Similarity Search Based on Random Projection for High Frequency Time Series

Wei Wu [1,2]
[1] Graduate School
Chinese Academy of Sciences
Beijing 100039, China
wuwei@sia.cn

Jingtao Hu [2]
[2] Shenyang Inst. of Automation
Chinese Academy of Sciences
Shenyang, 110016, China
hujingtao@sia.cn

Abstract—**Similarity search in high frequency time series of domains as diverse as finance, marketing and industry has attracted much research attention recently. The main notions used in similarity search for time series are defined in a formal way. And a fast algorithm of similarity search based on random projection for high frequency time series is proposed. In order to achieve the high-level representation of time series, this algorithm uses the random projection method to map the original time series to the lower space. Then, the spatial data index structure such as R* tree is built using the high-level representation of the original time series, and the Euclidean distance is used as the similarity measurement. It is a fast similarity searching algorithm with high accuracy for high frequency time series. The experimental results demonstrate that the method is effective and efficient.**

Keywords—similarity search, random projection, high frequency time series, data mining

I. Introduction

Time series data which depict the trends in the observed value over time arise naturally in many real world domains as diverse as stock market, weather forecasts, medicine and industry. Similarity search is useful in its own right as a tool for exploring time series databases, and it is also an important essential subroutine in many data mining applications such as clustering, classification and association rules discovery.

There has been an explosion of interest in time series similarity search recently. Agrawal et al. [1] developed one of the first solutions to this problem. They transformed the time series to the frequency domain by using DFT. And later, they reduced the number of dimensions to a feasible size by storing the first few frequency coefficients. Faloutsos et al. [2] introduced Generic Multimedia INdexIng (GEMINI) framework which can exploit any dimensionality reduction method to allow efficient indexing. Eamonn Keogh et al. proposed Piecewise Aggregate Approximation, Adaptive Piecewise Constant Approximation, Symbolic Aggregate Approximation and a bit level time series representation with implications [3][4][5][6][7]. Aiguo Li et al. [8] addressed a systemic method of time series similarity searching based on Piecewise Polynomial Representation that is to map each sub-sequence into a small set of multidimensional rectangles in feature space which is spanned by base of linear polynomial. Hui Xiao et al.[9] proposed Feature Points Segmentation and the feature points segmented time warping distance defined based on it.

Most existing time series similarity searching methods until now have focused on the slow-altered (low frequency) time series such as year/month/day stock exchange data, statistic data of commercial goods and the Web traffic etc., in that finance and marketing are the most active application domains of the data mining technique. However, with the requirement raising of analysis precision and the application domains widely spreading of data mining, the high frequency time series such as the transaction by transaction data or tick by tick data in the trades and quotes database of financial market and the vibration, current or voltage senor data sampled from the industry field etc. is becoming another important point of time series similarity searching research. The traditional methods can not handle the high frequency time series suitably, for this kind of time series are massive, super-multidimensional, frequently volatilizable in short term and noisy which are different from the low-altered ones.

In this paper we defined the main notions used in similarity search for time series, and proposed a novel fast algorithm of similarity search based on Random Projection for high frequency time series. Random Projections (RP) have recently appeared as a tool for dimensionality reduction and have been successfully used for image and text data. The similarity search method introduced here uses the RP method to map the original time series to the lower space, and then indexes the high-level representation of the time series using the spatial data index structure R* tree. The analysis was done from synthetic high frequency time series data and vibration severity data collected from rotating machines and the performance of the new approach indicate it is not sensitive to impulse noise and suitable to achieve high quality high frequency time series similarity search.

The render of the paper is organized as follows. Section II defines the main notions of time series similarity search systemically. Section III introduces the theory of the basic Random Projections method. Section IV proposes a time series similarity search method based on RP. And in Section V, the application of the method to a synthetic dataset and a rotating

978-1-4244-1674-5/08 /$25.00 ©2008 IEEE

machine observations dataset is demonstrated. Section VI concludes the paper.

