Chapter 2

A Survey of Recent Methods for Efficient Retrieval of Similar Time Sequences

Magnus Lie Hetland
Norwegian University of Science and Technology
Sem Sælands vei 7-9
Trondheim, NO-7491, Norway
magnus@hetland.org

Time sequences occur in many applications, ranging from science and technology to business and entertainment. In many of these applications, searching through large, unstructured databases based on sample sequences is often desirable. Such similarity-based retrieval has attracted a great deal of attention in recent years. Although several different approaches have appeared, most are based on the common premise of dimensionality reduction and spatial access methods. This paper gives an overview of recent research and shows how the methods fit into a general context of signature extraction.

Keywords: Information retrieval, sequence databases, similarity search, spatial indexing, time sequences.

1 Introduction

Time sequences arise in many applications—any applications that involve storing sensor inputs, or sampling a value that changes over time. A problem which has received an increasing amount of attention lately is the problem of *similarity retrieval* in databases of time sequences, so-called "query by example." Some uses of this are [Agrawal *et al.* (1993)]:

- Identifying companies with similar patterns of growth.
- Determining products with similar selling patterns.
- Discovering stocks with similar movement in stock prices.
- Finding out whether a musical score is similar to one of a set of copyrighted scores.

- Finding portions of seismic waves that are not similar to spot geological irregularities.

Applications range from medicine, through economy, to scientific disciplines such as meteorology and astrophysics [Faloutsos *et al.* (1994), Yi and Faloutsos (2000)].

The running times of simple algorithms for comparing time sequences are generally polynomial in the length of both sequences, typically linear or quadratic. To find the correct offset of a query in a large database, a naive *sequential scan* will require a number of such comparisons that is linear in the length of the database. This means that, given a query of length m and a database of length n, the search will have a time complexity of $O(nm)$, or even $O(nm^2)$. For large databases this is clearly unacceptable.

Many methods are known for performing this sort of query in the domain of strings over finite alphabets, but with time sequences there are a few extra issues to deal with:

- The range of values is not generally finite, or even discrete.
- The sampling rate may not be constant.
- The presence of noise in various forms makes it necessary to support very flexible similarity measures.

This chapter describes some of the recent advances that have been made in this field; methods that allow for indexing of time sequences using flexible similarity measures that are invariant under a wide range of transformations and error sources.

The chapter is structured as follows: Section 1.2 gives a more formal presentation of the problem of similarity based retrieval and the so-called *dimensionality curse*; Section 1.3 describes the general approach of signature based retrieval, or *shrink and search*, as well as three specific methods using this approach; Section 1.4 shows some other approaches, while Section 1.5 concludes the chapter. Finally, Appendix A gives an overview of some basic distance measures.[1]

[1] The term “distance” is used loosely in this paper. A distance measure is simply the inverse of a similarity measure and is not required to obey the metric axioms.

1.1 Terminology and Notation

A time sequence $\vec{x} = \langle x_1 = (v_1, t_1), \cdots, x_n = (v_n, t_n) \rangle$ is an ordered collection of elements x_i, each consisting of a value v_i and a timestamp t_i. Abusing the notation slightly, the value of x_i may be referred to as x_i. For some retrieval methods, the values may be taken from a finite class of values [Mannila and Ronkainen (1997)], or may have more than one dimension [Lee *et al.* (2000)], but it is generally assumed that the values are real numbers. This assumption is a requirement for most of the methods described in this chapter.

The only requirement of the timestamps is that they be nondecreasing (or, in some applications, strictly increasing) with respect to the sequence indices:

$$t_i \le t_j \Leftrightarrow i \le j \tag{1}$$

In some methods, an additional assumption is that the elements are *equispaced*: For every two consecutive elements x_i and x_{i+1} we have

$$t_{i+1} - t_i = \Delta \tag{2}$$

where Δ (the *sampling rate* of $\vec{x}$) is a (positive) constant. If the actual sampling rate is not important, Δ may be normalised to 1, and t_1 to 0. The *length* of a time sequence $\vec{x}$ is its cardinality, written as $|\vec{x}|$. The contiguous subsequence of $\vec{x}$ containing elements x_i to x_j (inclusive) is written $x_{i:j}$. A *signature* of a sequence $\vec{x}$ is some structure that somehow represents $\vec{x}$, yet is simpler than $\vec{x}$. In the context of this chapter, such a signature will always be a vector of fixed size *k*. (For a more thorough discussion of signatures, see Section 1.3.) Such a signature is written $\tilde{x}$. For a summary of the notation, see Table 1.

