Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp378-383)

Spider Algorithm for Clustering Time Series

SHOHEI KAMEDA, MASAYUKI YAMAMURA
Department of Computational Intelligence and Systems Science
Tokyo Institute of Technology
4259 Nagatsuta-cho, Midori-ku, Yokohama, 226-8503
JAPAN

Abstract: In proportion to the rapid development of information technology, time series are today accumulated in finance, medicine, industry and so forth. Therefore, an analysis of them is an urgent need for these applications. As solving these problems clustering time series has much been paid attention. The similarity for the clustering is commonly measured with Euclidean distance and dynamic time warping. In this paper we propose an innovative and novel algorithm for clustering multivariate time series. The algorithm is called "Spider Algorithm". We experimentally show that the similarity from spider algorithm is superior to Euclidean distance or warping path on dynamic time warping, especially when many clusters exist.

Key–Words: Clustering, Multivariate time series, Biological inspired algorithms, Data mining

1 Introduction

In proportion to the rapid development of information technology, digital data are piled up over the last decade. One of the data is time series, which is a collection of observations made sequentially in time. Since time series are today accumulated in finance, medicine, industry and so forth, an analysis of them is an urgent need for these applications. As solving these problems clustering time series has much been paid attention by data miners. Clustering is a data mining tool that divides data into clusters based on a similarity[1]. Many algorithms for time series clustering have been proposed[2]. Generally speaking, the performance of clustering is how to choose a similarity. Euclidean distance is often used as a similarity because this distance is very easy to implement and calculate. In terms of time series, however, Euclidean distance may fail to produce an intuitively correct clusters because it is very sensitive to small distortion in the time axis[3]. Another well-known similarity is a warping path on dynamic time warping. Dynamic time warping is developed originally for speech recognition in the 1970s[4]. This alignment technique is very accurate but computationally complex. For that reason, most methods point to a reduction of the calculation, not considering the superior way to a warping path. Moreover, in spite of existing a large number of methods, most of the present methods focus on only one dimension, not multivariate data. When those methods make clusters in multivariate data, the sum of similarity is calculated. As a matter of course, it is difficult for clustering to obtain a similarity by the sum because of the accumulation of error of the each dimension.

In this paper we propose an innovative and novel algorithm for clustering multivariate time series. The algorithm is called "Spider Algorithm" that uses the relation of each dimension which the present methods ignores. As the name represents, our algorithm is inspired by spider. There are many biological inspired approaches, from genetic algorithm to ant colony systems, which have recently appeared in machine learning. The advantage of those algorithms is intuitively understood by human. In spite of the complexity of algorithms users can apply the methods without hesitation.

We experimentally show that the similarity from spider algorithm is superior to Euclidean distance or a warping path on dynamic time warping, especially when many clusters exist.

The rest of this paper is organized as follows. In section 2 we give a brief explanation of clustering multivariate time series. In section 3 we explain spider algorithm in detail. In section 4 we conduct experiments. In section 5 we discuss the further possibility of spider algorithm.

2 Preliminaries

In this section we provide basic concepts of time series and clustering. Multivariate time series is defined as,

$$X = \{X_1, X_2, , , X_N\}$$

X is a set of time series which has N elements. An element *k* in *X* is,

$$X_k = (\mathbf{x_1^k}, \mathbf{x_2^k}, , , \mathbf{x_t^k})$$

where *t* is the time which is observed at certain intervals. And a multivariate data $\mathbf{x_t^k}$ is a vector as follows,

$$\mathbf{x_t^k} = (x_{t1}^k, x_{t2}^k, , , x_{tn}^k)$$

n represents the dimension of the data. For example, a climate data can be one of multivariate time series. This is observed at the places *X*=$\{X_1, X_2, X_3\}$=$\{$Tokyo, New York, London$\}$ and kept watch every hour. In Tokyo 24 hour data, X_1=$(\mathbf{x_1^1}, \mathbf{x_2^1}, , , \mathbf{x_{24}^1})$ is composed of three dimensional data, $\mathbf{x_1^1}$=(temperature, humidity, air pressure)=(20, 80%, 1013hPa). Note that we treat quantitative variables in our method.

