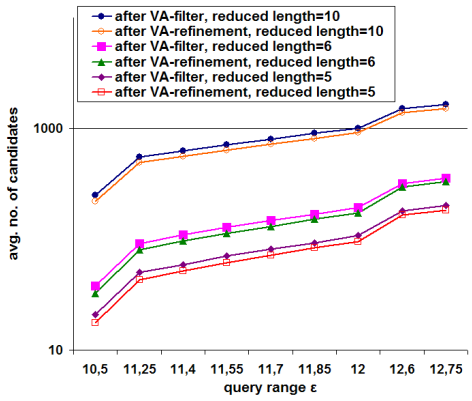


Figure 4: Average Number of Retrieved Candidates After VA-Filtering and after VA-Refinement

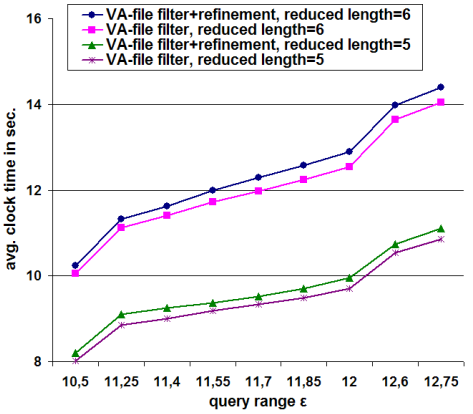


Figure 5: Average Query Retrieval Time for Reduced Length 5 and 6, Using VA-Filtering and Refinement vs. Using Only VA-Filtering

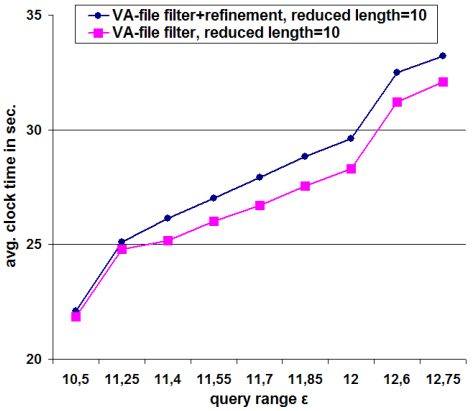
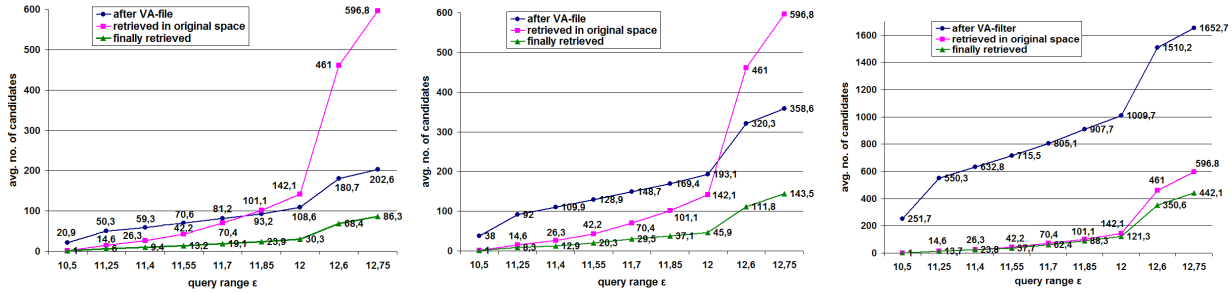


Figure 6: Average Query Retrieval Time for Reduced Length 10, Using VA-Filtering and Refinement vs. Using Only VA-Filtering

which entails storing the reduced-length time series on disk and performing random disk accesses to read all time series selected during VA-filtering. As shown in



(a) Length =5 (b) Length = 6 (c) Length = 10

Figure 7: Indexing Method Pruning Power for Time Series

the two figures, skipping the VA-file refinement yields a small performance improvement. This improvement is offset by only few extra random disk accesses to read the original time series during the final refinement.

As omitting the VA-refinement saves disk because the reduced time series do not need to be stored on disk (it is only necessary to compute them when generating the VA-file), for the rest of the experiments, each query is processed in two-steps: the VA-filtering and the final refinement where the true DTW is computed for original time series.