Table 1 Notation

$\vec{x}$	A sequence
$\tilde{x}$	A signature of $\vec{x}$
x_i	Element number i of $\vec{x}$
$x_{i:j}$	Elements i to j (inclusive) of $\vec{x}$
$\lvert\vec{x}\rvert$	The length of $\vec{x}$

2 The Problem

The problem of retrieving similar time sequences may be stated as follows: Given a sequence $\vec{q}$, a set of time sequences *X*, a (non-negative) distance measure *d*, and a *tolerance threshold* ε, find the set *R* of sequences closer to $\vec{q}$ than ε, or, more precisely:

$$R = \{\vec{x} \in X \mid d(\vec{q},\vec{x}) \leq \varepsilon\} \quad (3)$$

Alternatively, one might wish to find the *k* nearest neighbours of $\vec{q}$, which amounts to setting ε so that $|R| = k$. The parameter ε is typically supplied by the user, while the distance function *d* is domain-dependent. Several distance measures will be described rather informally in this chapter. For more formal definitions, see Appendix A.

Figure 1 illustrates the problem for Euclidean distance in two dimensions. In this example, the vector $\vec{x}$ will be included in the result set *R*, while $\vec{y}$ will not.

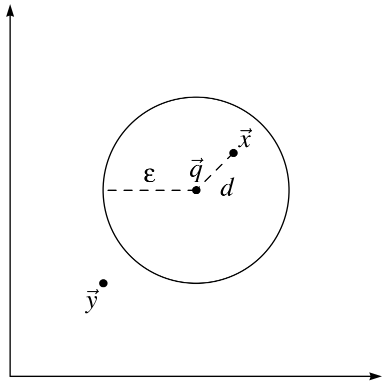

Figure 1 Similarity Retrieval.

A useful variation of the problem is to find a set of *subsequences* of the sequences in *X*. This, in the basic case, requires comparing $\vec{q}$ not only to all elements of *X*, but to all possible subsequences.[2]

If a method retrieves a subset *S* of *R*, the wrongly dismissed sequences in $R - S$ are called *false dismissals*. Conversely, if *S* is a superset of *R*, the sequences in $S - R$ are called *false alarms*.

2.1 Robust Distance Measures

The choice of distance measure is higly domain dependent, and in some cases a simple L_p norm such as Euclidean distance may be sufficient. However, in many cases, this may be too brittle [Keogh and Pazzani (1999b)] since it does not tolerate such transformations as scaling, warping, or translation along either axis. Many of the newer methods focus on using more robust distance measures, which are invariant under such transformations as *time warping* [Sankoff and Kruskal (1999)] without loss of performance.

[2] Except in the description of LCS in Appendix A, *subsequence* means *contiguous subsequence*, or *segment*.

2.2 Good Indexing Methods

Faloutsos *et al.* [Faloutsos *et al.* (1994)] list the following desirable properties for an indexing method:

1. It should be faster than a sequential scan.
2. It should incur little space overhead.
3. It should allow queries of various length.
4. It should allow insertions and deletions without rebuilding the index.
5. It should be correct: No false dismissals must occur.

To achieve high performance, the number of false alarms should also be low. Keogh *et al.* [Keogh *et al.* (2001b)] add the following criteria to the list above:

6. It should be possible to build the index in reasonable time.
7. The index should preferably be able to handle more than one distance measure.

2.3 Spatial Indices and the Dimensionality Curse

The general problem of similarity based retrieval is well known in the field of information retrieval, and many indexing methods exist to process queries efficiently [Baeza-Yates and Ribeiro-Neto (1999)]. However, certain properties of time sequences make the standard methods unsuitable. The fact that the value ranges of the sequences usually are continuous, and that the elements may not be equi-spaced, makes it difficult to use standard text-indexing techniques such as suffix-trees. One of the most promising techniques is multidimensional indexing (*R*-trees [Guttman (1984)], for instance), in which the objects in question are multidimensional vectors, and similar objects can be retrieved in sublinear time. One requirement of such spatial access methods is that the distance measure used obeys the triangle inequality ($d(\vec{x},\vec{z}) \leq d(\vec{x},\vec{y}) + d(\vec{y},\vec{z})$).

One important problem that occurs when trying to index sequences with spatial acces methods is the so-called *dimensionality curse*: Spatial indices typically work only when the number of dimensions is low [Chakrabarti and Mehrotra (1999)]. This makes it unfeasible to code the entire sequence directly as a vector in an indexed space.

The general solution to this problem is *dimensionality reduction*: to condense the original sequences into *signatures* in a *signature space* of low dimensionality, in a manner which, to some extent, preserves the distances between them. One can then index the signature space.

3 Signature Based Similarity Search

A time sequence $\vec{x}$ of length n can be considered a vector or point in an n-dimensional space. Techniques exist (spatial access methods, such as the *R*-tree and variants [Chakrabarti and Mehrotra (1999), Wang and Perng (2001), Sellis *et al.* (1987)]) for indexing such data. The problem is that the performance of such methods degrades considerably even for relatively low dimensionalities [Chakrabarti and Mehrotra (1999)]; the number of dimensions that can be handled is usually several orders of magnitude lower than the number of data points in a typical time sequence.

A general solution described by Faloutsos *et al.* [Faloutsos *et al.* (1994), Faloutsos *et al.* (1997)] is to extract a low-dimensional *signature* from each sequence, and to index the signature space. This *shrink and search* approach is illustrated in Figure 2.