II. Time Series similarity search

A time series X can be considered as a point in p-dimensional space. More formally, we describe it by the following definition:

Definition 1 A time series X is defined as a data sequence on the time axis, $X=\{x_1, x_2, ..., x_p\}$, where each data element $x_i=(t_i,a_1,a_2,\ldots,a_m)$, $1 \le i \le p$; a_j□R, $1 \le j \le m$, m is the dimension of X and equal to 1 in general, t_i□R, t_i-t_{i-1}=Δ, Δ is a constant more than zero, $|X|$=p is the length of X.

Definition 2 Given a set $S_n = \{X_1, X_2, \ldots, X_n\}$, where $X_1, X_2, \ldots, X_n$ are n time series, we denote S_n is a time series set. And S_n^p denotes the set of which length of any element are equal viz. $|X_i| = p$ for any i.

Definition 3 Given a time series X, $|X| = p$, F = Feature$(X) = \{f_1, f_2, ..., f_k\}$, where f_i is a data point in some feature space, and k<<p in general, we denote F is a representation of the original time series X, k is the representation length, Feature(X) is the representation function of X.

Definition 4 Given a time series set S_n , $|X_i|$=p_i , $1 \le i \le n$, Q is a query time series, $|Q|$=q, $q \le p_i$. F_X=Feature (X), F_Q=Feature (Q), a similarity measure model d and a searching strategy *find* (.), time series similarity search problem is defined as:

$$S_{TR} = \{X \in S_n \mid find(d(F_Q, F_X),\ S_n)\} \tag{1}$$

That is to find the time series set S_{TR} consist of the time series similar to Q among the time series S_n based on the similarity measurement d. Here the elements $X_{\mathrm{TR}i}$ of S_{TR} have equal length.

There are essentially two ways the time series data might be organized:

- *Whole Matching*, here $q = p_i$;
- *Subsequence Matching*, here $q < p_i$.

It is possible to convert subsequence matching to whole matching by sliding a "window" of length n across the longer time series.

The discovery of relation between time series involves mainly three tasks:

1) Similarity Measures: define a similarity measure between time series, in order to determine if they match. Generally we use some distance metric to express similarity, and choosing it must depends on the application domains, analysis task, and ever the data itself. Euclidean distance is the simplest and most widely used similarity measurement.

2) Indexing: to build index of the massive time series database in order to promote the searching efficiency. As mentioned a time series can be considered as a point in p-dimensional space. This immediately suggests that time series could be indexed by Spatial Access Methods such as the R-tree and its many variants, k-d tree, quad tree and grid files.

3) Representations: the representation and modeling of the data sequence in a suitable form. The indexing efficiency will be falling down with the increase of the data dimensionality. The ability of the R tree might be similar to the sequence scan when the dimensionality exceeds 6-20, and most spatial access methods begin to degrade rapidly at dimensionalities greater than 8-12[3]. However, usually the length of time series must be head and shoulders above the range, indexing time series using spatial access methods directly might lead the dimensionality curse. Therefore, it is necessary to choose a suitable representation of time series to achieve dimension reduction and feature extraction. Here Feature (.) function should remain the similarity relation between the original time series as perfectly as possible.

The common time series representation methods include: Discrete Fourier Transform (DFT), Discrete Wavelet Transform (DWT), Singular Value Decomposition (SVD), and so on [16].

There are two important kinds of queries that we would like to support in time series database:

- *Range queries,* return all time series within an epsilon of the query time series;
- *k nearest neighbors,* return the k closest time series to the query time series.

This paper focus on the k nearest neighbors query of subsequence matching.

III. Random projections

Random Projections have recently emerged as a powerful method for dimensionality reduction. Theoretical results indicate that the method preserves distances quite nicely. And this method has been successfully used for images and text documents while indicating that it is not sensitive to impulse noise.

In random projection, the set of points of size n in original p-dimensional Euclidean space is projected to a k-dimensional (k<<p) subspace through the origin, using a random $p \times k$ matrix R whose columns have unit lengths in order to achieve dimension reduction. The mapping process is:

$$X_{n\times k}^{RP} = X_{n\times p} R_{p\times k} , \tag{2}$$

where $R_{p\times k}$ is the random matrix, $X_{n\times p}$ is the original observations set of size n in p-dimension, and $X_{n\times k}^{RP}$ is the projection in k-dimension subspace.

