Stock Time Series Categorization and Clustering Via SB-Tree Optimization

Tak-chung Fu[1,2,*], Chi-wai Law[1], Kin-kee Chan[1], Fu-lai Chung[1], and Chak-man Ng[2]

[1] Department of Computing, The Hong Kong Polytechnic University, Hong Kong
{cstcfu, c1516689, c1922434, cskchung}@comp.polyu.edu.hk
[2] Department of Computing and Information Management, Hong Kong Institute of Vocational Education (Chai Wan), Hong Kong
cmng@vtc.edu.hk

Abstract. SB-Tree is a data structure proposed to represent time series according to the importance of the data points. Its advantages over traditional time series representation approaches include: representing time series directly in time domain (shape preservation), retrieving time series data according to the importance of the data points and facilitating multi-resolution time series retrieval. Based on these benefits, one may find this representation particularly attractive in financial time series domain and the corresponding data mining tasks, i.e. categorization and clustering. In this paper, an investigation on the size of the SB-Tree is reported. Two SB-Tree optimization approaches are proposed to reduce the size of the SB-Tree while the overall shape of the time series can be preserved. As demonstrated by various experiments, the proposed approach is suitable for different categorization and clustering applications.

1 Introduction

A time series is a collection of observations made chronologically. Time series data can be easily obtained from scientific and financial applications, e.g., daily temperatures, daily sale totals, and prices of mutual funds and stocks. The nature of time series data include: large in data size, high dimensionality and update continuously. Indeed, a large set of time series data is from the stock market. Stock time series has its own characteristics over other time series data like ECG. For example, a stock time series is typically characterized by a few critical points and multi-resolution consideration is always necessary for long-term and short-term analyses. In addition, technical analysis is usually used to identify patterns of market behavior, which have high probability to repeat themselves. These patterns are similar in the overall shape but with different amplitudes and/or durations. Moreover, these patterns are characterized by a few data points. Based on such characteristics, a representation of the time series data is needed for manipulating the stock time series effectively and efficiently. Our previous work [1] is proposed to deal with this problem.

[*] Corresponding author.

The state-of-the-art time series representation schemes are mainly based on different dimensionality reduction techniques, including Principal Component Analysis (PCA), Discrete Wavelet Transform (DWT), Piecewise Linear Representation (PLR), Piecewise Aggregate Approximation (PAA) and Piecewise Constant Approximation (PCA). Based on these time series representation approaches, various classification and clustering applications have been proposed, e.g., applying PCA [2] and PLR [3,4] to classification and applying PCA [5], DWT [6] and a bit-level representation [7] to clustering. Furthermore, moving average has been proposed for clustering task in [8,9]. Reference [9] compares the clustering result based on different representation schemes, including DFT, DWT, PCA and the proposed ARIMA approach.

In this paper, the time series categorization and clustering tasks which take advantages of our previously proposed time series representation scheme, i.e. specialized binary tree (SB-Tree) [1], are described. SB-Tree is particularly effective in stock time series data. Based on optimizing the size of the SB-Tree, both stock time series categorization and clustering can be facilitated. The paper is organized into five sections. A brief review on SB-Tree, which is based on reordering the time series data points according to their importance, is given in section 2. Two approaches for optimizing the size of SB-Tree are proposed in this section. Section 3 introduces the categorization and clustering processes based on the optimized SB-Trees. The simulation results are reported in section 4 and the final section concludes the paper.

2 A Specialized Binary Tree Representation and Its Optimization

In this section, the SB-Tree structure for financial time series representation is briefly revisited. It is based on determining the data point importance in the time series. Then, the optimization approaches for this time series representation are proposed.

2.1 Specialized Binary Tree Data Structure

In view of the importance of extreme points in stock time series, the identification of perceptually important points (PIP) is firstly introduced in [10]. The frequently used stock patterns are typically characterized by a few critical points. These points are perceptually important in the human identification process and should be considered as more important points. The proposed scheme follows this idea by reordering the sequence *P* based on PIP identification, where the data point identified in an earlier stage is considered as being more important than those points identified afterwards. The distance measurement for evaluating the importance is the vertical distance (VD) [10].

After introducing the concept of data point importance, a binary tree (B-tree) structure has been proposed to store the time series data and is called specialized binary tree (SB-Tree) [1]. To create a SB-Tree, the PIP identification process [10] is adopted. A sample time series and the corresponding SB-Tree built are shown in Fig.1. The arc of the tree represents the VD of the corresponding node (PIP). Detail creating and accessing process of the SB-Tree can be found in [1].