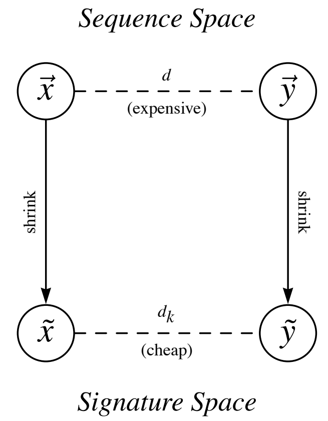

Figure 2 The Signature Based Approach

An important result given by Faloutsos *et al.* [Faloutsos *et al.* (1994)] is the proof that in order to guarantee completeness (no false dismissals), the distance function used in the signature space must underestimate the true distance measure, or:

$$d_k(\tilde{x}, \tilde{y}) \leq d(\vec{x}, \vec{y}) \tag{4}$$

This requirement is called the *bounding lemma*. Assuming that (1.4) holds, an intuitive way of stating the resulting situation is: "if two signatures are far apart, we know the corresponding [sequences] must also be far apart" [Faloutsos et al. (1997)]. This, of course, means that there will be no false dismissals. To minimise the number of false alarms, we want d_k to approximate d as closely as possible. The bounding lemma is illustrated in Figure 3.

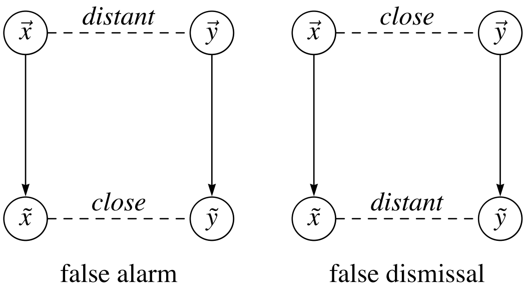

Figure 3 An Intuitive View of the Bounding Lemma

This general method of dimensionality reducion may be summed up as follows [Keogh *et al.* (2001b)]:

1. Establish a distance measure d from a domain expert.
2. Design a dimensionality reduction technique to produce signatures of length k, where k can be efficiently handled by a standard spatial access method.
3. Produce a distance measure d_k over the k-dimensional signature space, and prove that it obeys the bounding condition (4).

In some applications, the requirement in (4) is relaxed, allowing for a small number of false dismissals in exchange for increased performance. Such methods are called *approximate*.

The dimensionality reduction may in itself be used to speed up the sequential scan, and some methods (such as the piecewise linear approximation of Keogh *et al.*, which is described in Section 1.4.2) rely only on this, without using any index structure.

3.1 A Simple Example

As an example of the signature based scheme, consider the two sequences shown in Figure 4.

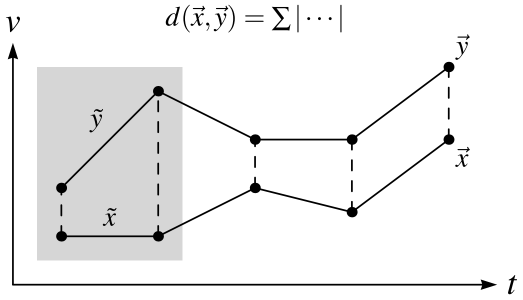

Figure 4 Comparing Two Sequences

The sequences, $\vec{x}$ and $\vec{y}$, are compared using the L_1 measure (Manhattan distance), which is simply the sum of the absolute distances between each aligning pair of values. A simple signature in this scheme is the prefix of length 2, as indicated by the shaded area in the figure. As shown in Figure 1.5, these signatures may be interpreted as points in a two-dimensional plane, which can be indexed with some standard spatial indexing method. It is also clear that the signature distance will underestimate the real distance between the sequences, since the remaining summands of the real distance must all be positive.

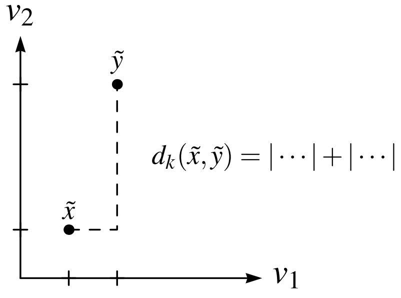

Figure 5 A Simple Signature Distance

Although correct, this simple signature extraction technique is not particularly precise. The signature extraction methods introduced in the following sections take into account more information about the full sequence shape, and therefore lead to fewer false alarms.

Figure 6 shows a time series containing measurements of atmospheric pressure. In the following three sections, the methods described will be applied to this sequence, and the resulting simplified sequence (reconstructed from the extracted signature) will be shown superimposed on the original.