The idea of this method is motivated by the Johnson and Lindenstrauss (JL Theorem) states.

Lemma 1 Johnson and Lindenstrauss embeddings.

Given $\varepsilon > 0$ and an integer *n*, let *k* be a positive integer such that $k \geq k_0 = O(\varepsilon^{-2} \log n)$. For every set *P* of *n* points in $\Re^p$ there exists $f : \Re^p \rightarrow \Re^k$ such that for all $u, v \in P$:

$$(1-\varepsilon)\| u-v\|^2 \leq \| f(u)-f(v)\|^2 \leq (1+\varepsilon)\| u-v\|^2 \tag{3}$$

This means a *p*-dimensional point set *P* can be embedded into *k*-dimensional space where *k* is independent of *p*. Euclidean distance are preserved within a factor $(1 \pm \varepsilon)$. Further, this map can be found in randomized polynomial time.

In the last few yeas, JL Theorem has been useful in solving a variety of problems such as ε -approximate nearest neighbor problem, clustering and the context of "data-stream" computation etc.

Dasgupta and Gupta [13] present a simpler proof of the JL Theorem, giving tighter bounds on ε and *k*, as follows:

$$k \geq 4 \times (\frac{\varepsilon^2}{2} - \frac{\varepsilon^3}{3})^{-1} \ln(n) \tag{4}$$

They also indicate that a matrix whose entries are normally distributed represents such a mapping with probability at least 1/*n*, and therefore doing *O*(*n*) projections will result in projection with an arbitrarily high probability of preserving distances.

There are many proposals for the choice of the random matrix. Typically, the elements in $R_{p\times k}$ are Gaussian distributed. Performing such a projection, while conceptually simple, is non-trivial, especially in a database environment. Moreover, its computational cost can be prohibitive for certain applications. Achlioptas [14] shows that there are simpler ways of producing random projections.

Theorem 1 Given a set *P* of *n* points in $\Re^p$ (in form of an matrix $X_{n\times p}$), choose $\varepsilon, \beta > 0$ and $k_0 = \frac{4+2\beta}{\varepsilon^2/2-\varepsilon^3/3}\log n$. For integer $k \geq k_0$, let *R* be a random matrix with *R*(*i*, *j*)=r_{ij}, $1 \leq i \leq p$, $1 \leq j \leq k$, where { r_{ij} } are independent random variables from either on of the following two probability distribution:

$$\mathrm{r}_{ij} = \begin{cases} +1 & with\ probability\ 1/2 \\ -1 & with\ probability\ 1/2 \end{cases}, \tag{5}$$

or

$$\mathrm{r}_{ij} = \sqrt{3}\begin{cases} +1 & with\ probability\ 1/6 \\ 0 & with\ probability\ 2/3 \\ -1 & with\ probability\ 1/6 \end{cases} \tag{6}$$

Let $$X^{RP}_{n\times k} = \frac{1}{\sqrt{k}} X_{n\times p} R_{p\times k} \tag{7}$$

Let $f : \Re^p \rightarrow \Re^k$ map the i^{th} row of $X_{n\times p}$ to the i^{th} row of $X^{RP}_{n\times k}$, $1 \leq i \leq n$. With probability at least $1-n^{-\beta}$, for all$u, v \in P$:

$$(1-\varepsilon)\| u-v\|^2 \leq \| f(u)-f(v)\|^2 \leq (1+\varepsilon)\| u-v\|^2 \tag{8}$$

As in lemma 1, the parameterε controls the accuracy in distance preservation, while here β controls the probability of success.

IV. TIME SERIES SIMILARITY SEARCH BASED ON RANDOM PROJECTIONS

A. Time Series Representation Algorithm

A time series *X*, |*X*|=*p*, might be considered as a point in $\Re^p$ space. The dimensionality of all the time series in the set S_n and the query time series *Q* must be reduced by the same representation function Feature (.). The pseudo code with a detailed explanation of the time representation algorithm Feature_RP is presented as Algorithm 1.