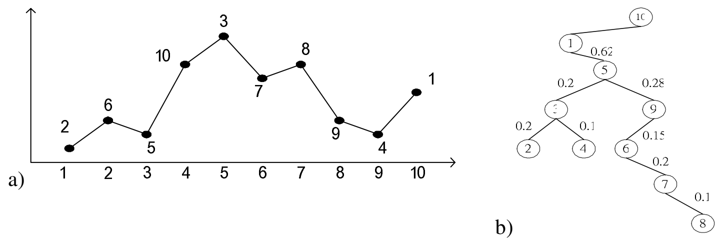

Fig. 1. (a) Sample time series and (b) the SB-Tree built

2.2 SB-Tree Optimization

After transforming the time series data into a SB-Tree, it is possible to further reduce the size of the tree so as to minimize the space consumption of a SB-Tree. It can be done by determining the number of PIP necessarily to represent the time series while the shape of the time series can still be preserved. If only a few more important PIPs are used to represent the whole time series, the error will be very large and the overall shape may also be deformed. Conversely, if all the PIPs are manipulated, the system performance will be very low.

The simplest way to reduce the size of the SB-Tree is applying a lossless pruning approach which only prunes the nodes with distance measured (VD) equal to 0. This kind of nodes has no effect on the shape of the time series because they are the data points located on the straight line formed by other PIPs only.

On the other hand, an acceptable level of error can be specified for which a large number of PIP can be filtered. Thus, a lossy approach is preferred for optimizing the SB-Tree to prune the "unnecessary" nodes of the SB-Tree. Error is defined as the mean square distance between the original time series and the series formed by n PIPs to represent the time series. In other words, the error is calculated by the linear interpolation between retrained points (i.e. PIPs from the optimized SB-Tree) and the original time series. Fig.2 shows the error when only 3 PIPs are used.

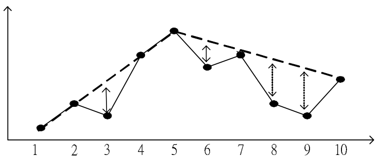

Fig. 2. Error of representing a time series with 3 PIPs compared to the original time series

Two optimization approaches are proposed below:

Tree Pruning Approach: Unimportant signals (i.e. data points) of a time series can be filtered according to a threshold λ. As the tree is accessed from the top and the VD of each node is considered. When the VD of a node is smaller than λ, the fluctuation does not vary a lot and the descendants are considered as less important to the users. Thus, this node and all its descendants should be pruned. Fig.3 shows the pruning of a sample SB-Tree using a threshold equal to 0.15 (i.e. λ =0.15).

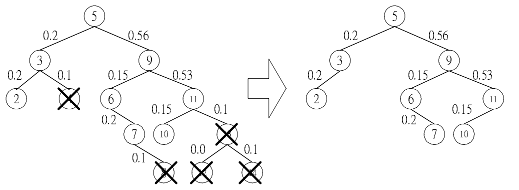

Fig. 3. Example of tree pruning approach using λ =0.15

However, it is difficult for the users to define the threshold λ. In addition, for different time series, different λ may be needed to preserve the general shape of the series. Therefore, an automatic approach for determining this threshold is necessary. It can be achieved by finding the natural gap of the change of VD along with the retrieval of PIPs from the SB-Tree. An example will be shown in section 4.

Error Threshold Approach: The second optimization approach is based on determining the error of the representation compared to the original time series. The PIPs are retrieved from the SB-Tree for representing the time series until the error is lower than a given threshold, α (cf. Fig.2). Again, it is necessary to determine the value of α. A reasonable α value is the point that has no significant decrease in the error. It can be determined by finding the largest decrease in error when adding one more PIP for representing the time series. By including such a PIP, one may come up with an optimized number of PIP for representing the time series as the decrease of error will be at a much lower level compared with the previous stages. Again, an example will be given in section 4.

3 SB-Tree Categorization and Clustering

After representing a set of time series by SB-Trees and determining the optimized tree sizes, it is possible to manipulate the set of optimized SB-Trees for different tasks. In this section, categorization and clustering of the stock time series based on SB-Tree representation are described.