The objective of clustering time series is to separate data from a set of *X* to match the human intuition or to get new knowledge. In above example, for instance, two clusters $\{X_1\}$ and $\{X_2, X_3\}$ would be built in comparison to the climate data of the cities $(\mathbf{x_1^k}, \mathbf{x_2^k}, , , \mathbf{x_{24}^k})$. The criterion of clusters is similarities. In case of Euclidean distance, the similarity between X_1 and X_2 is calculated by sum of squares, $\sqrt{\sum_{t=1}^{24}(\mathbf{x_t^1} - \mathbf{x_t^2})^2}$. In time series clustering, most of the methods are to compare each data point and compute similarity and then make clusters based on the similarities. We omit an explanation about dynamic time warping here because of the limitation of space, but the idea of comparing each point is same as Euclidean distance.

3 Spider Algorithm

The concept of spider algorithm is that time series is regarded as a bug's path and spiders capture bugs with their webs, then the combination of which spiders capture them produces a similarity.

Fig.1 shows that a concise flowchart of spider algorithm. First of all, standard scores are calculated by standardizing all data. This is for generalizing parameters which is used in spider algorithm. The next process is called "Adaptation process" that spiders capture bugs over the generations. This process outputs "Capture lists" at the end of each generation. A capture list is used to obtain the similarity, explain details later.

We begin to explain the details of spider algorithm from here.

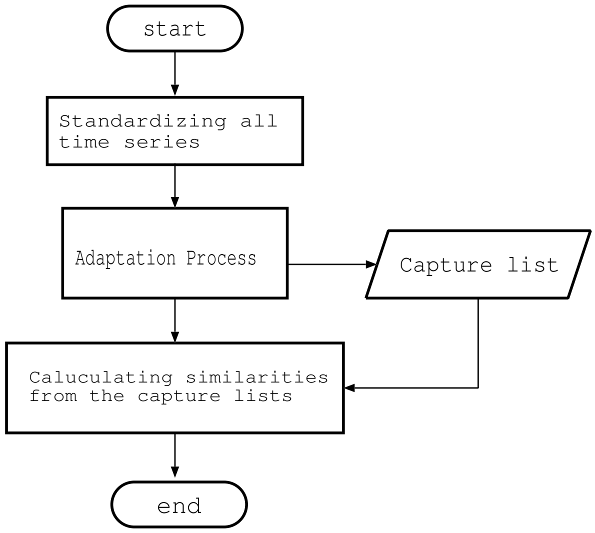


Fig. 1: A brief flowchart

3.1 Bug

As we mentioned above, we think of a time series as a bug's path or a path of spider's prey. We firstly define the terminologies in Spider Algorithm. *X* is a set of bugs. X_k is the path of the bug *k*. $\mathbf{x_t^k}$ is the position of the bug *k* at the time t. We use the words "A bug flies". This meaning is to input X_k into the adaptation process by sequential order $\mathbf{x_1^k}, \mathbf{x_2^k}, , , \mathbf{x_t^k}$.

Before the adaptation process all bug's paths, from X_1 to X_N, are standardizing by,

$$Z = \frac{X - \mu}{\sigma}$$

where μ is mean and σ is standard deviation of each dimension. The purpose for calculating standard scores of time series is to fix the parameters that used in spider algorithm in any situations.

At the beginning of each generation, the bugs in *X* are randomly sorted and fly. All N bugs fly only once during one generation. Therefore, the period of one generation is the time that all N bugs flied.

3.2 Adaptation Process

The adaptation process is that some number of the spiders move around in the multidimensional space in order to capture flying bugs. The spiders can produce their offspring for obtaining more bugs to the next generation. At the end of a generation this process outputs a capture list which is for calculating the similarity.

In this section, we explain the behavior of a spider, how to take over to the next generation and what a capture list is.

Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp378-383)

3.2.1 Spider

A spider has four characteristics listed below.

- A spider has its web to capture bugs.
- A spider adjusts its web after the bug got through the web.
- A spider moves to the place to hunt bugs till they get a bug.
- A spider fight with the other spiders if they intrude in his territory. The winner is alive and the loser is dead.

