World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:8, 2009

Clustering Categorical Data Using Hierarchies (CLUCDUH)

Gökhan Silahtaroğlu

***Abstract*—**Clustering large populations is an important problem when the data contain noise and different shapes. A good clustering algorithm or approach should be efficient enough to detect clusters sensitively. Besides space complexity, time complexity also gains importance as the size grows. Using hierarchies we developed a new algorithm to split attributes according to the values they have and choosing the dimension for splitting so as to divide the database roughly into equal parts as much as possible. At each node we calculate some certain descriptive statistical features of the data which reside and by pruning we generate the natural clusters with a complexity of O(n).

***Keywords*—**Clustering, tree, split, pruning, entropy, gini.

I. Introduction

WITH the increase of data stored in databases, data mining and a part of it clustering accelerated its development and many new models and techniques have been added to the literature in this field. In general, algorithms are classified as partitional and hierarchical models. In partitioning methods *n* objects are formed into k different pre-assigned clusters (k<n). The number of clusters to be constructed is known before hand. K-means, K-medoids, PAM, CLARA and CLARANS are examples of partitioning algorithms [1]. Some density based algorithms such as DBSCAN [2], OPTICS [3] and DENCLUDE [4] form clusters analyzing the density occupied by the objects in data space. To form a hierarchy there are two techniques; they are bottom up –agglomerative- and top down divisive approaches [5]. The bottom up approach or agglomerative algorithms see each object in the database as a separate, individual cluster. Putting all these objects (clusters) together based on any known similarity or distance measure, at the end they create clusters different from each other. Single-linkage or nearest neighbor method, complete-linkage method, average linkage method and centroid methods, Chameleon algorithm, BIRCH are some of them [6]. On the other hand, divisive algorithms which follow a top down approach, at the beginning of the scan see the database as a cluster, and then drilling down they partition the data into different clusters [7], [8]. While Polythetic methods use all of the variables to form splits, monotheistic methods use one variable only for successive splits [9]. As they use similarity and or distance measures they are easy to use and apply. But it is a disadvantage to give the number of clusters especially for divisive algorithms.

Neural Networks and Genetic algorithms are other methods of clustering [10], [11].

Creating tree to partition the data into clusters is used by Zhang, et.al. with BIRCH algorithm. They use CF tree to form clusters and it constructs clusters with one scan , collecting and using some descriptive statistical parameters [12]. In CF tree the root represents the whole database, another tree approach may be employed as in classification which is another branch of data mining; we use a decision tree to determine the possible natural clusters in the database. In our approach, the root does not represent the whole database as it does in BIRCH algorithm; however it is the most significant dimension of the whole database. By analyzing the database through the relative significance of the dimensions we reach the natural minimal clusters.

G. Silahtaroğlu is with Beykent University, Department of Mathematics and Computing, Istanbul 34900, Turkey (e-mail: gsilah@beykent.edu.tr).

II. Background

A. Clustering

Clustering is grouping items in database according to their similarity just like in classification. However, these groups are not predefined as it is in classification. We may define clustering as follows [5].

Definition1. Given a database $D = \{t_1, t_2, \ldots, t_n\}$ of tuples and an integer value k, the clustering problem is to define a mapping $f : D \to \{1,...,k\}$ where each t_i is assigned to one cluster K_j, $1 \le j \le k$. A cluster, K_j, contains precisely those tuples mapped to it; that is,

$$K_j = \{t_i \mid f(t_i) = K_j, 1 \le i \le n \quad and \quad t_i \in D\}.$$

Some of the clustering ways are Hierarchical Clustering, Partitional Clustering and Binning.

B. Hierarchical Clustering

This method examines all the items in the database one by one and handles each of them as separate clusters. The method recursively combines clusters by updating the intercluster distances. There are a lot of algorithms using this technique, they differ from each other how they create the sets. Generally, a tree data structure, called a dendrogram, is used to show the hierarchical clustering. The root of the dendrogram represents the whole database as a single cluster. Each leaf of the dendrogram gives a cluster with different

features. Uniting and merging those leaves depends on the distance between them and that is measured with centroid, radius and diameter parameters.