3.1 Categorization

Class generation and time series pattern classification are the two main steps of the categorization process. First, the class generation process is introduced. The generated classes will be used to categorize/index similar time series (or subsequences). Each class is constructed by a class pattern. Two parameters are required to determine the class patterns. The first parameter is the range of number of PIP for building the class patterns, pip_{min} to pip_{max}. In our targeting domain, i.e. stock time series, the common technical patterns are always constructed by 4 to 7 data points. The second parameter is the number of point level, θ. After normalizing the time series, the distribution space (y-axis of the data point) will be equally divided by θ between 0 and 1.

The class patterns are generated by all the combinations of different data point values, such as {0.0, 0.0, 0.0, 0.0}, {0.0, 0.0, 0.0, 0.3}, {0.0, 0.0, 0.0, 0.6} and so on. The generation process carries out for each possible number of PIP, i.e. from pip_{min} to pip_{max}. The total number of classes that can be generated will be equal to $\sum_{n=pip_{\min}}^{pip_{\max}} \theta^n$.

Fig.4 shows some examples of class pattern using θ=3 for pip_{min}=5 and pip_{max}=7. As the patterns have been normalized before categorization, some class patterns are not necessary such as those class patterns with constant data point values, e.g. {0.00, 0.00, 0.00} and {0.33, 0.33, 0.33}. Therefore, these class patterns can be filtered.

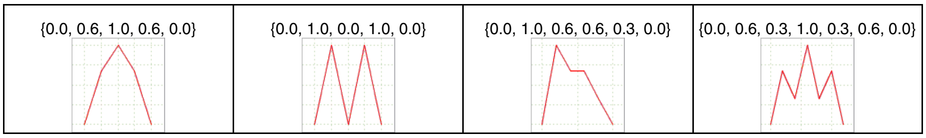

Fig. 4. Examples of class pattern

After a set of class patterns is generated, the time series represented by the optimized SB-Tree can be classified into one of the classes. The time series represented by the optimized SB-Tree will be compared with each class pattern with the same number of data points. The pattern will then be categorized to a class with the minimum distance. The simplest way to measure the distance (*dist*) is by the point-to-point Euclidean distance. Enhanced distance measure approach with horizontal distance consideration can also be applied [10].

Based on this categorization process, different applications can be considered. For example, users can select any interested classes based on the class patterns and investigate on the time series segments belonging to the corresponding classes. Another application is to speed up the time series pattern query process. Given a query time series pattern and a threshold δ, the query pattern will be first compared to all the class patterns, which have the same number of data points. If the distance is greater than δ, the whole class will be ignored; otherwise, the optimized SB-Trees in this class will be compared with the query pattern. If the distance between them is also less than δ, it will be selected as one of the query results.

3.2 Clustering

Clustering is a common approach for finding structure in the given data, in particular for finding structure related to time. There are many popular clustering techniques developed, such as hierarchical, nearest neighbor, and *k*-means algorithms. In the data mining context, the most common one perhaps is the *k*-means algorithm. In the *k*-means clustering, each cluster is represented by the center of the cluster called centroid. The *k*-means algorithm can be implemented with four steps: (1) cluster objects into *k* nonempty subsets, (2) compute seed points as the centroids of the clusters of the current partition, (3) assign each object to the cluster with the nearest seed point and (4) go back to step 2 and stop when no more new assignment is observed or the maximum number of iterations is reached. Suppose that there exist N objects, $x_1, x_2, \ldots x_N$, in the dataset and they fall into k compact clusters, $k<<N$. Let M_i be the mean of the vectors, i.e., the cluster centroid, in cluster i. If the clusters are well separated, a minimum distance classifier can be used to separate them.

To cluster a set of time series, the time series are the input objects. However, they may have different lengths and one might have to transform the time series data into other representation which provides the same number of features for clustering. By reducing the size of the SB-Trees using the proposed optimization process and based on the assumption that the common technical patterns are always constructed by 4 to 7 data points, the variation in the length of the time series can be greatly reduced. In other words, time series represented by the optimized SB-Tree will provide similar number of feature vectors (from 4 to 7 in this case). Moreover, the number of PIP required to represent the time series after optimization already provides the preliminary information for clustering, i.e., the patterns represented by 5 PIPs are fundamentally different from the patterns represented by 7 PIPs (see Fig.5a). Therefore, a two-step clustering approach is adopted here as shown in Fig.5b. First, the set of time series is clustered by the number of PIP in the corresponding optimized SB-Trees, ranging from pip_{min} to pip_{max}. Then, the subset of time series with the same number of PIP will be further clustered by the *k*-means clustering process with the same number of features and therefore, each of the time series will be grouped into one cluster in one of the *k*-means clustering processes. The point-to-point Euclidean distance can be applied to compute the distance between the time series and the cluster centroid.