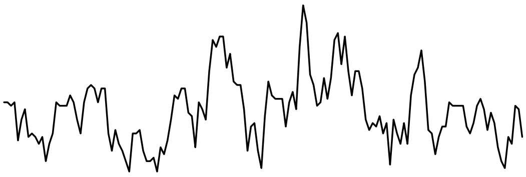

Figure 6 An Example Time Sequence

3.2 Spectral Signatures

Some of the methods presented in this section are not very recent, but introduce some of the main concepts used by newer approaces.
Agrawal *et al.* [Agrawal *et al.* (1993)] introduce a method called the *F*-index in which a signature is extracted from the frequency domain of a sequence. Underlying their approach are two key observations:

- Most real-world time sequences can be faithfully represented by their strongest Fourier coefficients.
- Euclidean distance is preserved in the frequency domain (Parseval's Theorem [Shatkay (1995)]).

Based on this, they suggest performing the Discrete Fourier Transform on each sequence, and using a vector consisting of the sequence's *k* first amplitude coefficients as its signature. Euclidean distance in the signature space will then underestimate the real Euclidean distance between the sequences, as required.

Figure 7 shows an approximated time sequence, reconstructed from a signature consisting of the original sequence's ten first Fourier components.

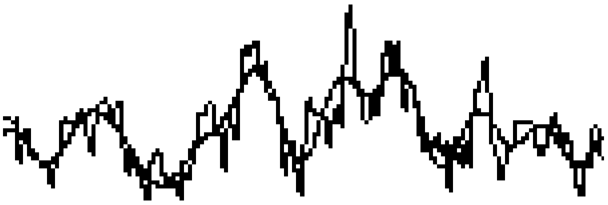

Figure 7 A Sequence Reconstructed from a Spectral Signature

This basic method allows only for whole-sequence matching. In [Faloutsos *et al.* (1994)], Faloutsos *et al.* introduce the *ST*-index, an improvement on the *F*-index that makes subsequence matching possible. The main steps of the approach are as follows:

1. For each position in the database, extract a window of length *w*, and create a spectral signature (a *point*) for it.

Each point will be close to the previous, because the contents of the sliding window change slowly. The points for one sequence will therefore constitute a *trail* in signature space.

2. Partition the trails into suitable (multidimensional) Minimal Bounding Rectangles (MBRs), according to some heuristic.
3. Store the MBRs in a spatial index structure.

To search for subsequences similar to a query $\vec{q}$ of length *w*, simply look up all MBRs that intersect a hypersphere with radius ε around the signature point $\tilde{q}$. This is guaranteed not to produce any false dismissals, because if a point is within a radius of ε of $\tilde{q}$, it cannot possibly be contained in an MBR that does not intersect the hypersphere.

To search for sequences longer than *w*, split the query into *w*-length segments, search for each of them, and intersect the result sets. Because a resulting sequence cannot be closer to the full query sequence than it is to any one of the window signatures, it has to be close to all of them, that is, contained in all the result sets.

These two papers ([Agrawal *et al.* (1993)] and [Faloutsos et al. (1994)]) are seminal; several newer approaches are based on them. For instance, Rafiei and Mendelzon [Rafiei and Mendelzon (1997)] show how the method can be made more robust by allowing various transformations in the comparison, and Chan and Fu [Chan and Fu (1999)] show how the Discrete Wavelet Transform can be used instead of the Discrete Fourier Transform, and that the DWT method is empirically superior. See [Wu *et al.* (2000)] for a comparison between DFT and DWT based similarity search.

3.3 Piecewise Constant Approximation

An approach independently introduced by Yi and Faloutsos [Yi and Faloutsos (2000)] and Keogh *et al.* [Keogh *et al.* (2001b), Keogh and Pazzani (2000)] is to divide each sequence into *k* segments of equal length, and to use the average value of each segment as a coordinate of a *k*-dimensional signature vector. Keogh et al. call the method *Piecewise Constant Approximation*, or PCA. This deceptively simple dimensionality reduction technique has several advantages [Keogh *et al.* (2001b)]: The transform itself is faster than most other transforms, it is easy to understand and implement, it supports more flexible distance measures than Euclidean distance, and the index can be built in linear time.

Figure 8 shows an approximated time sequence, reconstructed from a ten-dimensional PCA signature.

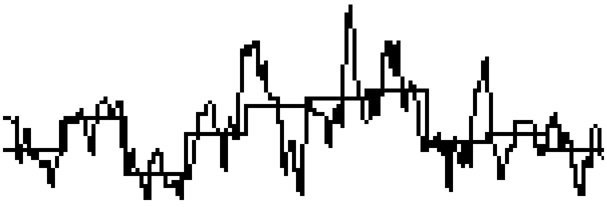

Figure 8 A Sequence Reconstructed from a PCA Signature

Yi and Faloutsos [Yi and Faloutsos (2000)] also show that this signature can be used with arbitrary L_p norms without changing the index structure, which is something no previous method (such as [Agrawal *et al.* (1993), Agrawal *et al.* (1995), Faloutsos *et al.* (1997), Faloutsos *et al.* (1994), Rafiei and Mendelzon (1997), Yi *et al.* (1998)]) could accomplish. This means that the distance measure may be specified by the user. Preprocessing to make the index more robust in the face of such transformations as *offset translation*, *amplitude scaling*, and *time scaling* can also be performed.