Algorithm 1 Time series representation

```
Procedure [$F_Q$, $S^k_m$] = Feature_RP($Q$, $S_n$, $k$, $w\_step$)
Input: $Q$: query time series; $S_n$: time series set; $k$: length of
       representation; $w\_step$: step of gliding window
Output: $F_Q$: the representation of $Q$; $S^k_m$: the set composed
        of the representation of the time series of $S_n$
    $l_Q$ = Length($Q$);
    $TB_{\text{original}}$ = SegmentWin($S_n$, $l_Q$, $w\_step$);
    $m$ = Count ($TB_{\text{original}}$); // the count of time series in $TB_{\text{original}}$
    $X_{(m+1)\times p}$ = OriginalMatrix($TB_{\text{original}}$, $Q$);
    $R_{p\times k}$ = RP_SparseMatrix ($k$, $p$);
    [$F_Q$, $S^k_m$] = $X_{(m+1)\times p} \times R_{p\times k}$;
```

For every incoming query *Q* the representation routine Fearture_RP is invoked with: a sample time series setS_n; the length of the final representation of the time series *k*; and the step of the gliding window for segmentation *w_step*. Fearture_RP uses SegmentWin to obtain a database TB_{original} of time series with the length equal to *Q* by a step *w_step* gliding window. OriginalMatrix create a matrix size of $(m+1)\times p$

composed of Q and all the time series in TB_{original}. Finally, the representation of sample time series and query time series Q are computed based on (6) and (7). F_{x1}, Fx_2,…, Fx_m the row vectors of S_m^k are the representation of the sample time series in TB_{original}.

Before the query, every time series to be used including Q must be pretreated by noise removal and normalization. The entry structure of database TB_{original} and S_m^k is:

$$T_{id}, t_{id}, x_{id}, x_1, x_2, ..., x_j$$

Where j is p and k respectively, T_{id} is the ID of original time series X_i, and t_{id} is the ID of the subsequence segmented from the origin time series.

The lengths of the query time series Q and the sample time series in TB_{original} is successfully decreased from p to k, $k<<p$, by the similarity searching algorithm Feature_RP based on Random Projections. That achieves the map $f:\Re^p \to \Re^k$ and the effect of the representation function Feature (.).

B. Similarity Searching Algorithm

The metric space index structure R* tree is a variation of the initial R tree. R trees are hierarchical data structures based on B+ tree. They are used for the dynamic organization of a set of k-dimensional geometric objects representing them by the Minimum Bounding *k*-dimensional Rectangles (MBRs). We apply R* tree to index the random projections representation of the sample time series $F_{x1}, F_{x2}, \ldots, F_{xm}$ as follows:

All leaves reside on the same level. Each leaf contains pairs of the form (S_{id}, F_x), where $S_{id} = T_{id} + t_{id}$ is the only id of the subsequence in the TB_{original}, $F_X = (f_{x1}, f_{x2}, ..., f_{xk})$ is a vector with the elements which is the coordinates of subsequence time series as a point in the *k*-dimensional space.

Each non-leaf node of the R* tree corresponds to the minimum MBR that bounds its children. The leaves of the tree contain pointers to the database objects, instead of pointers to children nodes. The nodes are implemented as disk pages. The MBRs that surround different nodes may be overlapping. Besides, an MBR can be included in many nodes, but can be associated to only one of them. This means that a spatial search may visit many nodes, before confirming the existence or not of a given MBR. Every non-leaf node contains pairs of the form (MBR$_i$, Child$_i$), where Child$_i$ is a pointer to a child of the node and MBR$_i$ is the MBR that contains spatially the MBRs contained in this child.

The Random Projections method is a mapping operation in the Euclidean space. The Euclidean distance relations among the multidimensional points in this space can be preserved nearly perfectly. In this paper we choose the Euclidean distance function as the similarity measurement of time series d. Given two time series $X = x_1, x_2, ..., x_p$ and $Q = q_1, q_2, ..., q_p$ with equal length, their Euclidean distance is defined as:

$$d(X,Q) = \sqrt{\sum_{i=1}^{i=p} (x_i - q_i)^2} \qquad (9)$$

The time series *k*-nearest neighbors searching algorithm based on Feature_RP procedure and the R* tree index structure Find_RP is described in Algorithm 2.