Spiders capture bugs with their webs. The web is a hyperplane in a multidimensional space. A hyperplane is described by equation,

$$\mathbf{a} \cdot (\mathbf{y} - \mathbf{p}) = \mathbf{0} \tag{1}$$

where $\mathbf{a}$ is an axis and $\mathbf{p}$ is the position of the spider. This equation means a spider needs an axis and a position to express spider's web. Therefore, the attribute of a spider includes an axis and a position. On the other hand, an instant path of the bug is explain by a straight line,

$$\mathbf{y} = s \cdot (\mathbf{x^k_t} - \mathbf{x^k}_{t-1}) + \mathbf{x^k_{t-1}} \tag{2}$$

By calculating the equations (1) and (2), s can be solved. If $0 < s < 1$, the web crossed with the bug's path. After making the judgment that the web is the dividing point of the bug's path, two procedures continue in order to make sure of the capture. *1)* The intersection $\mathbf{y}'$ is compared with the radius r of the web.

$$\sqrt{(\mathbf{p} - \mathbf{y}')^2} < r \tag{3}$$

The intersection is calculated by substituting *s* into (1) or (2). 2) The angle between bug's path and the web compared with the maximum angle θ,

$$\frac{(\mathbf{x^k_t} - \mathbf{x^k_{t-1}}) \cdot \mathbf{a}}{|\mathbf{a}||\mathbf{x^k_t} - \mathbf{x^k_{t-1}}|} < \cos\theta \tag{4}$$

After the evaluation of *s* the capture is accomplished when the equations (3) and (4) are satisfied simultaneously. Owing to survey *s*, both (3) and (4) doesn't need for all positions of the spiders, thus the computational cost can be reduced. Fig.2 shows a image of the web explained the first feature of a spider.

The second feature is a mechanism for the adjustment of the web. A spider adjusts its web in order to capture more bugs. The way of the adjustment is that the direction of the axis moves to the bug's path.

$$\mathbf{a} \leftarrow \mathbf{a} + ((\mathbf{x^k_t} - \mathbf{x^k_{t-1}}) - \mathbf{a}) \times f \tag{5}$$

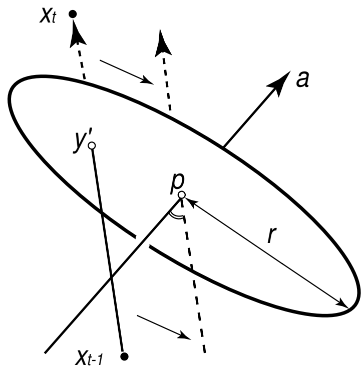


Fig. 2: A image of the web

where *f* is the rate of the adjustment. (5) makes the angle be small for the next bug which possibly exists. Thus, the spider is expected to capture bugs in the future.

The third feature is to move. Spiders in nature move randomly to find the place for making their web. In the computer, if spiders move randomly the convergence is difficult. We implement the movement of the spider as follows:

$$\mathbf{p} \leftarrow \mathbf{p} + (Random(\mathbf{X_k}) - \mathbf{p}) \times m \tag{6}$$

where *m* is the rate of movement. *Random()* chooses one position of the bugs in $\mathbf{X_k}$ randomly. Once the spider capture a bug, the spider stops until the end of generation.

The forth one is a spider's territory. When spiders come close to each other, spiders start to battle. The decision of loser or winner is to compare the number of capture The winner is alive and the loser is dead. To reduce population of spiders is to reduce computational costs.

3.2.2 Generation Change

The idea of the generation change is to keep the best individuals. The best individuals mean that the spiders can make differences of the data. In the view of clustering we need those spiders. At the end of each generation a assessment for the spiders is performed. The assessment has two roles. One for selection of the best spiders, two for output of a capture list. The way of the assessment are,

1. Fix the position of all spiders during the assessment.
2. Input all data by sequential order $X_1, X_2, , , X_N$ and output which spiders capture them. This output is a capture list.

Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp378-383)

3. The spiders that could capture bugs become the candidates for the next generation.

4. Choose the parents that could capture a few bugs from the candidates.

5. The parents produce children around their positions following a normal distribution.

You might wonder the forth selection of the parents. In natural spiders the spiders that catch a large number of bugs can be parents. However, in terms of clustering those spiders can not make difference. The spiders taken a fewer bug is more precious for clustering.

Another important part from system side is normalizing initial position. Spider algorithm has as a same drawback as the agent system. This well-known defect is that the agent system depends on the initial position. To avoid the problem of initial position, the repetition of the calculation is performed. Instead of this, our spider algorithm repeats more sophisticatedly as a generation change.