Our next experiments measure the impact of time series length reduction on the number of retrieved candidates. We reduce the time series length from 30 to 5, 6, and 10. For different ϵ values, we count the average number of candidates retrieved after VA-filtering and in the final result. The value of ϵ' is computed as before: $\epsilon' = \frac{N}{n}\epsilon$. The number of candidates returned after VA-filtering is closely correlated with the number of I/Os needed.

The results are reported in Figures 7(a), 7(b), and 7(c). To judge the recall of the query results, we compute the numbers of objects in the query results when the queries are computed in the original space. The figures show that, as expected, the smaller the length reduction is, the better the recall is (ϵ' gets closer to ϵ and fewer false dismissals occur).

For the same ϵ and $\epsilon' = \frac{N}{n}\epsilon$, more candidates are found when using a smaller length reduction. However, this requires much more candidates that need to be refined by the computation of true DTW distances. We can observe that the number of candidates returned from the VA-filtering grows much faster than the number of finally retrieved candidates as the length reduction becomes smaller.

Consequently, this results in worse performance. To evaluate whether it is worth using a smaller length reduction to obtain higher recall, we measure the average query retrieval time. Figure 8 shows the estimated average wall clock time for computing a query for different length reductions while varying ϵ and $\epsilon' = \frac{N}{n}\epsilon$.

As a sanity check, we also measure the performance of two naive search strategies: (1) scan all original time series and compute the true DTW distances for each;

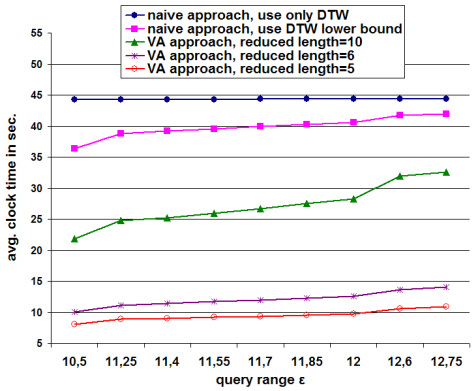


Figure 8: Average Query Retrieval Time When Varying the Time Series Length

(2) scan all original time series, compute $LB_{\mathrm{d-dim}}$ for each and compute the true DTW distances only for those that satisfy $LB_{\mathrm{d-dim}} \leq \epsilon$. When using smaller length reductions, the query retrieval time exhibits a tendency to increase quite quickly.

This leads to the following idea: to improve query performance and improve query recall, we may apply increased length reduction, but compensate by using $\epsilon' \geq \frac{N}{n}\epsilon$ in the reduced space—see Table 1.

For instance, with $\epsilon = 11.25$ and using a length of 10 and $\epsilon' = \frac{N}{n}\epsilon$, we find on average 93.83% of the candidates out of a total of 14.6 qualifying candidates in 24.8 seconds. Using a length of 5 and $\epsilon' \geq \frac{N}{n}\epsilon$ we can find on average 94.52% of the candidates in only 15.19 seconds. By lowering the query recall to approximatively 83%, we can use a length of 5 and obtain a query retrieval time of approximatively 12 seconds. We notice that increased length reduction is advantageous, even when comparing the results for reduced length 5 and reduced length 6. For instance, when $\epsilon = 11.4$, retrieving on average 90.11% of the total of 26.3 qualifying candidates requires on average 17.09 seconds for length 6, but only 15.22 seconds for length 5.

Music similarity is highly subjective, and there is no ground truth when querying a large music database. In practice, it is therefore likely to be preferable to

ϵ	ϵ'	N	retrieved (avg.)	% from total	time (sec.)
11.25	3.75	10	13.7	93.83%	24.8
11.25	2.15	5	10.3	70.54%	11.05
11.25	2.40	6	10.2	69.86%	12.48
11.25	2.25	5	12.2	83.56%	11.89
11.25	2.49	6	11.8	80.82%	13.28
11.25	2.50	5	13.8	94.52%	15.19
11.25	2.76	6	13.7	93.83%	16.73
11.4	3.8	10	23.8	90.49%	25.17
11.4	2.225	5	18.7	71.1%	11.68
11.4	2.49	6	18.8	71.48%	13.3
11.4	2.325	5	21.5	81.74%	12.8
11.4	2.58	6	21.1	80.22%	14.41
11.4	2.5	5	23.7	90.11%	15.22
11.4	2.79	6	23.7	90.11%	17.09

Table 1: Trading Query Retrieval Recall for Query Execution Time

compute an "approximate" query result very fast, over computing the "exact" result slowly.