Centroid, radius and diameter of for a cluster is well-known parameters [12]. Given N d-dimensional data points in a cluster $\{x_i\}$ where *i = 1,2,…, N* the centroid C_0, radius R and the diameter *D* of the cluster are defined as:

$$C_0 = \frac{\sum_{i=1}^{N} x_{mi}}{N} \quad (1)$$

$$R = \sqrt{\frac{\sum_{i=1}^{N} (x_{mi} - X_0)^2}{N}} \quad (2)$$

$$D = \sqrt{\frac{\sum_{i=}^{N}\sum_{j=1}^{N} (x_{mi} - x_{mj})^2}{N(N-1)}} \quad (3)$$

These parameters are used to measure the closeness of the found clusters, and help us to unite some certain clusters or not. *R* is the average distance from member points to the centroid, *D* is the pair wise distance within a cluster [12].

As it is well known the distance between two points may be measured by Euclidean distance as

$$dis(X_m, X_j) = \sqrt{\sum_{i=1}^{n} (x_{mi} - x_{ji})^2} \quad (4)$$

Another approach is to determine the similarity between objects as well as clusters [13]. Dice, Jaccard and Cosine are some of well known similarity measures [14].

$$sim(x_m, x_j)_{dice} = \frac{2\sum_{i=1}^{n} x_{mi} x_{ji}}{\sum_{i=1}^{n} x_{mi}^2 + \sum_{i=1}^{n} x_{ji}^2} \quad (5)$$

$$sim(x_m, x_j)_{jaccard} = \frac{\sum_{i=1}^{n} x_{mi} x_{ji}}{\sum_{i=1}^{n} x_{mi}^2 + \sum_{i=1}^{n} x_{ji}^2 - \sum_{i=1}^{n} x_{mi} x_{ji}} \quad (6)$$

$$sim(x_m, x_j)_{\cos ine} \frac{\sum_{i=1}^{n} x_{mi} x_{ji}}{\sqrt{\sum_{i=1}^{n} x_{mi}^2 \sum_{i=1}^{n} x_{ji}^2}} \quad (7)$$

We use any of the measures mentioned above to determine the distance between formed natural clusters if they needed to be united.

Within these techniques there are mainly two types of algorithms: Agglomerative and Divisive Algorithms.

Agglomerative algorithms start with considering each point in the dataset as a 'single member cluster' and iteratively merge them until all items group in one cluster and there is enough space among the clusters. Hierarchical clustering is an ideal method when it is used with a reasonable distance metric, however it is not used widely because of its $O(n^2)$ complexity.

C. Partitional Clustering

This technique creates the clusters in one step putting each point in a random or guessed clusters and moving the items from one cluster to another until some local minimum is found. Euclidean metric Eq.(4) may be used to measure the distances between formed clusters. One of the drawbacks of the technique is that the number of the clusters should be given by users. Another problem is that the technique suffers the possibility of different clustering solutions. Searching all possible clustering alternatives would not be feasible [5]. To go around this problem, minimum distance, minimum similarity and/or the number of the clusters are taken from the user as input. This is against the spirit of clustering since in the definition of the clustering it says 'unlike classification the number of the clusters and the items of the clusters are not known at the beginning' [19].

D. Binning

The technique splits the space into some number of bins and forms clusters spotting the bins which contain a number of data points higher than defined by the user. Determining the bin boundaries is one of the difficulties of the technique; another one is that especially multi dimensional spaces require a great deal of bins. Determining this parameter may upset the accuracy.

III. Related Work

A. Balanced Iterative Reducing and Clustering using Hierarchies

There are many algorithms to find clusters using a dendrogram. One them is *Balanced Iterative Reducing and Clustering using Hierarchies* algorithm (BIRCH) [12]. It is suitable for very large databases. BIRCH incrementally clusters incoming multi-dimensional metric data points to try to produce the best quality clustering. Basically a tree or dendrogram is built to get all needed information to perform clustering.

In BIRCH algorithm a CF tree is created [12]: A CF tree is a height balanced tree with two parameters: branching factor B and threshold T. In this algorithm, each node, excluding the leaves, contains at most b entries of the form. It is defined as [12].

Definition 2. Given N d-dimensional data points in a cluster: $\{X_i\}$ where *i = 1, 2, …,N*, the Clustering Feature (CF) vector of the cluster is defined as a triple: CF = ($N, \vec{LS}, SS$), where N is the number of data points in the cluster, $\vec{LS}$ is the linear sum of the N data points, and SS is the square sum of the N data points, i.e., $\sum_{i=1}^{n} \vec{X}^2_i$.