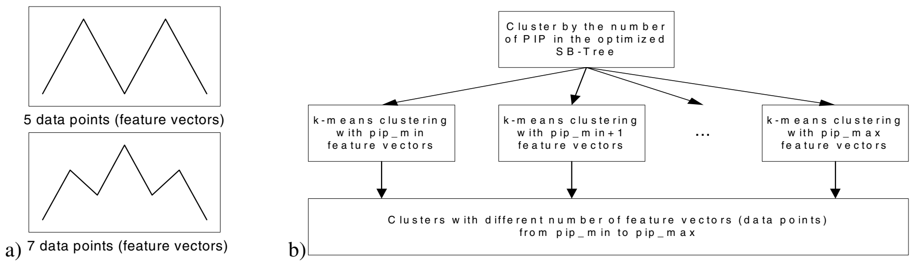
5 data points (feature vectors)
a) 7 data points (feature vectors)
b)
Cluster by the number of PIP in the optimized SB-Tree
k-means clustering with pip_min feature vectors
k-means clustering with pip_min+1 feature vectors
...
k-means clustering with pip_max feature vectors
Clusters with different number of feature vectors (data points) from pip_min to pip_max

Fig. 5. (a) Time series patterns with different numbers of PIP and (b) A two-step clustering process

In this section, the usages of the optimized SB-Trees in time series categorization and clustering have been described. As the unimportant points of the time series are filtered after the optimization process, time series with different lengths, but with similar shape, will be represented by the same number of PIP for different data analysis tasks.

4 Experimental Results

In this section, three sets of experimental results are reported. First, the performance of the two SB-Tree optimization approaches is shown. Then, the applications of the optimized SB-Tree to categorization and clustering are evaluated. A synthetic time series dataset, which consists of 110 time series with different lengths (25, 43 and 61 data points) was used. Each of them belongs to one of the five technical patterns: head-and-shoulder (H&S), double tops, triple tops, rounded top and spike top. Each technical pattern was used to generate 22 variants by applying different levels of scaling, time wrapping and noise. First, the patterns are uniform time scaling from 7 data points to 25, 43 and 61 data points. Then, each critical point of the patterns can be warp between its previous and next critical points. Finally, noise is added to the set of patterns. Adding noise is controlled by two parameters, namely, the probability of adding noise for each data point and the level of noise being added to such point.

4.1 SB-Tree Optimization

The optimized numbers of data point for representing the five technical patterns are: H&S=7, double tops=5, triple tops=7, rounded top=5 and spike top=5 as the corresponding variants were generated based on the primitive patterns constructed by these numbers of PIP. However, after investigating the generated patterns, it is reasonable to represent the rounded top pattern by either 4 or 5 data points. Therefore, both 4 and 5 PIPs were considered as correct for representing the rounded top pattern in the following experiments.

Taking the H&S pattern as an example, Fig.6a shows the VD values for different numbers of PIP. By using the tree pruning approach, Fig.6b indicates that the optimal number of PIP can be obtained by locating the PIP number with peak change of VD. The dotted lines in the figures show the optimized size of the SB-Tree (i.e. the number of PIP) identified for representing the time series. Fig.7 shows the result using the error threshold approach. Both approaches could identify the correct number of PIP, i.e., 7 PIPs.

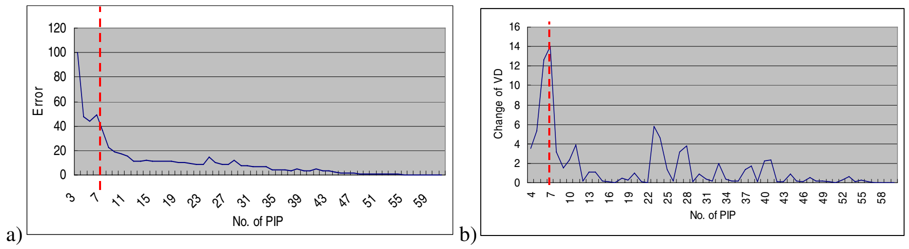

Fig. 6. (a) The VD curve with different numbers of PIP retrieved by applying the tree pruning approach and (b) The change of VD with different numbers of PIP. The peak value corresponds to the optimal number of PIP.