The next experiments measure the effects of varying the warping range r, used when computing the DTW distance. Figures 9 and 10 report the average number of retrieved candidates and the average query retrieval time, respectively, when using the VA approach for time series of reduced length 5 and when increasing the value of r from 2 to 3 and 4. As expected, enlarging the warping range allows for a less strict similarity matching. Thus, more candidates are found and more time is needed for processing a query.

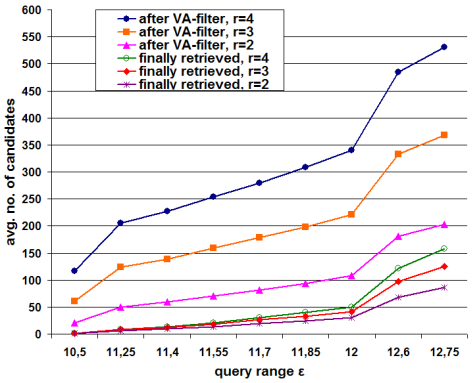


Figure 9: Average Number of Retrieved Candidates for Reduced Length 5, for Various Warping Ranges

In a last round of experiments, we evaluate the utility of using the VA-file vs. using a linear scan in the reduced length space. For different length reductions and ϵ values, we measure the estimated average clock time for solving a query (see Figure 11). Using the VA-file involves both filtering and refinement, while the linear scan is performed by reading the reduced-length time series from disk, once for every query. We see that use of the VA-file yields on average 2 times better performance than linear scan for reduced lengths

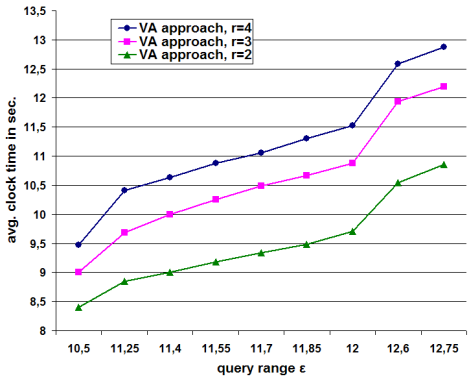


Figure 10: Average Query Retrieval Time for Reduced Length 5, for Various Warping Ranges

smaller than 10. However, above this value, the VA-file becomes slower as it needs much more computations to reconstruct the time series from the indexed multi-dimensional points.

Finally, we have tested whether the VA-file retrieval results are affected by the use of a non-L_p distance function. All experiments, run for different length reductions and ϵ values, have shown that the VA-file refinement step returns the same number of candidates and the same candidates as a linear scan in the reduced space.

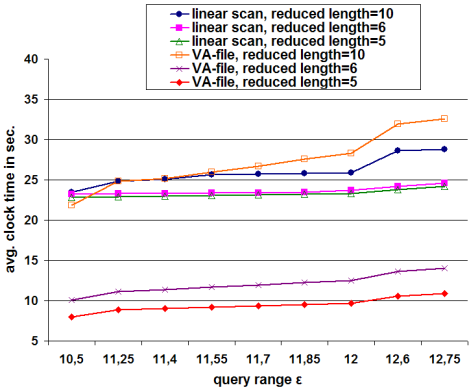


Figure 11: Average Query Retrieval Time for the VA-File vs. Linear Scan

6 Conclusion and Research Directions

We have proposed techniques for efficient similarity range query processing in music databases where each music object is represented as a high-dimensional time series and where dynamic time warping distance is use as the notion of similarity between music objects.

The proposal extends known techniques for defining a lower bounding function for dynamic time warping distance and for time series length reduction to the multi-dimensional case.

It applies the VA-file to the reduced-length time series for performing approximate similarity range

queries. This entails adaption of the VA-file to index time series as well as a new algorithm that takes into account that time series of multi-dimensional points, not simply multi-dimensional points, are being indexed. The proposal also involves the use of a non-L_p metric as a distance function, without this affecting the query retrieval precision.