So, a non-leaf node represents a cluster made up of all sub-clusters represented by its entries. A leaf node has got maximum L number of entries and each node contains two pointers, "pre" and "next" which are used to link all of the leaf nodes for an efficient scan. A leaf node represents a cluster made up of all sub-clusters represented by its entries. In this approach each leaf node should satisfy a threshold value calculated trough radius, diameter or centroid. The tree size may be another constraint. The tree holds values N number of data points in the cluster, $\vec{LS}$ the linear sum of the N data points $\sum_{i=}^{n} \vec{X}_i$ and SS which represents square sum of the N data points $\sum_{i=1}^{n} \vec{X}^2{}_i$.

From the CF definition and additivity theorem, it is clear that the CF vectors of clusters can be stored and calculated incrementally and accurately as clusters are merged [12].

Root node represents the whole database and each leaf represents a separate cluster or sub-cluster depending on the T threshold value which is the diameter of the expected cluster.

BIRCH is an alternative way of hierarchical clustering, however, it is applied on continuous data only.

B. Tree Based Classification

A tree approach is used for classification which is another model of data mining [15]. To reach the classes through the shortest path the most suitable attribute is chosen to be the root and recursively the algorithm make calculations to determine the most suitable attribute in the dataset to be the next node. Here 'the most suitable' is to divide the rest of the database roughly into two or more equal parts. The attribute which is the closest to this is chosen as the most suitable attribute.

In literature there are different approaches to form the tree. One is to use entropy concept. ID3 and C4.5 algorithms use *entropy* to find the node representatives [16]. If we represent the probabilities as $<p_1, p_2,\ldots,p_n>$ then the sum equals to 1.

$$\sum_{i=1}^{n} p_i = 1 \tag{8}$$

In this case, entropy is

$$H(\mathrm{p}_1, \mathrm{p}_2, ..., \mathrm{p}_n) = \sum (p_i \log(1/p_i)) \tag{9}$$

ID3 selects the nodes with a gain value

$$Gain(D,S) = H(D) - \sum_{i=1}^{n} P(D_i)H(D_i) \tag{10}$$

The attribute which yields the highest gain is chosen as the root or the next node.

C4.5 divides this gain value by splitting information which is calculated as:

$$Split(D,S) = H\left(\frac{|D_1|}{|D|},...,\frac{|D_s|}{|D|}\right) \tag{11}$$

Another approach is to use *gini* index as it is in SLIQ algorithm [17],[18].

$$gini(K) = 1 - \sum p_j^2 \tag{12}$$

Using the equation below the most suitable attribute to split the database two equal (or more) parts is roughly found.

$$gini_{split}(K) = \frac{n_1}{n_2} gini(K_1) + \frac{n_2}{n_2} gini(K_2) \tag{13}$$

IV. Our Contribution

BIRCH algorithm is used for continuous values to form the tree. Our aim is to form a tree which will yield quality clusters for the data which hold categorical variables. Besides, in BIRCH algorithm the root node represents the whole data in the database; however our approach is closer to *entropy* or *gini* index approaches in which starting from the root data are split into clusters.

Indeed, to split data starting from the very beginning of the tree (root) and in each node throughout the branches we determine clusters and sub clusters. Each node including the root is calculated so as to split the database on the best point to create equal parts.

In order to achieve this we use a new algorithm.

To determine the best attribute we use an equal-split parameter (EP) Eq. 14 and Eq. 15.

$$EP = \sum_{i=1}^{n} |CA - NV_i| \tag{14}$$

$$CA = \frac{N}{NV} \tag{15}$$

Input:
D: Data // Get Data

Output
K: Clusters

Step 1:
For each data atribute **do**
NV = number of variables
CA = N / NV // *Center is calculated*
For i = 1 to NV
NV_i = number of each variable
EP = CA - NV_i
End
For i = 1 to NV
EP = abs(CA - NV_i) + EP // *EP value is calculated for each attribute*
End
Loop
If leaf is not reached
go to step 1
Else
Step2:
Prune the tree according to the given initial criteria.
End If
STOP

Fig. 1 Tree based clustering algorithm for categorical data

Where N represents the sum of all records in the attribute, NV is the number of variables, NV_i is the number of each variable.

As it is depicted in Fig. 1 Clustering Categorical Data Using Hierarchies (CLUCDUH) algorithm is a recursive algorithm which re-calculates EP for each branch and its nodes separately. Minimum EP value is taken as the root or the node to be added to the tree. The best split value for EP is zero. This may occur in situations like when half of the values in the attribute are yes and the other half consist of Nos; when all the values in an attribute are the same value such as yes EP = 0 again. So, this attribute should be skipped in this kind of situations.