Keogh et al. demonstrate that the representation can also be used with the so-called *weighted Euclidean distance*, where each part of the sequence has a different weight.

Empirically, the PCA methods seem promising: Yi and Faloutsos demonstrate up to a ten times speedup over methods based on the discrete wavelet transform. Keogh et al. do not achieve similar speedups, but point to the fact that the structure allows for more flexible distance measures than many of the competing methods.

In [Keogh et al. (2001a)] Keogh *et al.* propose an improved version of the PCA, the so-called *Adaptive Piecewise Constant Approximation*, or APCA. This is similar to the PCA, except that the segments need not be of equal length. Thus regions with great fluctuations may be represented with several short seqments, while reasonably featureless regions may be represented with fewer, long segments. The main contribution of this representation is that it is a more effective compression than the PCA, while still representing the original faithfully.

Two distance measures are developed for the APCA, one which is guaranteed to underestimate Euclidean distance, and one which can be calculated more efficiently, but which may generate some false dismissals. It is also shown that this technique, like the PCA, can handle arbitrary L_p norms. The empirical data suggest that the APCA outperforms both methods based on the discrete Fourier transform, and methods based on the discrete wavelet transform with a speedup of one to two orders of magnitude.

3.4 Landmark Methods

In [Keogh and Smyth (1997)] Keogh and Smyth introduce a probabilistic method for sequence retrieval, where the features extracted are characteristic parts of the sequence, so-called *feature shapes*. In [Keogh (1997)] Keogh uses a similar *landmark based* technique. Both these methods also use the dimensionality reduction technique of piecewise linear approximation (see Section 1.4.2) as a preprocessing step. The methods are based on finding similar landmark features (or shapes) in the target sequences, ignoring shifting and scaling within given limits. The techique is shown to be significantly faster than sequential scanning (about an order of magnitude), which may be accounted for by the compression of the piecewise linear approximation. One of the contributions of the method is that it is one of the first that allows some longitudinal scaling.

A more recent paper by Perng *et al.* [Perng *et al.* (2000)] introduces a more general landmark model. In its most general form, the model allows any point of great importance to be identified as a landmark. The specific form used in the paper defines an n-th order landmark of a one-dimentional function to be a point where the function's n-th derivative is zero. Thus, first-order landmarks are extrema, second-order landmarks are inflection points, and so forth. A smothing technique is also introduced, which lets certain landmarks be overshadowed by others. For instance, local extrema representing small fluctuations may not be as important as a global maximum or minimum.

Figure 8 shows an approximated time sequence, reconstructed from a twelve-dimensional landmark signature.

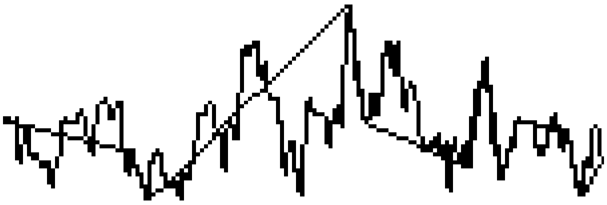

Figure 9 A Landmark Approximation

One of the main contributions of [Perng *et al.* (2000)] is to show that for suitable selections of landmark features, the model is invariant with respect to the following transformations:

- Shifting
- Uniform amplitude scaling
- Uniform time scaling
- Non-uniform time scaling (time warping)
- Non-uniform amplitude scaling

It is also possible to allow for several of these transformations at once, by using the intersection of the features allowed for each of them. This makes the method quite flexible and robust, although as the number of transformations allowed increases, the number of features will decrease; consequently, the index will be less precise.

A particularly simple landmark based method (which can be seen as a special case of the general landmark method) is introduced by Kim *et al.* in [Kim *et al.* (2001)]. They show that by extracting the minimum, maximum, and the first and last elements of a sequence, one gets a (rather crude) signature that is invariant to time warping. However, since time warping distance does not obey the triangle inequality [Yi *et al.* (1998)], it cannot be used directly. This problem is solved by developing a new distance measure that underestimates the time warping distance while simultaneously satisfying the triangle inequality.

4 Other Approaches

Not all recent methods rely on spatial access methods. This section contains a sampling of other approaches.

4.1 Using Suffix Trees to Avoid Redundant Computation

Baeza-Yates and Gonnet [Baeza-Yates and Gonnet (1999)] and Park *et al.* [Park *et al.* (2000)] independently introduce the idea of using suffix trees [Gusfield (1997)] to avoid duplicate calculations when using dynamic programming to compare a sequence with a database. In [Baeza-Yates and Gonnet (1999)] *edit distance* is used, while in [Park *et al.* (2000)] *time warping* is used (see Appendix A for a details).