3.2.3 Capture list

Table.1 shows a sample of a capture list. We intend to form clusters from the time series data X_1 to X_6. The right next to the data, X_k, is the numbers of the spiders that could capture the bugs during the assessment. The bug X_1 was captured by 8 spiders, number 16, 15, 7, 11, 24, 25 and 0.

Table 1: A capture list

X_1: 16,15,7,11,24,25,0
X_2: 16,7,25,12,4,11
X_3: 16,15,0,11
X_4: 1,7,11,25,21
X_5: 1,7,11,25,20
X_6: 1,11,21,23

3.3 Similarity

Each generation outputs a capture list. We obtain similarity from those capture lists. We achieve it by using rank correlation[7]. Rank correlation is a method of finding the degree of association between two variables. This is calculated using the ranks of the observation, not their numerical values. This method is useful when the data are not available in numerical form but information is sufficient to rank the data. There are two major rank correlations known as Spearman's correlation coefficient(Spearman's ρ) and Kendall's correlation coefficient(Kendall's τ). We chose Spearman's ρ because Kendall's τ is more computational expensive than Spearman's ρ. Spearman correlation coefficient defines as,

$$\rho = \frac{\sum_{i=1}^{n}(r(x_i)-\bar{r})(r(y_i)-\bar{r})}{\sqrt{\sum_{i=1}^{n}(r(x_i)-\bar{r})^2)}\sqrt{\sum_{i=1}^{n}(r(y_i)-\bar{r})^2}}$$

where $r()$ is the rank of x_i, $\bar{r}$ is the mean of the ranks. We need to modify the Spearman's ρ because it requires the same number of sequences. We propose the modified Spearman's ρ,

$$\rho' = \frac{\sum_{x,y\in X\cap Y}(r(x)-\bar{r_x})(r(y)-\bar{r_y})}{\sqrt{\sum_{x\in X}(r(x)-\bar{r_x})^2}\sqrt{\sum_{y\in Y}(r(y)-\bar{r_y})^2}} \tag{7}$$

Since clustering often uses dissimilarity instead of similarity, we define $1-\rho'$ as dissimilarity. We shows an example in Fig.3. This dendrogram comes from Table.1. First, we compute the similarities between pairs in the capture list by using equation (7). Then, Ward's method(a method of hierarchical clustering) with the similarities outputs the tree.