As the tree is built for each node including the leaves, number of data which satisfies the conditions of the nodes, centroid and diameter of the formed clusters in nodes, and standard deviation of the cluster are calculated.

Since our algorithm produces natural clusters the tree will need a pruning. We use threshold values for pruning like the one used in BIRCH algorithm. Starting from the leaves if any leaf is below the threshold value given by the user that leaf joins its sibling. Thus we reduce the number of the natural clusters produced by the algorithm.

After forming the clusters we use another clustering algorithm such as PAM, CLARA or K-Means to reduce the number of leaves to the number required by the user.

Sample Application 1:

Table I depicts a three dimensional database i.e. colorful, size, tail.

TABLE I
SPECIES

COLORFUL	*SIZE*	*TAIL*
YES	SMALL	NO
YES	SMALL	NO
YES	SMALL	NO
YES	MEDIUM	YES
YES	SMALL	YES
YES	SMALL	YES
YES	SMALL	YES
NO	BIG	NO
NO	BIG	NO
NO	BIG	NO
NO	BIG	NO
NO	BIG	NO

Here,

N = 12,

For *colorful* dimension;

NV = 2 as NV_{yes} and NV_{no} .

NV_{yes} = 7 and NV_{no} = 5. Thus,

From Eq.14 and Eq.15,

$CA_{colorful}$ = 6;

$EP_{colorful}$ = $\left|6-7\right| + \left|6-5\right| = 1 + 1 = 2$

For *size* dimension

NV = 3; NV_{small} = 6, NV_{medium} =1 and NV_{big} =5. Thus

CA_{size} = 12/4 = 3.

EP_{size} = $\left|4-6\right| + \left|4-1\right| + \left|4-5\right| = 6$.

For *tail*,

NV = 2 as NV_{yes} = 4 and NV_{no} = 8.

CA_{tail} = 6

EP_{tail} = $\left|6-4\right| + \left|6-8\right| = 4$.

So, *colorful* dimension with min EP = 2 will be assigned as the root.

The rest of the tree and the nodes will be formed in the same way using the rest of the database.

V. PERFORMANCE ANALYSIS

We generated several synthetic datasets depending on some pre-assigned parameters in order to do a series of performance

analysis of the algorithm. Synthetic datasets have been generated as it is depicted in Table I.

For the test a Pentium Dual Core 2.0 GHZ CPU PC with a 1GB RAM has been used.

TABLE II
SYNTHETIC DATASETS

Dataset	*No. Records*	*No .of Fields*	*Original No. of Clusters*	*Time (Seconds)*	*No .of Clusters (Detected)*
DS1	64.000	6	4	13.2	4
DS2	75.000	8	3	16.1	3
DS3	250.000	7	4	43.2	4

It was clear that the clusters formed artificially in the data sets beforehand were determined by the algorithm successfully. If the clusters are not separated from each other naturally, the user should not give any interfering stopping criteria to the algorithm and this does not waste the running time of the algorithm. Nevertheless, giving stopping criteria such as tree depth, support and confidence level may affect the quality of the clusters detected if they are not separated from each other remarkably. To go around this difficulty we calculated the interior distances and the distance between clusters at each node using Eq.(2), Eq.(3) and Eq.(4). It is clever to give interior distance as a stopping criterion.

We also applied the algorithm for different data shapes and received remarkably quality results. As it is well-known outlier is important for data mining algorithms. Thanks to the pruning and the nature of the approach, outliers are eliminated within our algorithm.

TABLE III
COMPARATIVE PERFORMANCE TEST WITH CLARA AND BIRCH

Dataset	Error Rate BIRCH	Error Rate CLARA	Error Rate CLUCDUH	Clustering Time – CLUCDUH (Seconds)	Clustering Time – BIRCH (Seconds)	Clustering Time – CLARA (Seconds)
DS1	0.013	0.010	0.11	13.2	18.6	19.5
DS2	0.014	0.012	0.12	16.1	24.2	35.4
DS3	0.013	0.010	0.11	43.2	55.2	90.9

Clustering Categorical Data Using Hierarchies (CLUCDUH) algorithm does not need to rescan the dataset, so the complexity is O(n). We also have comparative performance analysis with BIRCH and CLARA algorithms. The same datasets, DS1, DS2 and DS3 have been used for the comparison. Results are depicted in Table II.