The basic idea of the approach is as follows: When comparing two sequences $\vec{x}$ and $\vec{y}$ with dynamic programming, a subtask will be to compare their prefixes $x_{1:i}$ and $y_{1:j}$. If two other sequences are compared that have identical prefixes to these (for instance, the query and another sequence from the database), the same calculations will have to be performed again. If a sequential search for subsequence matches is performed, the cost may easily become prohibitive.

To avoid this, all the sequences in the database are indexed with a suffix tree. A suffix tree stores all the suffixes of a sequence, with identical prefixes stored only once. By performing a depth-first traversal of the suffix tree one can access every suffix (which is equivalent to each possible subsequence position) and backtrack to reuse the calculations that have already been performed for the prefix that the current and the next candidate subsequence share.

In [Baeza-Yates and Gonnet (1999)] it is assumed that the sequences are strings over a finite alphabet; Park et al. avoid this assumption by classifying each sequence element into one of a finite set of categories. Both methods achieve subquadratic running times.

4.2 Data Reduction through Piecewise Linear Approximation

Keogh *et al*. have introduced a dimensionality reduction technique using piecewise linear approximation of the original sequence data [Keogh (1997), Keogh and Pazzani (1998), Keogh and Pazzani (1999a), Keogh and Pazzani (1999b), Keogh and Smyth (1997)]. This reduces the number of data points by a compression factor typically in the range from 10 to 600 for real data [Keogh (1997)], outperforming methods based on the Discrete Fourier Transform by one to three orders of magnitude [Keogh and Pazzani (1999b)]. This approximation is shown to be valid under several distance measures, including dynamic time warping distance [Keogh and Pazzani (1999b)]. An enhanced representation is introduced in [Keogh and Pazzani (1998)], where every line segment in the approximation is augmented with a weight representing its relative importance; for instance, a combined sequence may be constructed representing a class of sequences, and some line segments may be more representative of the class than others.

4.3 Search Space Pruning through Subsequence Hashing

In [Keogh and Pazzani (1999a)] Keogh and Pazzani introduce an indexing method based on hashing, in addition to the piecewise linear approximation. An equi-spaced template grid window is moved across the sequence, and for each position a hash key is generated to decide into which *bin* the corresponding subsequence is put. The hash key is simply a binary string, where 1 means that the sequence is predominantly increasing in the corresponding part of the template grid, while 0 means that it is decreasing. These bin keys may then be used during a search, to prune away entire bins without examining their contents. To get more benefit from the bin pruning, the bins are arranged in a *best-first* order.

5 Conclusion

This chapter has sought to give an overview of recent advances in the field of similarity based retrieval in time sequence databases. First, the problem of similarity search and the desired properties of robust distance measures and good indexing methods were outlined. Then, the general approach of signature based similarity search was described. Following the general description, three specific signature extraction approaches were discussed: Spectral signatures, based on Fourier components (or wavelet components); piecewise constant approximation, and the related method adaptive piecewise constant approximation; and landmark methods, based on the extraction of significant points in a sequence. Finally, some methods that are not based on signature extraction were mentioned.

Although the field of time sequence indexing has received much attention and is now a relatively mature field [Keogh *et al.* (2002)] there are still areas where further research might be warranted. Two such areas are (1) thorough empirical comparisons and (2) applications in data mining.

The published methods have undergone thorough empirical tests that evaluate their performance (usually by comparing them to sequential scan, and, in some cases, to the basic spectral signature methods), but comparing the results is not a trivial task—in most cases it might not even be very meaningful, since variations in performance may be due to implementation details, available hardware, and several other factors that may not be inherent in the indexing methods themselves. Implementing several of the most promising methods and testing them on real world problems (under similar conditions) might lead to new insights, not only about their relative performances in general, but also about which methods are best suited for which problems. Although some comparisons have been made (such as in [Wu *et al.* (2000)] and, in the more general context of spatial similarity search, in [Weber *et al.* (1998)]), little research seems to have been published on this topic.

Data mining in time series databases is a relatively new field [Keogh *et al.* (2002)]. Most current mining methods are based on a full, linear scan of the sequence data. While this may seem unavoidable, constructing an index of the data could make it possible to perform this full data traversal only once, and later perform several data mining passes that only use the index to perform their work. It has been argued that data mining should be interactive [Das *et al.* (1998)], in which case such techniques could prove useful. Some publications can be found about using time sequence indexing for data mining purposes (such as [Keogh *et al.* (2002)], where a method is presented for mining patterns using a suffix tree index) but there is still a potential for combining existing sequence mining techniques with existing methods for similarity-based retrieval.