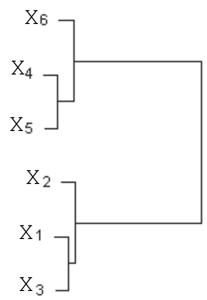
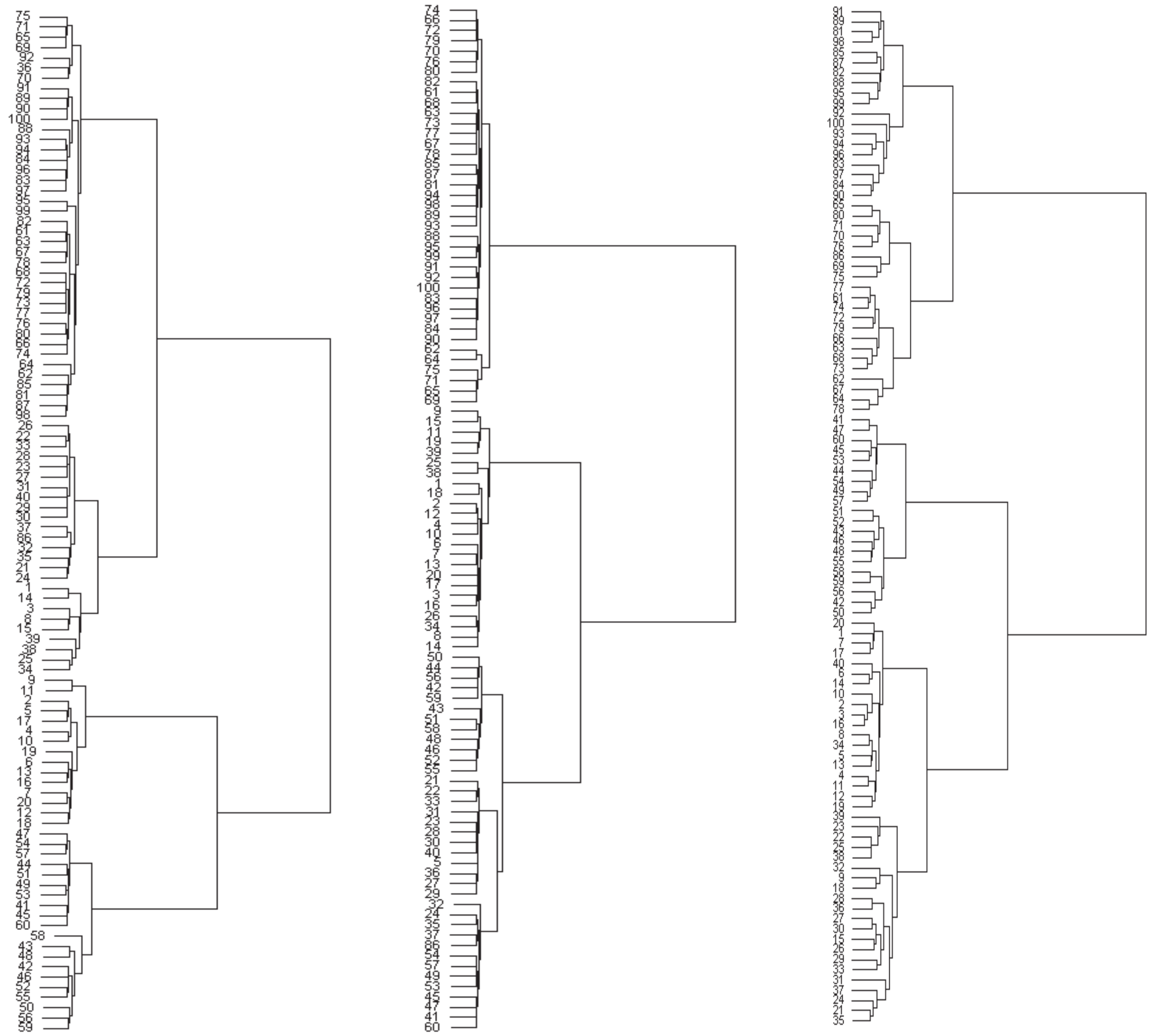


Fig. 3: A dendrogram from Table.1

4 Experimental Results

4.1 Dataset

We utilized the Australian sign language dataset[5] for our experiment and evaluated the similarity of spider algorithm comparing to Euclidean distance and dynamic time warping. Australian sign language is a clean preprocessed version of the sign language data, collected by Mohammed Waleed Kadous and located at the UCI KDD Archive. The dataset consists of various sensors that measure the positional information of a subject's right hand with a electronic glove. There are 10 signs {come, girl, man, maybe, mine, name, read, right, science, thank}. Each sign has 20 examples and all time series are the same length. We used three dimensional data, X-axis, Y-axis and Z-axis.



(a) (b) (c)

Fig. 4: Dendrograms, (a) Euclidean distance, (b) Dynamic time warping and (c) Spider algorithm

4.2 Evaluation Methodology

In general it is difficult to evaluate how well similarity performs for clustering. However, owing to the sign language data is labeled(ex.come,girl,man...), we could evaluate by using F-Measure. F-Measure has been used in natural language processing for evaluating clustering[6]. This measure compares how closely each hierarchy generated by the system matches a set of categories previous assigned the data by human judge. The label of human judge is not perfect in terms of clustering raw data, but in practice it captures the strengths and shortcomings of the algorithms. F-Measure defines as follows:

$$F = \frac{2 \cdot \frac{N_1}{N_2} \cdot \frac{N_1}{N_3}}{\frac{N_1}{N_2} + \frac{N_1}{N_3}} \qquad (8)$$

where N_1 is the number of data judged to be of label L in cluster A, N_2 is the number of data in cluster A and N_3 is the number of data judged to be of label L in entire hierarchy. If an algorithm make perfect clusters, then F-Measure become 1.

(8) is for each labeled data. The average F-Measure in all data is called Overall F-Measure defined,

$$Overall\ F = \frac{\sum_{L \in M}(|L| * F(L))}{\sum_{L \in M}|L|} \qquad (9)$$

where M is the set of labels, $|L|$ is the number of data judged of label L, and F(L) is the F-Measure for label L. We employed (9) as the evaluation measurement.