Finally, the paper reports on empirical performance studies that demonstrate that it is possible to trade query recall for improved performance, by using time series length reduction and appropriate query expansion. The studies also suggest that the refinement step in the VA-file is ineffective and thus should be omitted.

An interesting aspect left for future work is to provide means of estimating an appropriate ϵ' value for a given music database, ϵ-range query, and time series length reduction. Moreover, performance studies with larger music collections and full-length songs are warranted.

Next, while quite different from the research reported on here, it is relevant to study the meaningfulness of using the DTW distance and the AR coefficients for comparing similar songs. While it seems this combination yields good comparison results, systematical studies are needed. Finally, it is relevant to attempt to identify better notions of music similarity.

While we apply the proposed techniques in the framework of music databases, the techniques are more general, and it is of interest to apply the techniques to different multi-dimensional time series databases.

References

[1] Amazon. Available via `http://www.amazon.com/`.

[2] All Music Guide. Available via `http://www.allmusic.com/`.

[3] T. Argyros and C. Ermopoulos. Efficient subsequence matching in time series databases under time and amplitude transformations. In *Proc. of IEEE International Conference on Data Mining*, pp. 481–491, 2003.

[4] Z. Bar-Joseph, G. Gerber, D. Gifford, T. Jaakkola, and I. Simon. A new approach to analyzing gene expression time series data. In *Proc. of International Conference on Research in Computational Molecular Biology*, pp. 39–48, 2002.

[5] N. Beckmann, H. P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and rectangles. In *Proc. of ACM SIGMOD*, pp. 322–331, 1990.

[6] D. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *AAAI-94 Workshop on Knowledge Discovery in Databases*, pp. 229–248, 1994.

[7] Colt. A set of open source libraries for high performance scientific and technical computing in java. Available via `http://dsd.lbl.gov/~hoschek/colt/index.html`.

[8] C. Faloutsos and K.-I. Lin. FastMap: A fast algorithm for indexing, data-minining and visualization of traditional and multimedia datasets. In *Proc. of ACM SIGMOD*, pp. 163–174, 1995.

[9] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proc. of ACM SIGMOD*, pp. 419–429, 1994.

[10] D. M. Gavrila and L. S. Davis. Towards 3-d model-based tracking and recognition of human movement: a multi-view approach. In *International Workshop on Automatic Face- and Gesture-Recognition*, pp. 272–277, 1995.

[11] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proc. of ACM SIGMOD*, pp. 47–57, 1984.

[12] F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions On Acoustics, Speech and Signal Processing*, ASSP-23(1):67–72, 1975.

[13] T. Kahveci, A. Singh, and A. Gurel. Similarity searching for multi-attribute sequences. In *Proc. of SSDBM*, pp. 175–184, 2002.

[14] E. Keogh. Exact indexing of dynamic time warping. In *Proc. of VLDB*, pp. 406–417, 2002.

[15] E. Keogh, K. Chakrabarti, S. Mehrotra, and M. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In *Proc. of ACM SIGMOD*, pp. 151–162, 2001.

[16] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems*, 3(3):263–286, 2001.

[17] S. Kim, S. Park, and W. Chu. An index-based approach for similarity search supporting time warping in large sequence databases. In *Proc. of ICDE*, pp. 607–614, 2001.

[18] Z. M. Kovacs-Vajna. A fingerprint verification system based on triangular matching and dynamic time warping. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 22(11):1266–1276, 2000.

[19] S.-L. Lee, S.-J. Chun, D.-H. Kim, J.-H. Lee, and C.-W. Chung. Similarity search for multidimensional data sequences. In *Proc. of ICDE*, pp. 599–608, 2000.

[20] S.-H. Lim, H.-J. Park, and S.-W. Kim. Using multiple indexes for efficient subsequence matching in time-series databases. In *Proc. of DASFAA*, pp. 65–79, 2006.

[21] A. Meng, P. Ahrendt, and J. Larsen. Improving music genre classification by short-time feature integration. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. V, pp. 497–500, 2005.

[22] Pandora. Music Genome Project. Available via `http://www.pandora.com/`.