Algorithm 2 Similarity Searching Algorithm

```
Procedure  $S_{TR}$ = Find_RP($S_m^k$, $F_Q$, $K$)
Input: $S_m^k$: a representations set of the sample time series;
       $F_Q$: the representation of the query time series $Q$;
       $K$: the number of the nearest neighbors
Output: $S_{TR}$: the set of the results of $k$-nn classify

    R_index = RstarIndexBuildup($S_m^k$);
    condition = $F_Q \subset MBR$;
    S_MBR = SearchIndex(R_index, condition);
    $S_{TR}$ = zeros ($K$); //a array of $K$ zero objects.
    while (S_MBR){
        // $F_{Xi}$ is the element time series of S_MBR
        $D = d(F_{Xi}, F_Q)$;
        if (D > S$_{TR}$[0].distance) {
            S$_{TR}$[0].distance = $D$;
            S$_{TR}$[0].timeseries = $F_{Xi}$;
            SortArray(S$_{TR}$, ASC);
        }
    }
    return S$_{TR}$;
```

The input of the Find_RP algorithm S_m^k and F_Q are the results of the Feature_RP procedure. The algorithm search K nearest neighbors of the query time series through the R* tree structure R_index built up with the sample time series in S_m^k. S_MBR is the time series whose rectangles overlap the rectangle of F_Q in R_index. For every element F_{Xi} of S_MBR, if D the distance between it and F_Q is larger than the minimum value of the K nearest neighbors so far, the S_{TR} needs to be update with F_{Xi} and D.

Given a sample time series set S_n^p, mapping it to S_n^k needs the multiplication by the random matrix $R_{p\times k}$. So forming the random matrix $R_{p\times k}$ and projecting the $p \times n$ data matrix S_n^p into k dimensions is of order O (pkn), and if the random matrix $R_{p\times k}$ is sparse with about c nonzero entries per row, the complexity is of order O (ckn). For high frequency time series, the lengths of them are huge in general, the values of p will outclass the values of k. So the high frequency time series representation algorithm should be of order O (cn).

Reference [3] introduces a dimensionality reduction technique called PAA (Piecewise Aggregate Approximation) and theoretically and empirically compares it to the SVD (Singular Value Decomposition), DFT (Discrete Fourier Transform) and DWT (Discrete Wavelets Transform) methods. The time complexity of this method appears to be $O(nm)$, where m is the size of sliding window. Thus it can be seen the Random Projections similarity search methods proposed in this paper could be as fast as the PAA method. Moreover, when the random matrix is a sparse matrix with the form of (6), the searching method based on Random Projections may save 2/3 computational time, which has higher efficiency for massive high frequency time series similarity search.

V. Experiments for Similarity Search Method Based on Ranndom Projections for High Frequency Time Series

A. Data Collection

We performed all the experiments with two kinds of time series datasets: synthetic high frequency time series dataset (dataset A1 and dataset A2) and the vibration signal data observed from a rotor test rig (dataset B).

1) The time series data in the dataset A are obtained synthetically by the following formula:

$$x_t = A_1\sin(2\pi f_1 t+\phi_1)+A_2\sin(2\pi f_2 t+\phi_2)+z_t \tag{10}$$

where A_1 and A_2 are 100 and 50 respectively, f_1 and f_2 are 256 and 512 respectively, ϕ_1 and ϕ_2 are 0 and 32 respectively. z_t is white noise, and Time series $X=\{x_t \mid t=1,2,...,n_A\}$. Calculate by (10) to obtain two time series set S_{500}^{1024} (dataset A1) and S_{500}^{2048} (dataset A2) with two sliding windows respective size of 1024 and 2048 and step of 4096 and 2048 respectively. The length of elements in the set is 1024000.

2) The dataset B is vibration severity observations dataset. Experiments were carried out on a rotor test rig, which can simulate the operation status of many rotating machine equipment, such as gas turbines, compressors, pump etc. It is composed of a rotor and a stator, a driving motor, journal bearings and couplings, as shown in Fig. 1. Vibration signals were collected from the rotor system using magnetoelectric sensors and non-contact eddy-current sensors at a sampling of 2 kHz.