4.3 Clustering Results

In order to evaluate clustering for the similarities, we made a dendrogram by a conventional hierarchical clustering method. Many different clustering methods are available, however, we chose Ward's method which is said to tend to give realistic results. Fig.4 is the result of clustering five signs, girl, man, name, right and science. Each sign has 20 example, thus, 1-20 are the signs of girl,21-40 are man, 41-60 are name, 61-80 are right and 81-100 are science. This shows spider algorithm(c)[1]. is more accurate than Euclidean distance(a) or dynamic time warping(b).[2] Overall F-Measure of (a) is 0.82, (b) is 0.78 and (c) is 0.90 respectively. You can also see, the branch of the tree(c) is more clear than the other algorithms.

Table.2 shows the average of Overall F-Measure in every combination of sign languages. 2 signs means picking up two signs from ten signs, for example, come and girl, come and man, come and maybe and so on. Therefore, the total of combination is 45 in 2 signs. We computed Overall F-Measure in every combination and then calculated the average of them. This result told spider algorithm is same performance as the other algorithms in a small number of clusters, however, in a large number of clusters spider algorithm is superior than the others.

Table 2: Overall F-Mesure of each algorithm

	Euclidean	DTW	Spider Algorithm
2 signs($_{10}C_2$)	0.94	0.94	0.95
3 signs($_{10}C_3$)	0.89	0.89	0.89
4 signs($_{10}C_4$)	0.85	0.85	0.85
5 signs($_{10}C_5$)	0.83	0.81	0.83
6 signs($_{10}C_6$)	0.77	0.72	0.81
7 signs($_{10}C_7$)	0.78	0.71	0.80
8 signs($_{10}C_8$)	0.73	0.67	0.79
9 signs($_{10}C_9$)	0.75	0.70	0.78

[1]The parameters of spider algorithm as follows: the number of spider is half size of input number N, 30 generations, *r* is 1.0, θ is $1/\pi$, *f* is 0.8 and *m* is 0.1

[2]The similarities of Euclidean distance and dynamic time warping were calculated by the sum of each dimension.

5 Conclusions

The results show the similarity from spider algorithm can be a powerful tool for clustering. We recognize there are two shortcomings. First, spider algorithm is not deterministic. The similarity of spider algorithm makes good clusters, however, clusters is not always the same clusters. Second, it requires parameter setting. Although the choice of the parameters is difficult in general, there are many parameters in spider algorithm. We aimed at investigating the possibility of spider algorithm in this paper. For future works, we plan to examine tuning parameters or computational costs.

Euclidean distance is a simple and useful, however, it is used only as a measure. On the other hand, spider algorithm produces the spiders during clustering. Because each spider represents the time series , it is possible to find a key spider or a special spider for a certain application. We surely consider that keeping track of spiders is interesting in data stream[8] or other fields.

Acknowledgements: Thanks to the members of Yamamura laboratory in Tokyo Institute of Technology, we had fruitful discussions for this research.

References:

[1] Everitt, B. S.: *Cluster Analysis*, Edward Arnold, third edition, 1993.

[2] T.Warren Liao: Clustering of time series data – a survey, *Pattern Recognition.* 38, 2005, pp.1857-1874

[3] Keogh, E. J. and Pazzani, M. J.: Scaling up Dynamic Time Warping for Datamining Application, *Principles of Data Mining and Knowledge Discovery* 1704, 1999, pp.1-11

[4] Sakoe, H. and Chiba, S.: Dynamic programming algorithm optimization for spken word recognition. *IEEE Trans. Acoustics, Speech and Signal Process.* Vol.ASSP-26,43-49

[5] Keogh, E. and Folias, T.: The UCR Time Series Data Mining Archive Riverside CA. University of California - Computer Science & Engineering Department, 2002, http://www.cs.ucr.edu/ẽamonn/TSDMA/index.html

[6] Bjornar, L. and Chinatsu, A.: Fast and Effective Text Mining Using Linear-time Document Clustering. *Conference on Knowledge Discovery in Data* 1999, pp.16-22

[7] Kendall, M. and Gibbons, J. D.: *Rank Correlation Methods*, Oxford University Press, fifth edition 1990

[8] Barbará,D.: Requirements for Clustering Data Streams,*SIGKDD Explorations*, 2002 Vol.3, No.2, pp.23-27