It is clear that especially from the time complexity point of view CLUCDUH does better than CLARA and with a less error rate in comparison with BIRCH.

Sample Application 2: This application is aimed to test the performance of the algorithm. For this purpose we used a synthetic database name Synthetic Control Chart Time Series which contains 600 examples of control charts synthetically generated by the process [23]. There are six different classes of control charts [20].

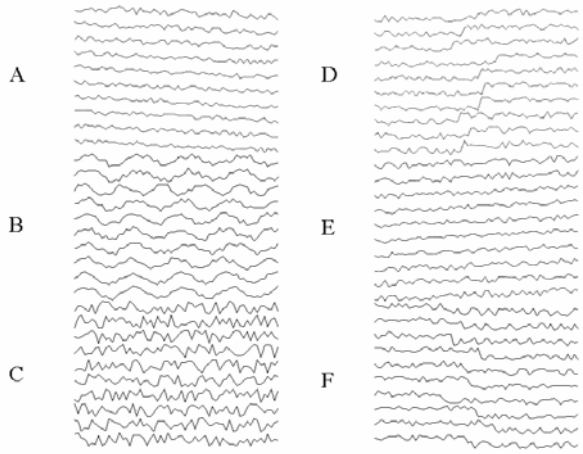

Fig. 2 Examples from each class

Fig. 2 depicts ten examples from each class labeled from A to F.

A. Normal
B. Cyclic
C. Increasing trend
D. Decreasing trend
E. Upward shift
F. Downward shift

The reason to choose this data set to test for clustering algorithm is that Euclidean distance will not be able to achieve perfect accuracy, so the following pairs of classes will often be confused (Normal/Cyclic) (Decreasing trend/Downward shift) and (Increasing trend/ Upward shift) [21]. Before the algorithm was run, the data had been categorized into ten different categories with binning method, so values vary between 1 and 10. Our algorithm -a tree approach to clustering data has been tested and compared with the results derived from Euclidean Distance clustering and Derivative Dynamic Time Warping. Fig. 3 shows a subset of the results for grouped average hierarchical clustering on this dataset. In the original form each class is represented by a distinct color and they look different from each other as depicted in Fig. 2. The left column shows the results by using Euclidean Distance and the right column shows the results for Derivative Dynamic Time Warping [22] and our results are depicted in Fig. 4. As it is seen in Fig. 4, the algorithm found the six pre-assigned classes as different clusters correctly. The same correct results cannot be held when BIRCH algorithm is used since it is designed for continuous values and does not work on categorical values. The result we have produced is also

very close to the one produced with Derivative Dynamic Time Warping.

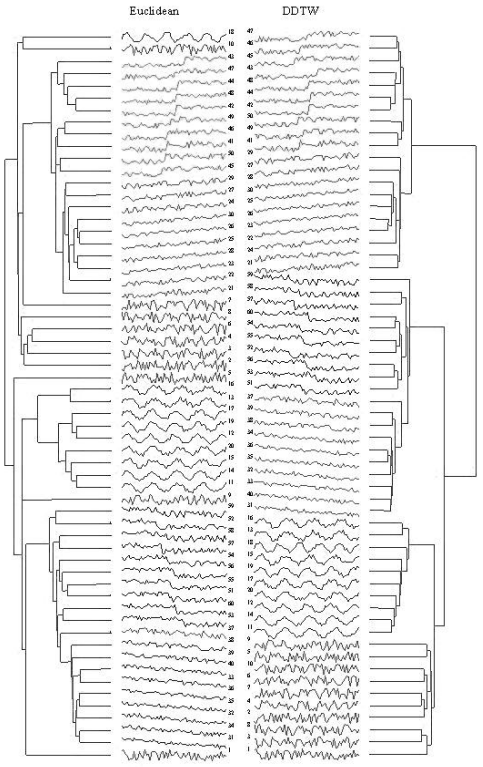

Fig. 3 Results by using Euclidean distance and derivative dynamic time warping [22]

In our case, the original values were continuous and converted to categorical data in order to test the proposed algorithm, because it is easier to produce synthetic data with pre-assigned clusters with continuous data. However, in real life there are plenty of categorical data to need to be clustered.