Appendix A Distance Measures

Faloutsos *et al.* [Faloutsos *et al.* (1997)] describe a general framework for sequence distance measures (a similar framework can be found in

[Jagadish *et al.* (1995)]). They show that many common distance measures can be expressed in the following form:

$$d(\vec{x},\vec{y}) = \min\begin{cases}\min_{T_1,T_2\in T}\{c(T_1)+c(T_2)+d(T_1(\vec{x}),T_2(\vec{y}))\}\\ d_0(\vec{x},\vec{y})\end{cases} \tag{5}$$

T is a set of allowable transformations, $c(T_i)$ is the *cost* of performing the transformation T_i, $T_i(\vec{x})$ is the sequence resulting from performing the transformation T_i on $\vec{x}$, and d_0 is a so-called *base distance*, typically calculated in linear time. For instance, L_p norms (such as Manhattan distance and Euclidean distance) results when $T=\varnothing$ and

$$d_0(\vec{x},\vec{y}) = L_p = \sqrt[p]{\sum_{i=1}^{l}\left|x_i-y_i\right|^p} \tag{6}$$

where $|\vec{x}|=|\vec{y}|=l$.

Editing distance (or Levenshtein distance) is the weight of the minimum sequence of editing operations needed to transform one sequence into another [Sankoff and Kruskal (1999)]. It is usually defined on strings (or equi-spaced time sequences), but in [Mannila and Ronkainen (1997)] Mannila and Ronkainen show how to generalise this measure to general (non equi-spaced) time sequences. In the framework given above, editing distance may be defined as:

$$d_{ed}(\vec{x},\vec{y}) = \min\begin{cases}c(del(x_1))+d_{ed}(x_{2:m},\vec{y})\\ c(del(y_1))+d_{ed}(\vec{x},y_{2:n})\\ c(sub(x_1,y_1))+d_{ed}(x_{2:m},y_{2:n})\end{cases} \tag{7}$$

where $m=|\vec{x}|$, $n=|\vec{y}|$, $del(x_1)$ and $del(y_1)$ stand for deleting the first elements of $\vec{x}$ and $\vec{y}$, respectively, and $sub(x_1,y_1)$ stands for substituting the first element of $\vec{x}$ with the first element of $\vec{y}$.

A distance function with time warping allows non-uniform scaling along the time axis, or, in sequence terms, *stuttering*. Stuttering occurs when an element from one of the sequences is repeated several times. A typical distance measure is:

$$d_{tw}(\vec{x},\vec{y}) = d_0(x_1,y_1) + \min\begin{cases} d_{tw}(\vec{x},y_{2:n}) & (\vec{x}\text{ - stutter}) \\ d_{tw}(x_{2:m},\vec{y}) & (\vec{y}\text{ - stutter}) \\ d_{tw}(x_{2:m},y_{2:n}) & (\text{no stutter}) \end{cases} \tag{8}$$

Both d_{ed} and d_{tw} can be computed in quadratic time ($O(mn)$) using dynamic programming [Cormen et al. (1993), Sankoff and Kruskal (1999)]: An $m \times n$ table D is filled iteratively so that $D[i,j] = d(x_{1:i}, y_{1:j})$. The final distance $d(\vec{x},\vec{y})$ is found in $D[m,n]$.

The *Longest Common Subsequence* (LCS) measure [Cormen et al. (1993)], $d_{lcs}(\vec{x},\vec{y})$, is the length of the longest sequence $\vec{s}$ which is a (possibly non-contiguous) subsequence of both $\vec{x}$ and $\vec{y}$, in other words:

$$d_{lcs}(\vec{x},\vec{y}) = \max\{|\vec{s}| \mid \vec{s} \subseteq \vec{x}, \vec{s} \subseteq \vec{y}\} \tag{9}$$

In some applications the measure is normalised by dividing by $\max(|\vec{x}|,|\vec{y}|)$, giving a distance in the range $[0,1]$. $d_{lcs}(\vec{x},\vec{y})$ may be calculated using dynamic programming, in a manner quite similar to d_{ed} .

Bibliography

Agrawal, R., Faloutsos, C., and Swami, A. N. (1993). Efficient Similarity Search in Sequence Databases, *Proc. 4th Int. Conf. on Foundations of Data Organization and Algorithms*, FODO, pp. 69–84.

Agrawal, R., Lin, K., Sawhney, H. S., and Shim, K. (1995). Fast similarity search in the presence of noise, scaling, and translation in time-series database, *Proc. 21st Int. Conf. on Very Large Databases*, VLDB, pp. 490–501.

Baeza-Yates, R. and Gonnet, G. H. (1999). A fast algorithm on average for all-against-all sequence matching, *Proc. 6th String Processing and Information Retrieval Symposium*, SPIRE, pp. 16–23.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval.* ACM Press/Addison–Wesley Longman Limited.

Chakrabarti, K. and Mehrotra, S. (1999). The hybrid tree: An index structure for high dimensional feature spaces, *Proc. 15th Int. Conf. on Data Engineering*, ICDE, pp. 440–447.

Chan, K. and Fu, A. W. (1999). Efficient time series matching by wavelets, *Proc. 15th Int. Conf. on Data Engineering*, ICDE, pp. 126–133.

Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1993). *Introduction to Algorithms*. The MIT Press.

Das, G., Lin, K., Mannila, H., Renganathan, G., and Smyth, P. (1998). Rule discovery from time series, *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining*, KDD, pp. 16–22.