[23] C. Ratanamahatana and E. Keogh. Making time-series classification more accurate using learned constraints. In *Proc. of SDM*, pp. 11–22, 2004.

[24] T. Rath and R. Manmatha. Word image matching using dynamic time warping. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 521–527, 2003.

[25] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions On Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

[26] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. Keogh. Indexing multi-dimensional time-series with support for multiple distance measures. In *Proc. of ACM KDD*, pp. 216–225, 2003.

[27] R. Weber and S. Blott. An approximation-based data structure for similarity search. *Technical Report 24, ESPRIT project HERMES (no. 9141)*, 1997.

[28] R. Weber and K. Bohm. Trading quality for time with nearest-neighbor search. In *Proc. of EDBT*, pp. 21–35, 2000.

[29] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proc. of VLDB*, pp. 194–205, 1998.

[30] B.-K. Yi, H. V. Jagadish, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *Proc. of ICDE*, pp. 201–208, 1998.

[31] Y. Zhu and D. Shasha. Warping indexes with envelope transforms for query by humming. In *Proc. of ACM SIGMOD*, pp. 181–192, 2003.

Appendix

Lemma 2 - Proof: We have to prove that

$$\sqrt{\sum_{i=1}^{n} d(Env(Q)_i, C_i)} \leq \sqrt{\sum_{k=1}^{K} w_k}$$

The proof is similar to that of Lemma 1 [14]. We assume the opposite and show that this leads to a contradiction:

$$\sqrt{\sum_{i=1}^{n} d(Env(Q)_i, C_i)} > \sqrt{\sum_{k=1}^{K} w_k}$$

The terms under the radicals are positive, so we can square both sides:

$$\sum_{i=1}^{n} d(Env(Q)_i, C_i) > \sum_{k=1}^{K} w_k$$

We know that $n \leq K$, so we can match every term on the left with a unique term on the right, which leaves $K - n$ terms unmatched.

$$\sum_{i=1}^{n} d(Env(Q)_i, C_i) > \sum_{k \in matched} w_k + \sum_{k \in unmatched} w_k$$

We map the i^{th} term on the left side with one of the i'^{th} terms on the right side. Having several values of i' for a single i, we can choose for the mapping the i' with the smallest value. All the other w_k are placed in the unmatched summation.

In the following, we try to determine what relational operator is to be applied between one term on the left side and its matched term on the right side.

$$d(Env(Q)_i, C_i) <?> w_k$$

$$\sum_{j=1}^{d} \begin{cases} (c_{ij} - u_{ij})^2 & \text{if } c_{ij} > u_{ij} \\ (c_{ij} - l_{ij})^2 & \text{if } c_{ij} < l_{ij} \\ 0 & \text{otherwise} \end{cases} <?> \sum_{j=1}^{d} (c_{ij} - q_{i'j})^2$$

Next, we consider the relation between each j terms in the summations and analyze the three cases that are possible. We are in the 1-dimensional case, so we can apply the same logics as in the proof of Lemma 1. First, we consider the case when $c_{ij} > u_{ij}$.

$$(c_{ij} - u_{ij})^2 <?> (c_{ij} - q_{i'j})^2$$

$$(c_{ij} - u_{ij}) <?> (c_{ij} - q_{i'j})$$

$$-u_{ij} <?> -q_{i'j}$$

$$q_{i'j} <?> u_{ij}$$

$$q_{i'j} <?> max(q_{(i-r)j} : q_{(i+r)j})$$

Because time series Q and C have the same length n, $i' - r \leq i \leq i' + r$ implies $i - r \leq i' \leq i + r$. So,

$$q_{i'j} \leq max(q_{(i-r)j} : q_{(i+r)j})$$

The case where $c_{ij} < l_{ij}$ is handled similarly. The third case is trivial ($0 \leq (c_{ij} - q_{i'j})^2$).

We can conclude that $d(Env(Q)_i, C_i) \leq w_k$. But if all matched terms in $\sum_{k \in matched} w_k$ are larger than their counterparts on the left side then $\sum_{k \in unmatched} w_k$ has to be negative, which is wrong as the square root of a sum of squared terms cannot be negative. Thus, we have reached a contradiction.