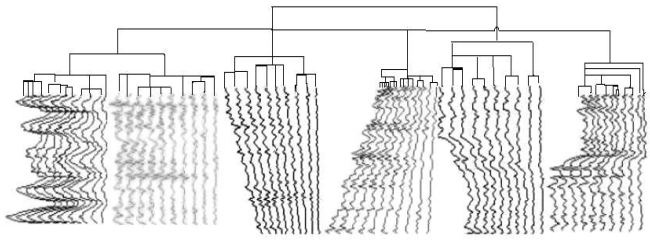

Fig. 4 Clustering results produced by the algorithm

REFERENCES

[1] Raymond T Ng. & Jiawei Han. (1994). Efficient and Effective Clustering Methods for Spatial Data Mining, Proceedings of 20th International Conference on Very Large Data Bases, Santiago de Chile, (pp. 144 – 155). Morgan Kauffmann.

[2] Ester Martin, et. al. (1996). A Density Based Algorithm for Discovering Clusters in LargeSpatial Databases with Noise, Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (pp. 169- 194). Kluwer Academic Publishers.]

[3] Ankerst Mihael, et.al. (1999) OPTICS: Ordering Points to Identify the Clustering Structure, Proceedings of ACM SIGMOD (pp. 5761 -5767). Pergamon Press.

[4] Hinneburg, Alexander and Keim, Daniel A. (1998). An Efficient Approach to Clustering in Large Multimedia Databases with Noise, Proceedings of Knowledge Discovery and Data Mining (pp. 58 -65). AAAI Press.

[5] Han J., & Kamber, Micheline. (2001). Data Mining Concepts and Techniques, Morgan Kaufman Publishers Academic Press.

[6] Karypis, George, et.al. (1999). CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling, Poceedings of IEEE COMPUTER, V.32, (pp. 68 – 75). IEEE Computer Society Press.

[7] Duda R. & Hart P. E. (1973). Pattern Classification and Scene Analysis, Wilry.

[8] Kauffman, L., & Rousseeuw P.J. (1990), Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley and Sons.

[9] Fisher Douglas H.(1995). Iteraritive Optimisation and Simplification of Hierarchical Clusterings, Technical Report CS-95-01, Vabderbilt University.

[10] Fausett L. (1994). Fundamentals of Neural Networks, Prentice-Hall, New Jersey.

[11] Maulik U. & Sanghamitra B.(2000). Genetic Algorithm-based clustering technique, Journal of the Pattern Recognition, Pergamon, issue: 33.

[12] Zhang Tian et.al. (1996). BIRCH: An Efficient Data Clustering Method for Very Large Databases, Proceedings of ACM International Conference on Management of Data, (409 – 418). Oxford University Press.

[13] Kreyzig E.(1989). Introductory Functional Analysis With Applications, Wiley.

[14] Bill F. (Ed.) (1992). Information retrieval: data structures & algorithms. Prentice Hall.

[15] Mitchell T.(1997). Machine Learning, McGraw-Hill International.

[16] Quinlan,J.Ross. (1987). Simplifying decision trees, International Journal of Man-Machine Studies,issue: 27(3), (pp. 221 – 234).

[17] Breiman L., & Friedman J. H., & Olshen R. A., & Stone C. J. (1984). Classification and Regression Trees, Wadsworth, Belmont.

[18] Mehta M., & Agrawal R., & Rissanen J. (1996). SLIQ: A Fast Scalable Classifier for Data Mining, Proceedings of 5th International Extending Database Technology Conference.France. (pp. 18-32). Springer-Verlag, London.

[19] Agrawal R. & Shafer J.C. (1996). Parallel Mining of Association Rules, Proceedings. of IEEE Transactions on Knowledge and Data Engineering, Vol. 8, No. 6. (962- 969). IEEE Educational Activities Department. USA.

[20] Hettich, S. , & Bay, S. D. (1999). The UCI KDD Archive, Department of Information and Computer Science, University of California, Irvine, CA. Retrieved September 1, 2008, from http://kdd.ics.uci.edu.

[21] Pham D.T., & Chan A.B.(1998). Control Chart Pattern Recognition using a New Type of Self Organizing Neural Network. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering. Vol 212, No 1, (pp. 115-127). Professional Engineering Publishing.

[22] Keogh, E. & Pazzani, M. (2001). Derivative Dynamic Time Warping. In First SIAM International Conference on Data Mining (SDM'2001), Chicago, USA.

[23] Alcock R.J. & Manolopoulos Y. (1999). Time-Series Similarity Queries Employing a Feature-Based Approach. 7th Hellenic Conference on Informatics. Ioannina,Greece.