Faloutsos, C., Jagadish, H. V., Mendelzon, A. O., and Milo, T. (1997). A signature technique for similarity-based queries, *Proc. Compression and Complexity of Sequences*, SEQUENCES.

Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases, *Proc. of the 1994 ACM SIGMOD Int. Conf. on Management of Data*, pp. 419–429.

Gusfield, D. (1997). *Algorithms on Strings, Trees and Sequences*. Cambridge University Press.

Guttman, A. (1984). *R*-trees: A dynamic index structure for spatial searching, *Proc. 1984 ACM SIGMOD Int. Conf. on Management of Data*, pp. 47-57.

Jagadish, H. V., Mendelzon, A. O., and Milo, T. (1995). Similarity-based queries, *Proc. 14th Symposium on Principles of Database Systems*, PODS, pp. 36–45.

Keogh, E. J. (1997). A fast and robust method for pattern matching in time series databases, *Proc. 9th Int. Conf. on Tools with Artificial Intelligence*, ICTAI, pp. 578–584.

Keogh, E. J. and Pazzani, M. J. (1998). An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback, *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining*, KDD, pp. 239–243.

Keogh, E. J. and Pazzani, M. J. (1999a). An indexing scheme for fast similarity search in large time series databases, *Proc. 11th Int. Conf. on Scientific and Statistical Database Management*, SSDBM, pp. 56–67.

Keogh, E. J. and Pazzani, M. J. (1999b). Scaling up dynamic time warping to massive datasets, *Proc. 3rd European Conf. on Principles of Data Mining and Knowledge Discovery*, PKDD, pp. 1–11.

Keogh, E. J. and Pazzani, M. J. (2000). A simple dimensionality reduction technique for fast similarity search in large time series databases, *4th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, PAKDD, pp. 122–133.

Keogh, E. J. and Smyth, P. (1997). A probabilistic approach to fast pattern matching in time series databases, *Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining*, KDD, pp. 24–30.

Keogh, E. J., Chakrabarti, K., Mehrotra, S., and Pazzani, M. J. (2001a). Locally adaptive dimensionality reduction for indexing large time series databases, *Proc. 2001 ACM SIGMOD Conf. on Management of Data*, 151–162.

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

Keogh, E. J., Lonardi, S., and Chiu, B. (2002). Finding surprising patterns in a time series database in linear time and space, *Proc. 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, KDD, pp. 550–556.

Kim, S., Park, S., and Chu, W. W. (2001). An index-based approach for similarity search supporting time warping in large sequence databases, *Proc. 17th Int. Conf. on Data Engineering*, ICDE, pp. 607–614.

Lee, S., Chun, S., Kim, D., Lee, J., and Chung, C. (2000). Similarity search for multidimensional data sequences, *Proc. 16th Int. Conf. on Data Engineering*, ICDE, pp. 599–609.

Mannila, H. and Ronkainen, P. (1997). Similarity of event sequences, *Proc. 4th Int. Workshop on Temporal Representation and Reasoning*, TIME, pp. 136–139.

Park, S., Chu, W. W., Yoon, J., and Hsu, C. (2000). Efficient search for similar subsequences of different lengths in sequence databases, *Proc. 16th Int. Conf. on Data Engineering*, ICDE, pp. 23–32.

Perng, C., Wang, H., Zhang, S. R., and Parker, D. S. (2000). Landmarks: a new model for similarity-based pattern querying in time series databases, *Proc. 16th Int. Conf. on Data Engineering*, ICDE, pages 33–42.

Rafiei, D. and Mendelzon, A. (1997). On similarity-based queries for time series data, *SIGMOD Record*, 26(2):13–25.

Sankoff, D. and Kruskal, J., editors (1999). *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. CSLI Publications, reissue edition.

Sellis, T. K., Roussopoulos, N., and Faloutsos, C. (1987). The R^+-Tree: A dynamic index for multi-dimensional objects, *Proc. 13th Int. Conf. on Very Large Database*, VLDB, p. 507–518.

Shatkay, H. (1995). *The fourier transform: a primer*. Technical Report CS-95-37, Brown University.

Wang, H. and Perng, C. (2001). The S^2-tree. An index structure for subsequence matching of spatial objects, *Proc. of the 5th Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, PAKDD, pp. 312–323.

Weber, R., Schek, H., and Blott, S. (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces, *Proc. 24th Int. Conf. on Very Large Databases*, VLDB, 194–205.

Wu, Y., Agrawal, D., and Abbadi, A. E. (2000). A comparison of DFT and DWT based similarity search in time-series databases, *Proc. 9th Int. Conf. on Information and Knowledge Management*, CIKM, pp. 488–495.

Yi, B. and Faloutsos, C. (2000). Fast time sequence indexing for arbitrary L_p norms, *The VLDB Journal*, pp. 385–594.

Yi, B., Jagadish, H. V., and Faloutsos, C. (1998). Efficient retrieval of similar time sequences under time warping, *Proc. 14th Int. Conf. on Data Engineering*, ICDE, pp. 201–208.