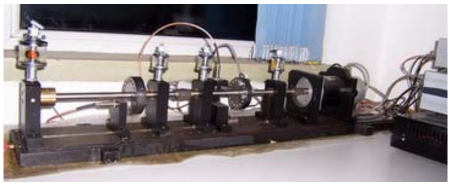

Figure 1. The rotor test rig

The vibration severity measurement, the root mean square (RMS) of the vibration speed is the ISO recommended method for general machine condition monitoring. Below is the equation that is used to calculate Vrms the vibration severity value of a vibration speed signal data series, x_n over length N.

$$V_{rms}=\sqrt{\frac{1}{N}\times\sum_{n=1}^{N}x_n^2} \tag{11}$$

B. Time series similarity searching results

We made a large amount of numerical tests with the two types of data sets, the following are the forested results of two sets of each them (we achieved similar results on the other data sets).

Firstly, we compare the running times of the method proposed in this paper (RPS for short) and the PAA method in [3]. Select one time series from the datasets stochastically, and perform 1-nn query on the corresponding dataset. Let RPSTime and PAATime represent the running time of RPS method and PAA method that have run 100 times. Choose the length range of representation to be 6-20. The query time comparison results are showed in Fig.2 and Fig.3.

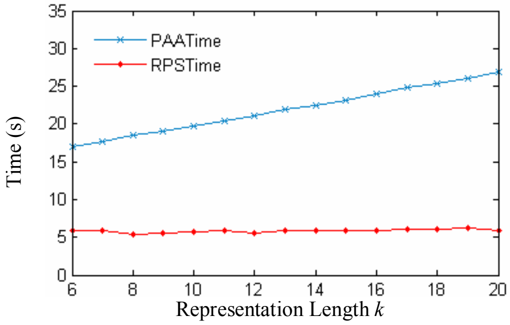

Figure 2. The comparison of RPSTime and PAATime based on dataset A1

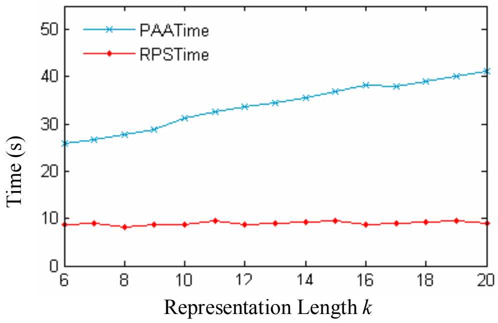

Figure 3. The comparison of RPSTime and PAATime based on dataset B

We can see that RPS method is faster than PAA method. Moreover, the longer the representation length k selected, the more running time of PAA needs. But keeping a low level, the running time of RPS is not sensitive to k.

To evaluate the similarity searching accuracy, we took one nearest neighbor query for example. Select 200 time series to be the random query time series Q, and choose the length range of representation to be 6-36. Fig.4 illustrates the accuracy rate results of RPS method with different data and representation length.

The distances between multidimensional points are preserved approximately through the mapping, because

Random Projections method is a mapping with probability condition restrictions. So there exist certain extent calculation errors. However, Fig.4 demonstrates that the accuracy rate of RPS method is promoted evidently with the increase of the representation length. The accuracy achieves higher level (above 81.5%) at representation length points of 16, 17, and 19. It has been confirmed above that the capability of performing fast of RPS method is not sensitive to the representation length, so that RPS method can achieve a fast high frequency time series similarity search with high accuracy rate around the representation length of 20.