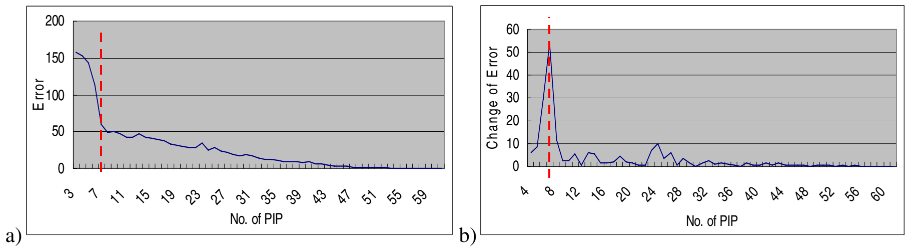

Fig. 7. (a) The error curve with different numbers of PIP retrieved by applying the error threshold approach and (b) The change of error with different numbers of PIP. The peak value corresponds to the optimal number of PIP.

Table 1 compares the accuracy and processing time of the two optimization approaches. Accuracy here means the correct number of PIP identified by the optimization process. According to Table 1, the accuracy of the error threshold approach outperformed the tree pruning approach. However, the time consumed by the error threshold approach was 1/3 higher than that of the tree pruning approach. It is because from 3 PIPs till obtaining the correct number of PIP, the error between the original time series and the time series constructed by the selected PIPs has to be calculated correspondingly. On the other hand, the low accuracy of the tree pruning approach was due to the over pruning of the SB-Tree based on determining the largest gap among the VD. Furthermore, the accuracy of the proposed optimization approaches is independent from the length of the time series. Fig.8 shows three sample time series with incorrect optimized size.

Table 1. Comparisons on the two optimization approaches

	Tree Pruning	**Error Threshold**
Length of time series = 25		
Accuracy	72.22%	97.30%
Processing Time	0.09s	0.12s
Length of time series = 43		
Accuracy	62.63%	94.74%
Processing Time	0.28s	0.49s
Length of time series = 61		
Accuracy	72.86%	100.00%
Processing Time	0.65s	1.19s
Overall		
Accuracy	69.09%	97.27%
Processing Time	1.02s	1.80s

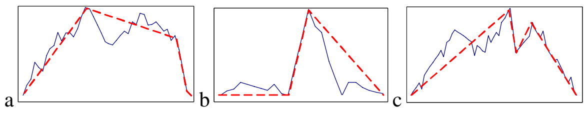

Fig. 8. Wrong optimization cases: (a) resulting from both approaches, (b) resulting from the error threshold approach and (c) resulting from the tree pruning approach

4.2 SB-Tree Categorization

The 110 optimized SB-Trees based on the error threshold approach were used in both the categorization and clustering tasks. During the categorization process, the first step is to generate the class patterns. The range of the number of PIP used for representing the class patterns was 4 to 7 and the point level, θ, was 3. The numbers of class pattern generated after filtering are 110, 570, 2702 and 12138 according to the numbers of PIP are 4, 5, 6 and 7 respectively.

After the categorization process, the 110 optimized SB-Trees or time series took up 20 classes (out of the total 15520 classes), i.e. with an average of 5.5 time series in a class. The implication here is that the subsequent query process can be limited to only one or a few classes in order to save the processing time. At the very beginning of this section, we mentioned that the 110 syntactic time series were generated from 5 technical patterns. They are expected to be assigned to only 5 classes. It is not the case because similar class patterns exist. As exemplified in Fig.9, the 4 class patterns found should belong to the triple tops pattern. However, this is not an undesired behavior as the users can choose to search in both these classes or not during the query process. Furthermore, by investigating these class patterns visually, users can determine which classes they are interested in.