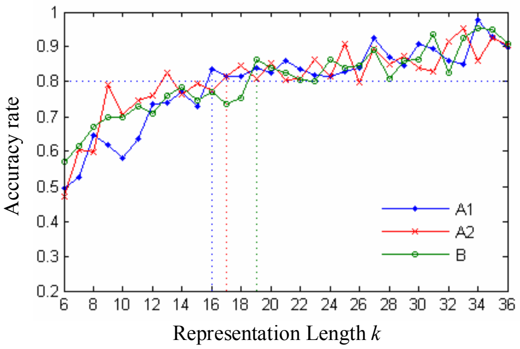

Figure 4. The accuracy rate of the RPS method

VI. CONCLUSIONS

This paper defines the main notions used in similarity search for time series and proposes a novel similarity search method for high frequency time series that are generally massive, super-multidimensional, noisy and frequently volatilizable in short term which are different from the low-altered ones. This approach applies improved Random Projections method that has advantages of lower computational cost and powerful ability of preservation of Euclidean distance between multidimensional points to be the high level representation of time series, and then searches to obtain the similar time series to the queries through a spatial data structure R* tree built up with the RP representation of time series. The accuracy and efficiency of this new approach is validated and illustrated by synthetic high frequency time series datasets and a vibration severity dataset observed from a rotating machine. The results show that keeping high accuracy, the similarity search approach can achieve a fast query for high frequency time series. In future work we intend to further apply this proposed approach to the real field data to validate the feasibility of it.

ACKNOWLEDGMENT

The authors wish to acknowledge the substantial financial support of the National High Technology Research and Development program of China (863 program) under Grant No. 2003AA414210 and the Shenyang Science and Technology Program under Grant No. 1053084-2-02.

REFERENCES

[1] R. Agrawal, C. Faloutsos, A. Swami, “Efficient similarity search in sequence databases”. Proceedings of the 4th international Conference on Foundations of Data Organization and Algorithms, Chicago, IL, 1993: 69-84.

[2] C. Faloutsos, M. Ranganathan, Y. Manolopoulos. “Fast sub-sequence matching in time-series databases”. Proceedings of the ACM SIGMOD International Conference on Management of Data. Minneapolis, MN, 1994:419-429.

[3] E. Keogh, K. Chakrabarti, M. Pazzani et al. “Dimensionality reduction for fast similarity search in large time series data bases”, Proceedings of the ACM SIGMOD International Conference on Management of Data. New York. 2001:151-162

[4] E. Keogh, M. Pazzani. “An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback”. Proceedings of the 4th International Conference of Knowledge Discovery and Data Mining. AAAI Press, 1998:239-241

[5] E. Keogh, K. Chakrabarti, S. Mehmtra et a1. “ Locally adaptive dimensionality reduction for indexing large time series databases”. ACM SIGM0D 2001, Santa Barbara, California, 2001.

[6] C. Ratanamahatan, , E. Keogh, ,Bagnall, A.J. et al. “A novel bit level time series representation with implications for similarity search and clustering”. Proc of the Pacific-Asia Conference on Knowledge Discovery and Data Mining. Hanoi, Vietnam, 2005:771-777.

[7] E. Keogh, M. Pazzani. “An indexing scheme for fast similarity search in large time series databases. Scientific and Statistical Database Management”, Eleventh International Conference on. 1999:56-67

[8] Aiguo Li, Zheng Qin. “Dimensionality reduction and similarity search in large time series database”. Chinese Journal of Computers. 2005, 28(9):1467-1475

[9] H. Xiao, Y. Hu. “Data Mining Based on Segmented Time Warping Distance in Time Series Database”. Journal of Computer Research and Development 2005, 42(1):72:78.

[10] Rong Jiang, DeyiLi. Similarity search based on shape representation in time-series data sets. Journal of Computer Research and Development, 2000, 37(5):601-608.

[11] Witten I., Frank E.. Data Mining: Practical Machine Learning Tools and Technique. Morgan Kaufmann: 2 edition. 2005.

[12] W. B. Jonnson and J. Lindenstrauss. “Extensions of Lipschitz mappings into a Hibert space”. Conference in modern analysis and probability (New Haven, Conn., 1982), Amer. Math. Soc., Providence, R.I., 1984.

[13] S. Dasgupta and A. Gupta. “An elementary proof of the Johnson-Lindenstrauss lemma”. Technical Report TR-99-006, International Computer Science Institute, Berkeley, California, USA, 1999.

[14] D. Achlioptas. “Database-friendly random projections”. ACM Symposium on Principles of Database Systems, 2001:274-281.

[15] E. Keogh, S. Kasetty. “On e need for time series data mining benchmarks: A survey and empirical demonstration”. Proceedings of the 8the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, 2002:102-111.

[16] E. Keogh. A Tutorial on Indexing and Mining Time Series Data. The 2001 IEEE International Conference on Data Mining, November 29, San Jose