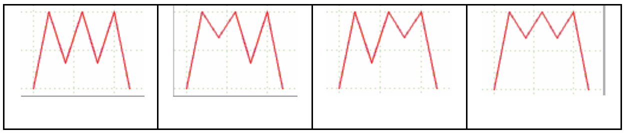

Fig. 9. Class patterns that should belong to the sample technical pattern

4.3 SB-Tree Clustering

In the last experiment, the proposed time series clustering process was simulated. After the first-step clustering, i.e. clustering by the number of PIP in the optimized SB-Tree, the 110 synthetic time series were clustered into 3 groups as shown in Table 2. Both the H&S and triple tops patterns were clustered to the group with 7 PIPs. The correctly optimized SB-Trees of the spike top and double tops were clustered to the group with 5 PIPs. The problem occurred in the rounded top pattern as both 4 and 5 PIPs for representing it was considered as correct. In the first-step clustering process, the accuracy is the same as the optimization process (i.e. 97.27% in this case) and it can be easily seen that the incorrect patterns here come from double tops (1) and spike tops (2) patterns and both were over optimized to 4 PIPs.

Table 2. Number of time series in the clusters with different numbers of PIP

No. of PIP	4	5	6	7
H&S	0	0	0	22
Double Tops	1	21	0	0
Triple Tops	0	0	0	22
Rounded Top	15	7	0	0
Spike Top	2	20	0	0
No. of SB-Tree	18	48	0	44

Then, the *k*-means clustering process was further carried out for each number of PIP. It was benchmarked with the clustering process using other representation scheme, i.e. the Piecewise Aggregate Approach (PAA) [11]. By using PAA, the dimension of the time series will be reduced to the same as the minimum length of the time series in the dataset (i.e. 25 in this experiment). The result shows that there are fewer incorrect groupings of patterns by using the proposed method (i.e. accuracy=91.8%) than the benchmarked approach (i.e. accuracy=75.5%). Based on the finding from the experiment, one may suggest to introduce a redundancy removal process to group similar clusters. First, it is necessary to combine the clusters in each number of PIP when *k* is larger than the number of actual groups. Also, it is necessary to combine the clusters across different numbers of PIP that are representing the same pattern, such as the rounded top pattern in this experiment. It exists in both 4 and 5 PIPs clusters. This is one of our current works.

5 Conclusion

In this paper, two approaches for optimizing the size of the SB-Tree are proposed. By carrying out the optimization process, the size of the SB-Tree can be reduced under the control of the user preference while the overall shape of the time series can be preserved. In other words, dimensionality reduction of the time series can be achieved. The proposed approach is customized for representing stock time series based on its unique properties. Furthermore, the usages of the optimized SB-Tree are demonstrated by two applications, namely, categorization and clustering. A comprehensive evaluation on the proposed approaches is now conducting on real dataset and will be reported in the near future.

References

1. Fu, T.C., Chung, F.L., Luk, R. and Ng, C.M.: A specialized binary tree for financial time series representation. The 10th ACM SIGKDD Workshop on Temporal Data Mining (2004) 96-104
2. Geurts, P.” Pattern extraction for time series classification. In Proc. of the 5th PKDD (2001) 115-127
3. Smyth, P. and Keogh, E.: Clustering and mode classification of engineering time series data. In Proc. of the 3rd Int.l Conf. on KDD (1997) 24-30
4. Keogh, E. and Pazzani, M.: An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In Proc. of the 4th Int. Conf. on KDD (1998) 239-341
5. Abonyi, J., Feil, B., Nemeth, S. and Arva, P. Principal component analysis based time series segmentation - Application to hierarchical clustering for multivariate process data. In Proc, of the IEEE Int. Conf. on Computational Cybernetics (2003) 29-31
6. Lin, J., Vlachos, M., Keogh, E. and Gunopulos, D.: Iterative incremental clustering of time series. In Proc. of the 9th EDBT (2004) 106-122
7. Ratanamahatana, C.A., Keogh, E., Bagnall, A.J. and Lonardi, S.: A novel bit level time series representation with implications for similarity search and clustering. Technical Report, UCR, TR-2004-93 (2004)
8. Xiong, Y. and Yeung, D.Y.: Mixtures of ARMA models for model-based time series clustering. In Proc. of ICDM (2002) 717-720
9. Kalpakis, K., Gada, D. and Puttagunta, V.: Distance measures for effective clustering of ARIMA time-series. In Proc. of ICDM (2001) 273-280
10. Chung, F.L., Fu, T.C., Luk, R. and Ng, V.: Flexible Time Series Pattern Matching Based on Perceptually Important Points. International Joint Conference on Artificial Intelligence Workshop on Learning from Temporal and Spatial Data (2001) 1-7
11. Keogh, E., Chakrabarti, K., Pazzani, M. and Mehrotra, S.: Dimensionality reduction for fast similarity search in large time series databases. JKIS (2000) 263-286